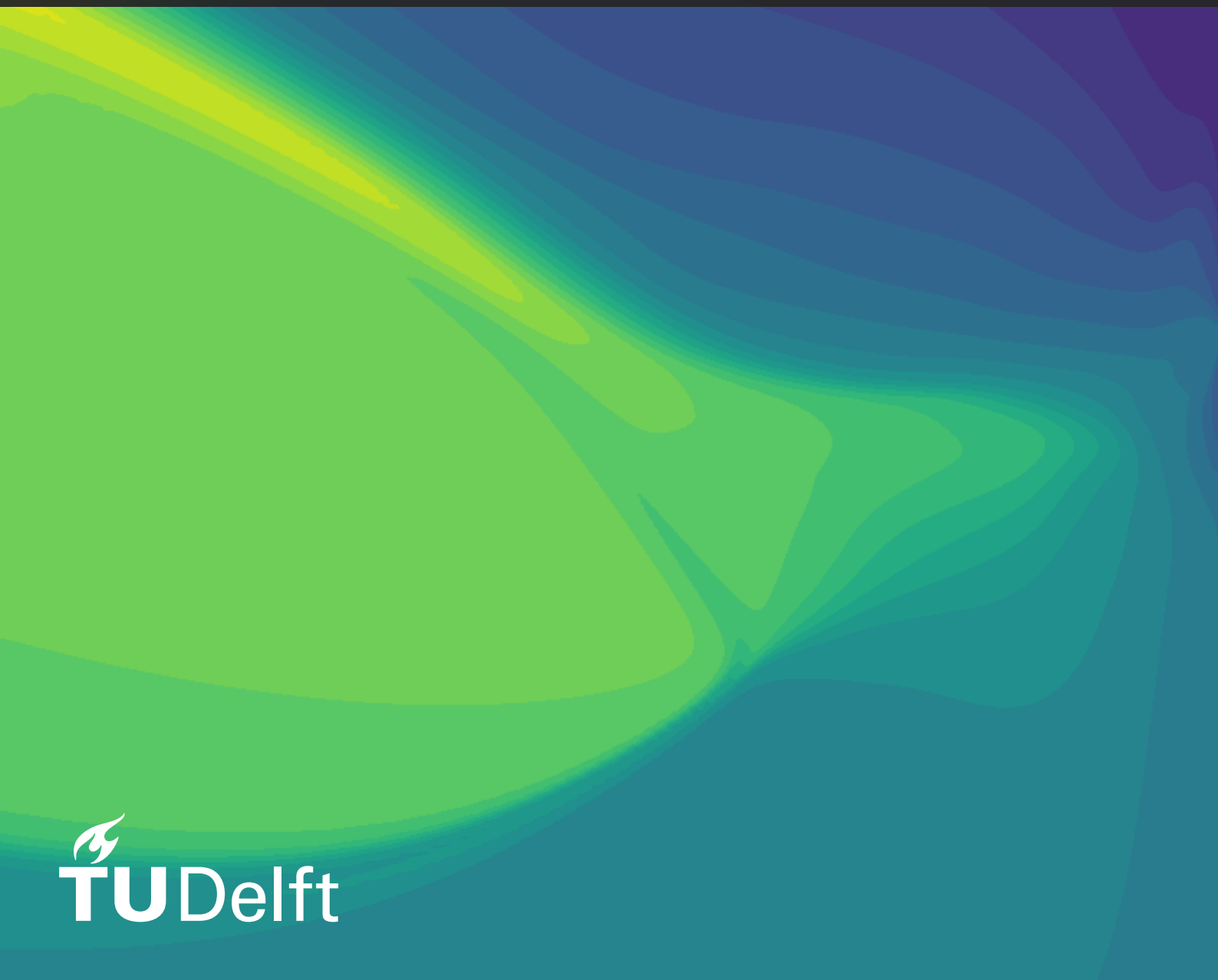
Mixture fraction enhanced physics-informed neural networks for flow reconstruction in pool fires

Master of Science Thesis Report

Pablo González Martínez



TU Delft

Mixture fraction enhanced physics-informed neural networks for flow reconstruction in pool fires

Master of Science Thesis Report

by

Pablo González Martínez

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on May 21, 2024 at 10:00

Student number:	4582675	
Project duration:	May 2023 — May 2024	
Thesis committee:	Dr. N.A.K. Doan	(Supervisor)
	Dr. A. Sciacchitano	(Chair)
	Dr. I. Langella	(External examiner)
Place:	Faculty of Aerospace Engineering, Delft University of Technology	

TU Delft

Acknowledgments

Many years ago, before I started my journey in Delft, I recall watching an online lecture by Professor Steve Brunton about the use of machine learning techniques to model fluid dynamics. I did not realize this at the time, but seeing the possibilities that exist at the intersection of these two fields, both of which I am fascinated by, was an eye-opening moment that motivated me to pursue my studies in Aerospace Engineering and to carry out my Master's degree in the Aerodynamics group specifically.

I feel thankful to have had the chance to work on precisely this topic for my Master's thesis. The result of this work, carried out over the last 12 months, is summarized in this report. I would like to thank my supervisor, Dr. Anh Khoa, for his guidance during the project. A special mention also goes to Dr. Philip Sitte. His feedback and advice have been incredibly helpful to navigate the issues I have encountered during this thesis, in particular those related to combustion, a topic that I was not familiar with prior to this work.

In addition, I want to use this space to express gratitude to those people that have played a crucial role in my journey, and without whom this work could not have been completed. I have to start with my mom. It is because of your effort, your sacrifice, and the constant attention and care you gave Maya and I that I stand here today. You have been a source of constant support, and you taught me to strive for great things and to demand excellence from myself while teaching me to be kind to myself and to others. I want to thank my sister, Maya, for being my number one supported, my best friend, and a constant source of laughter and joy. I am incredibly proud of you, and your support means the world to me. I also want to thank my grandmother Carmen, for teaching me invaluable life lessons and for reminding me that life is not so serious after all. To my father and my late grandparents, I want to say thank you your constant love and the discipline you have instilled in me. And to everyone else in my family — Tía Esther, Guille, la familia de la Arnía, y todos los demás. Gracias por vuestro cariño, vuestros consejos, y la sensación de estar en casa que siento cuando estamos juntos. A los que ya no estáis, os llevo conmigo siempre y os pienso a menudo.

Finally, I want to thank my friends and the many wonderful people I have encountered during my time in Delft. It all began in Foulkes quite some time ago, and we have come a long way since then. Many people have come and go. It is hard to put into words the impact that you have had in my life during these years. You guys know me better than anyone else in the world. You have taught me many lessons about the type of person I want to become, and I look up to you and remind myself that I am truly blessed to be surrounded by people like you. From the bottom of my heart I want to thank you for being yourselves and for helping me find who I am.

Lastly, to the interested reader, I hope you find this report useful and insightful. And if you plan on working with PINNs, I wish you luck!

Pablo González Martínez
Delft, May 2024

"Truth... is much too complicated
to allow anything but approximations."

— JOHN VON NEUMANN (1947)

Abstract

This thesis explores the use of physics-informed neural networks (PINNs) to reconstruct the flow fields in a pool fire flame, a canonical configuration in non-premixed combustion. Due to the difficulty in obtaining adequate experimental characterizations of such flows, reacting flows like pool fires stand in need of novel methods capable of reconstructing the principal flow features from limited measurement data.

The present work extends the capabilities of previous PINN reconstruction frameworks through the addition of a new physical prior that embeds information about the flame structure: the mixture fraction. This passive scalar quantifies the local state of mixing in diffusion flames, playing a crucial role in their analysis.

A novel reconstruction framework that incorporates the mixture fraction has been developed. The mixture fraction is found to be a valuable addition to the reconstruction framework: it either reduces the amount of flow variables required to obtain successful velocity reconstructions, leads to higher reconstruction accuracies, alleviates some of the PINN's well-known failure modes, or a combination of the above. Furthermore, it allows for adequate first order estimates of the heat release rate (HRR) of the flame even in the absence of pressure, density, and temperature data.

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

Abbreviation	Definition
AD	Automatic differentiation
BFGS	Broyden–Fletcher–Goldfarb–Shanno algorithm
BOS	Background-oriented Schlieren
CFD	Computational fluid dynamics
CFL	Courant–Friedrichs–Lewy number
CNN	Convolutional neural network
CPINN	Competitive physics-informed neural network
CPU	Central processing unit
DGNN	Dual graph neural network
DL	Deep learning
DMD	Dynamic mode decomposition
DNS	Direct numerical simulation
DSC/MS	Downsampled skip-connection/multi-scale model
ERA	Eigenvalue realization algorithm
FFNN	Feed-forward neural network
FPGA	Field programmable gate array
GDP	Gross domestic product
GNN	Graph neural network
GOBAL	Goal oriented basis learning
GPU	Graphical processing unit
HRR	Heat release rate
ICA	Intracranial aneurysm
k-SVD	K-means singular value decomposition
L-BFGS	Limited-memory BFGS
LDV	Laser Doppler velocimetry
LES	Large eddy simulation
LSTM	Long short-term memory (network)
ML	Machine learning
MLP	Multi-layer perceptron
MRI	Magnetic resonance imaging
MSE	Mean squared error
NN	Neural network
NTK	Natural tangent kernel
PDE	Partial differential equation
PINN	Physics informed neural network
PIV	Particle image velocimetry
PLIF	Planar laser induced fluorescence
POD	Proper orthogonal decomposition
PTV	Particle tracking velocimetry
RANS	Reynolds-averaged Navier-Stokes
RNN	Recurrent neural network
ROM	Reduced order model
SciML	Scientific machine learning
SD	Shallow decoder
SiLU	Sigmoid linear unit
UQ	Uncertainty quantification

Symbols

Symbol	Definition	Unit
a	Pool radius	m
C	Measurement operator	-
C_L	Liquid heat capacity	J kg^{-1}K^{-1}
C_p	Specific isobaric heat capacity	m^{2}s^{-2} K^{-1}
D	Diameter	m
$D_{T,A}$	Thermal diffusivity of air	m^{2}s^{-1}
$\mathcal{D}$	Mass diffusivity	m^{2}s^{-1}
f	Frequency	Hz
g	Gravitational acceleration	ms^{-2}
Δh_f	Specific enthalpy of formation	J kg^{-1} or J mol^{-1}
h_s	Specific sensible enthalpy	J kg^{-1} or J mol^{-1}
$\mathcal{L}$	Neural network loss	-
L_f	Flame length	m
L_v	Latent heat	Jkg^{-1}
$\dot{m}$	Mass flow	kg s^{-1}
s	Stoichiometric mass ratio	-
p	Pressure	kg m^{-1}s^{-2}
r	Radial coordinate	m
$\mathcal{R}$	Universal gas constant	J mol^{-1}K^{-1}
T	Temperature	K
T_s	Surface temperature	K
T_0	Initial temperature	K
u	Axial velocity	ms^{-1}
v	Radial velocity	ms^{-1}
$\boldsymbol{V}$	Velocity vector	ms^{-1}
Y_k	Mass fraction of species k	-
Y_F	Fuel mass fraction	-
Y_O	Mass fraction of oxidizer	-
z	Axial coordinate	m
α	Entrainment coefficient	-
β	Convection coefficient	-
μ	Dynamic viscosity	kg m^{-1} s^{-1}
ν	Kinematic viscosity	m^{2}s^{-1}
ν_A	Kinematic viscosity of air	m^{2}s^{-1}
ξ	Mixture fraction	-
ξ_{st}	Stoichiometric mixture fraction	-
ρ	Density	kg m^{-3}
τ_c	Characteristic chemical timescale	s
τ_f	Characteristic flow timescale	s
ϕ	Equivalence ratio	-
Ψ	Training library (sparse representations)	-
ω	Vorticity	rad/s
Da	Damköhler number	-
Le	Lewis number	-
Pr	Prandtl number	-
Re	Reynolds number	-

1 Introduction

Fluid flows are complex. Their complexity arises from the remarkable paradox that lies at the heart of fluid dynamics — despite having a near-perfect model to describe the behavior of fluid flows, the Navier-Stokes equations, finding accurate solutions within this framework remains an open scientific problem and an area of active research. In the last 50 years significant progress has been made in the field of computational fluid dynamics, simulating flows with ever-increasing levels of fidelity. A wide range of numerical methods for the study of flows have been developed over the years, including finite difference methods, finite element methods based on weak formulations, finite volume methods and spectral methods, among many others.

Despite their success, these methods are unable to seamlessly incorporate the abundance of multi-fidelity data that is readily available today. Recent developments in experimental measurement techniques and the constant improvements in our ability to simulate fluid flows mean that we now produce more data than at any other point in history. With more than a trillion sensors expected to be deployed in the next decade [1], it is evident that our ability to generate and collect data will only continue to increase in the near future. In the field of fluid dynamics, this trend presents a unique opportunity to construct novel methods that can leverage the information contained in these datasets to improve our understanding of flow physics and our ability to simulate such flows. So-called data-driven methods are now being applied to construct simple and accurate models useful for design, control, and optimization tasks in fluid dynamics [2].

One area of research where data-driven models can be particularly useful is *flow field reconstruction*, the process of reconstructing a flow field from limited data, obtained numerically or experimentally. Being able to reconstruct coherent flow structures from limited (and possibly corrupt) real-world data can be critically enabling for applications such as active flow control, faster design optimization loops, and uncertainty quantification. These applications are relevant across many domains including wind energy, robotics, cardiac blood flow modeling, climate science, oceanography, and geophysics, among others [3–5]. Flow field reconstruction is an inverse problem that remains prohibitively expensive to solve using traditional numerical methods. And even more novel data-driven reconstruction methods suffer from some key limitations: they are sensitive to noise, prone to overfitting, and require large volumes of data to achieve adequate reconstruction accuracies [6].

Recently, new reconstruction methods that make use of deep learning techniques have emerged as a promising alternative to overcome the limitations of traditional methods. One of the most promising methods relies on a novel machine learning algorithm known as a *physics-informed neural network* (PINN). PINNs embed knowledge about the underlying physical system, typically in the form of a system of partial differential equations, directly into the neural network architecture. This physics-based regularization is generally introduced as a soft constraint through the loss function of the PINN, which typically takes the form of a multilayer perceptron (MLP).

PINNs have been shown to be much better suited than traditional numerical methods for solving inverse problems like flow field reconstruction [1, 3, 5]. However, most of the existing research has focused

on simple geometries with laminar, low-Reynolds number flows that can be described by a Boussinesq approximation. Very little work has been done on their application to reconstruct multiphysics and multiscale flows with large density variations. Such flows are ubiquitous in many real-life engineering problems and their reconstruction using PINNs, or any other technique for that matter, remains an open problem and an area where further research is needed [1].

This thesis aims to bridge this gap by using PINNs to reconstruct the velocity field in a *pool fire* flame, a canonical flow configuration for the study of non-premixed combustion. The reason for selecting pool fires as the target for the reconstruction is twofold. On one hand, the application of PINNs to multiphysics problems is a necessary step to develop PINN-based reconstruction algorithms into a practical engineering tool applicable to real world problems. Secondly, and more importantly, reacting flows like pool fires are notoriously difficult to study experimentally [7]. The harsh environments they create difficult the data collection process and hinder our ability to obtain detailed flow field characterizations experimentally. Additionally, low-cost numerical simulations of these flows are generally hard to achieve due to their complexity [8]. Consequently, reacting flows like pool fires stand in need of novel methods capable of reconstructing flow quantities that cannot be easily measured experimentally. PINNs have emerged as a promising tool to perform this task. Their physics-based regularization can guide the network towards an accurate reconstruction of the flow field even in the absence of measurement data.

This work aims to extend the capabilities of PINNs for flow field reconstruction in pool fires by incorporating an additional physical prior — the *mixture fraction*. This scalar parameter is crucial in the analysis of diffusion flames, quantifying the local state of mixing within them and embedding information about the variation of thermochemical properties like temperature and composition. By integrating the mixture fraction, this thesis seeks to advance PINNs as a robust reconstruction method that can augment limited experimental datasets, mitigating a pressing issue in the study of non-premixed combustion.

Research objectives

In short, the research objective of this thesis is:

> To extend the capabilities of physics-informed neural networks as a flow field reconstruction method in pool fire flames, focusing on the introduction of new physical constraints via the mixture fraction.

In order to tackle this objective the following research questions have been formulated:

1. **How can existing PINN-based reconstruction algorithms be extended to successfully incorporate the mixture fraction?** — In order to use the mixture fraction to enhance the capabilities of PINNs a new reconstruction framework needs to be developed, tested, and verified. This framework should include an appropriate formulation of a governing physical law to embed information about the flame structure through the mixture fraction.

2. **What is the effect of the mixture fraction on the reconstruction process, and how can we use it to improve the reconstruction accuracy?** — It is not clear what the role of the mixture fraction should be within the reconstruction process. This thesis explores the effect of the mixture fraction when used as an additional target for the reconstruction and as a data source to guide the velocity reconstruction process, assessing the effect on the reconstruction accuracy in each of the two cases.

3. **How can the mixture fraction be used in the search for minimal data solutions to the reconstruction problem?** — Minimal data solutions are defined as successful reconstructions that rely on a minimal amount of data to be performed. From a practical standpoint it is highly desirable to reduce the amount of variables that need to be provided to the PINN to obtain adequate velocity

reconstructions. This thesis explores how the information embedded by the mixture fraction can be exploited to find new minimal data solutions.

4. **How can the mixture fraction be used to estimate the Heat Release Rate (HRR) of the pool fire flame, and what reconstruction accuracy can we obtain?** — The addition of the mixture fraction into the reconstruction framework may provide an opportunity to estimate the HRR, a key parameter of interest in the study of combustion flows. This thesis attempts to perform the first HRR reconstructions of a pool fire flame using PINNs, and explores the reconstruction accuracy that can be obtained while doing so.

Thesis outline

This thesis is structured as follows. Chapter 2 introduces pool fires as a canonical flow configuration in the study of non-premixed combustion, highlighting the difficulties in their experimental study and the need for new flow field reconstruction methods capable of reconstructing missing quantities. The concept of the mixture fraction is also introduced, discussing its role in the study of non-premixed combustion and motivating its use to enhance the reconstruction algorithm. Chapter 3 provides background theory on flow field reconstruction methods, including traditional techniques and state of the art methods that leverage the power of deep learning, contextualizing the use of PINNs as a reconstruction method. Chapter 4 then focuses on physics-informed neural networks themselves, explaining their typical architecture, use cases, and their advantages over more traditional flow field reconstruction methods. Chapter 5 presents the methodology used in this thesis, including the formulation of the mixture fraction equation and the method used to estimate the HRR. Chapter 6 then presents the results of the thesis, discussing a series of reconstruction attempts perform to answer the aforementioned research questions. Chapter 7 concludes the findings of this thesis, and finally Chapter 8 provides recommendations for future work.

Part I

Background

2

Pool fires

This chapter is devoted to pool fires. Section 2.1 motivates their study by highlighting the importance of pool fires as a canonical configuration for non-premixed combustion. A detailed characterization of pool fires is then given in Section 2.2, including a description of their anatomy and the key physical mechanisms that characterize them. Then, Section 2.3 discusses experimental studies of pool fires. Particular attention is placed on the inherent difficulty to obtain adequate experimental data for these flows, highlighting the need for flow field reconstruction. Lastly, Section 2.4 introduces the concept of the mixture fraction. Motivation for its use a tool to extend the reconstruction algorithm is provided, and its role in the computation of diffusion flame problems is discussed.

2.1. The role of pool fires

Uncontrolled fire is a great destroyer of lives and property. At the end of the 20th century the total annual cost of fire to the developed nations of the world was estimated to be approximately 1% of their combined Gross Domestic Product (GDP) [9]. More recent estimates put this number as high as 2% of GDP [10]. In the United States alone, the cost of fires amounted to $328.5 billion in the year 2014, equal to 1.9% of the GDP [11]. Similar figures are reported in the European Union [12].

In addition to the large financial burden associated with fire-related incidents, there also exists a significant human cost. The World Health Organization estimates over 300,000 annual casualties from fire-induced burns, most of them in developing nations with poor fire safety practices and standards [13]. Current population estimates by the United Nations highlight the likely rise of population density in urban areas in the coming decades, a phenomenon that could significantly increase the occurrence and potential damage of fire related incidents [14].

The are many different types of fire and fire-related incidents that can occur, including wildfires, accidental industrial fires, and transportation fires. Wildfires, particularly relevant today, have increased in frequency and severity since the 21st century [15], significantly impacting regions like western North America, southeastern Australia, and the Mediterranean [16]. These fires are estimated to account for up to 20% of global greenhouse gas emissions, a figure that could grow to 30% by the end of the century [17]. Fire safety research is crucial in order to prevent and mitigate the economic, human, and environmental losses caused by uncontrolled fires like wildfires. By studying the behavior of fires, researchers aim to develop better models to predict the occurrence of fire-related incidents, to aid policy and decision making, and to develop critical fire safety practices that lead to increased safety and damage mitigation.

Fires, however, are notoriously difficult to predict and analyze, since they can occur in any situation where fuel, oxidizer, and an ignition source coexist. For the phenomenological study of fire, it is desirable to study configurations that have reduced geometrical complexity while maintaining relevance to real applications. For that reason, fire research has traditionally focused on two key canonical configurations. The first one are wall-bounded reacting boundary layers, and the second are free-field reacting plumes. The latter category is also sometimes referred to as *pool fires*. These canonical

flows have provided the basic understanding needed to construct empirical correlations and numerical models applied to a practical fire problems [18]. In other words, the study of fundamental flow configurations like pool fires is vital to the development of tools and methods that can be used to model real fires.

The term *pool fire* is broad and can encompass many different fires. A detailed definition is given in Section 2.2 — here it suffices to say that a pool fire is a buoyancy-driven diffusion flame that forms over a horizontal fuel surface, generally in liquid form, hence the use of the term *pool*. The resulting reacting plume is the most commonly studied type of free flow for fires [18] due to its broad applicability to many real fires. Pool fires exhibit the basic phenomena that are applicable to many fires ranging from the small laminar flame of a typical candle lighter, all the way up to a medium scale industrial fire or a highly-turbulent large scale wildfire extending over hundreds of kilometers. Due to its broad applicability their study has both theoretical and practical significance for fire researchers. The study of pool fires allows researchers to deepen their understanding on the physical mechanisms that drive non-premixed combustion processes. The interest in the subject is reflected by the large amount of studies with different focus and complexity, ranging from basic studies investigating the effect of the pool diameter on the flame height to much more detailed studies of the flame structure and soot composition [18].

From a practical point of view, pool fires are a great building block in the modeling of more complex fires. The unsteady puffing motion present in pool fires has been shown to be relevant for the study of wildfire spread [19, 20]. Pool fires are also useful in the study of mass fires, which occur when the fuel is too dispersed and/or when there are multiple fuel or ignition sources available. Such fires can be thought of as a conflagration made up of multiple flames, each modelled as a pool fire. Pool fires have also been used as an ignition source to model transportation fires such as those in railway vehicles [21, 22]. Additionally, liquid sodium pool fires are relevant for the design of sodium cooled nuclear reactors [23]. Pool fires have also been used to model the effects of accidental fires in offshore oil and gas facilities [24] and as a model fire in fire-suppression research [25].

In conclusion, the study of pool fires holds substantial practical and theoretical value for the fire research community.. The pool fire serves as a great problem to study the effect of different parameters on the flow topology, such as the pool diameter and the fuel composition. Simultaneously, the wide applicability of pool fires to many real fires further motivates their study. Investigating pool fires is essential to understand and predict the dynamics of complex combustion processes in practical settings.

2.2. Characterization of pool fires

As mentioned previously, the definition of the term pool fire is not straightforward. The term is used somewhat loosely in the scientific community — there is no straightforward definition of what constitutes a pool fire, and it can only be described based on its main characteristics [7].

In general, it can be stated that a pool fire is a buoyancy-driven diffusion flame that forms and stabilizes over a horizontal fuel surface, generally in the form of a liquid hydrocarbon, although the fuel can also be a solid. Diffusion flames like pool fires, also known as non-premixed flames, receive their name due to the fact that they are driven by diffusion — the vaporized fuel diffuses into the oxidizer, mixing them and leading to combustion. In the case of a liquid fuel it mixes with the oxidizer through vaporization; in the case of a solid fuel through pyrolysis. It should be noted that although diffusion is the dominant transport mechanism that governs the mixing of fuel and oxidizer, convection also contributes to the mixing process. Pool fires are further characterized by zero or low initial momentum of the vaporized fuel stream, as opposed to jet plumes [26].

The basic anatomy of a pool fire is shown in Figure 2.1. Three clearly discernible regions can be observed in the structure of the flame. At the bottom of the fire, near the fuel pool, there is a permanently luminous flame. Unburned fuel vapors emanating from the fuel pool are present in this region. Air is radially entrained into the flame, driving the mixing process between fuel and oxidizer and sustaining the combustion process. The key measurable parameter in this base region is the amount of fuel that is burnt off, which can be measured by the heat release rate (HRR). The HRR is a measure of the rate of heat generation by a fire. It is an important parameter that can be used to estimate the

destructive power of a pool fire [7].

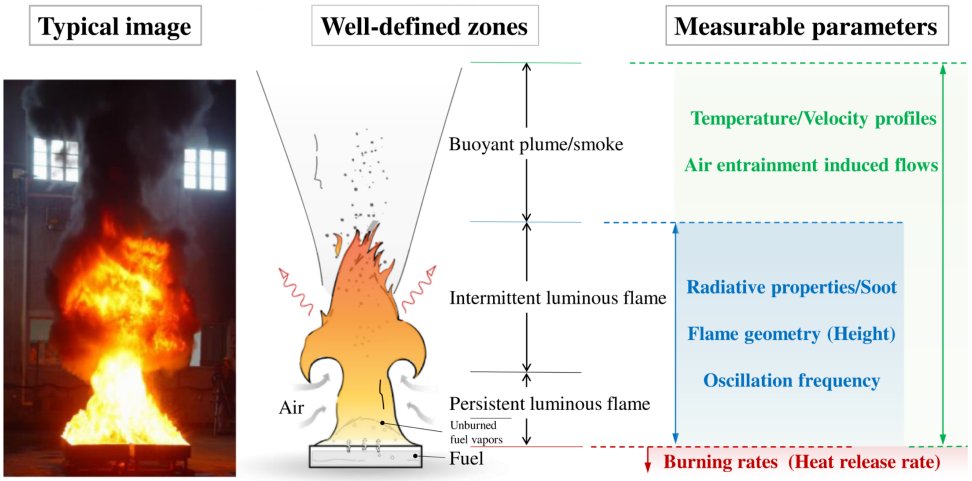


Figure 2.1: The anatomy of a typical medium-scale turbulent pool fire. A typical image for such a fire in shown on the left. A schematic representation of the flame structure, subdivided into three clearly discernible regions, is shown in the middle. The key measurable parameters that correspond to each of these regions is shown on the right. Reprinted from [27].

On top of the base region there is an intermittently luminous flame which may exhibit an unsteady oscillatory flickering known as puffing, shown in Figure 2.2. This region can be characterized through the oscillation frequency of the puffing behavior, the flame geometry (mainly the flame height), and the radiative properties and composition of the soot. The bottom two sections of the flame exhibit a characteristic red or yellow color typical of most fires. This is the consequence of the unburnt soot particles that are glowing due to the high temperatures inside of the flame. As the soot diffuses upwards it eventually reaches the boundary of the flame. At the boundary the mixing of fuel and oxidizer occurs at stoichiometric conditions. At this condition is, approximately, when the burning of the fuel and oxidizer occurs [8]. Therefore, it is only near the flame boundary that the burning process takes place — inside of the flame the soot is still unburnt and glowing hot due to the high temperatures.

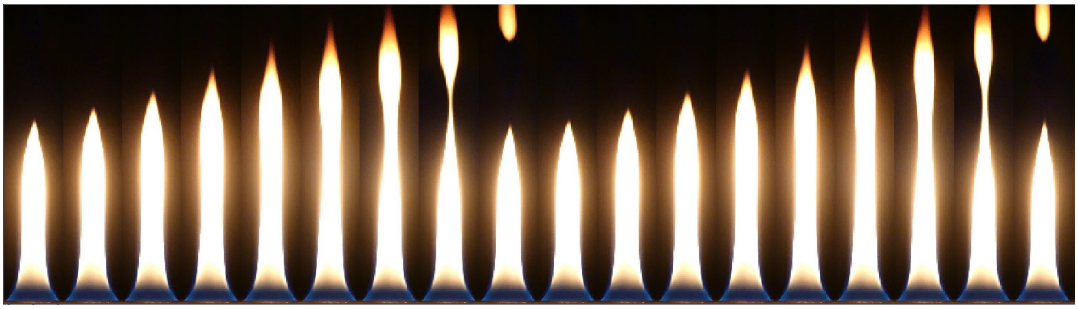


Figure 2.2: Unsteady puffing behavior of a heptane pool fire flame. Two complete puffing cycles are shown. The time interval between the images is 1/120 s. The measured puffing frequency is f = 12.8 Hz. Adapted from [28].

On top of the flame there is a non-reacting buoyant plume in the form of a smoke cloud. Although not all fires produce smoke, it is typical for pool fires to do so, especially turbulent ones of medium to large scale [7]. The appearance of smoke indicates the presence of unburnt soot that has managed to escape through the burning interface, that is, the edge of the flame. Smoke is produced when there is not enough time to burn all of the soot particles at the interface. This occurs when the so-called residence time of the soot inside of the flame is too small. Generally a small laminar pool fire like that of a candle flame will not produce smoke, see Figure 2.3. However, the occurrence of smoke can be triggered in many ways such as by moving the flame quickly from side to side, by introducing an object

(e.g. a metal piece) inside the flame, which locally quenches the flame and allows unburnt soot particles to escape, or by straightening the candle wick and therefore elongating the flame [29]. It is therefore common to observe small pool fires like candle flames with non-reacting buoyant smoke clouds above them.

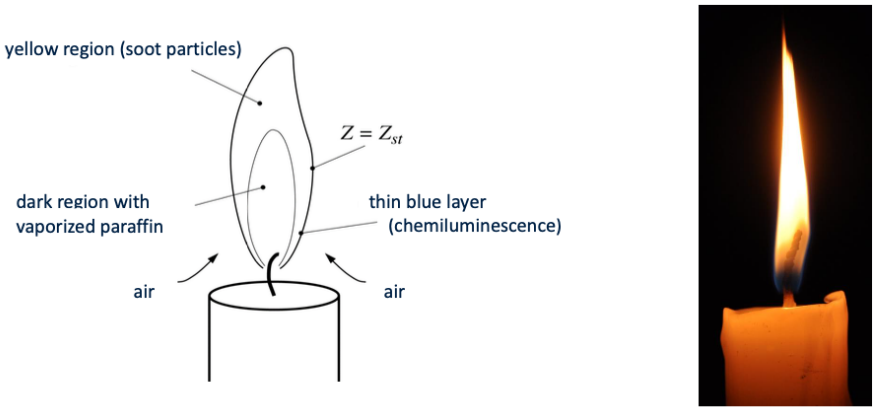


Figure 2.3: Morphology of a small laminar pool fire. A candle flame is an example of a small and generally laminar pool fire. Burning occurs at the edge of the flame, where the mixing between fuel and oxidizer is at stoichiometric conditions. Air entrainment and buoyancy sustain the burning process. Reprinted from [29].

Baroclinic vorticity and the burning process

It is worth briefly considering what exactly sustains the burning process in a pool fire flame. In order for a stable pool fire to form there must exist a feedback between the heat supply from the flame to the fuel and the evaporation and burning of the fuel itself. The fuel is heated by the flame by radiation, convection, and/or conduction. The former is the dominant heat transfer mechanism for medium and large pool fires [7]. As a result of heating, the fuel evaporates and mixes with incoming entrained air. The mixture combusts and expands, rising due to buoyancy and promoting radial air entrainment towards the center of the flame, eventually diffusing upwards in the form of a non-reacting buoyant plume. A pool fire is hence a diffusion flame where buoyancy forces are the controlling transport mechanism [7].

The way in which buoyancy generates vorticity is crucial to the development of pool fires. This so-called baroclinic vorticity introduces rotational motion into the flow field, increasing the mixing between fuel and oxidizer and generating vortices that convect upwards, breaking up into smaller vortices until they dissipate due to viscous effects. This intuitively makes sense — as the flow rises due to buoyancy and air is driven radially towards the center of the flame, rotational motion is introduced into the flow. But what is the specific physical process through which buoyancy leads to the generation of vorticity?

The relation between density variations (that is, buoyancy) and vorticity can be clearly understood by considering the Helmholtz decomposition of the velocity field into its curl ($\nabla \times V$ — the vorticity), its divergence ($\nabla \cdot V$ — the dilation), and the boundary conditions. When performing the decomposition the buoyancy term ends up in the vorticity transport equation, and is known as *baroclinic* vorticity [30]:

$$\begin{aligned} \frac{D\boldsymbol{\omega}}{Dt} &= \frac{\partial \boldsymbol{\omega}}{\partial t} + (\mathbf{u}\cdot\nabla)\boldsymbol{\omega} \\ &= (\boldsymbol{\omega}\cdot\nabla)\mathbf{u} - \boldsymbol{\omega}(\nabla\cdot\mathbf{u}) + \underbrace{\frac{1}{\rho^2}\nabla\rho\times\nabla p}_{\text{baroclinic contribution}} + \nabla\times\left(\frac{\nabla\cdot\tau}{\rho}\right) + \nabla\times\left(\frac{\mathbf{B}}{\rho}\right) \end{aligned} \tag{2.1}$$

As indicated by Equation 2.1, baroclinic vorticity is generated by the relative misalignment between the density and pressure gradients in the flow field. In the case of a pool fire, the direction of the density gradient is primarily radial — due to the high temperature inside the flame there is a strong temperature gradient in the radial direction, which in turn leads to a radial density gradient.

Additional density variations occur due to changes in composition through the spatial domain; these differences affect the density gradient, but it still acts primarily along the radial direction. On the other hand, the pressure gradient is directed in the vertical direction due to variations in ambient pressure as a function of height. Therefore, it can be seen how the pressure and density gradients in a pool fire will tend to be misaligned in the general case. It is this misalignment between the two gradients that is responsible for the generation of baroclinic vorticity, as demonstrated by Equation 2.1. The resulting vortices that form around the flame are shown schematically in Figure 2.4.

2.3. Measuring pool fires

As a canonical flow for non-premixed combustion, pool fires have been studied extensively both numerically and experimentally. The goal of this section is to provide a brief overview of the methods used to model and measure pool fires. Due to the vast amount of published literature on the topic it is important to clearly define the scope of the analysis; it is beyond the scope of the present work to give a comprehensive overview of all research related to numerical or experimental investigations of pool fires. For that purpose readers are directed to recent reviews such as that of Guo et al. [27] and Chen et al. [32], or slightly older but relevant reviews like that of Tieszen [18] and Joulain [7].

In this thesis, the key motivation for the study of pool fires is the difficulty in obtaining adequate experimental characterizations of their flow fields, specifically their velocity fields. For this reason, the focus of this section is to describe these difficulties in more detail and to discuss the role of flow field reconstruction as a way to close that gap. Numerical studies of pool fires are not considered, since their study falls outside of the scope of the present work.

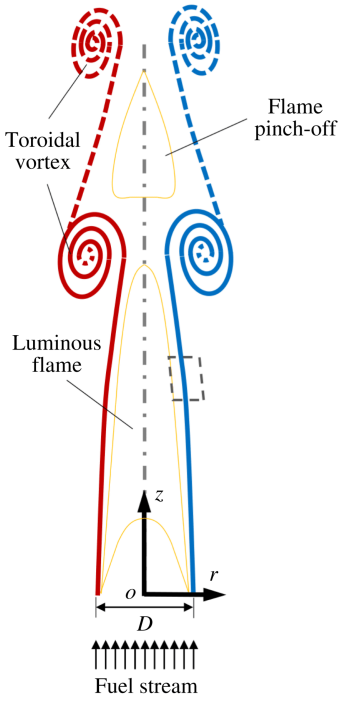


Figure 2.4: Schematic representation of the formation of buoyancy-driven vortices and their role in the puffing behavior and flame pinch-off. Adapted from [31].

Let us begin by briefly discussing the key parameters of interest when studying pool fires. From the perspective of fire safety, the main objective of an experimental campaign involving pool fires is to quantify the potential damage caused by the fire. This is the case, for example, in experimental investigations of transportation fires [21] and housing fires [22]. When studying the effects of accidental fires in a train carriage, or in the battery system of an electric car, engineers are interested in quantifying the potential effect of such accidents and the extent of the damaged caused by the fire.

For this reason, the key parameters of interest in traditional experimental campaigns of pool fires are those that are directly related to the toxicity of the fire or to its energy release to its environment in the form of heat. The toxicity can generally be estimated through the concentration and composition of the soot. The radiative heat transfer is generally measured through the heat release rate (HRR). The HRR is an integral measure of the total energy released by the fire to its surroundings in the form of heat. It is one of the most important parameters in the experimental study of pool fires due to the fact that it provides a way to compare the destructive power of different fires — two fire geometries with the same HRR can be expected to cause a similar amount of destruction to their environment [7].

In addition to estimating the HRR, it is desirable to obtain a complete characterization of the velocity fields around the fire. The reason for this is twofold. Firstly, by measuring the velocity fields around the pool fire one can obtain a better characterization and understanding of the physical mechanisms that govern such flows. Having sufficient high-quality data on the velocity profiles of a pool fire is a key element needed to improve our understanding of diffusion flames [7]. An example of a mechanism that is not yet fully understood is the air entrainment that occurs near the base of the flame, which

lies at the heart of the puffing instability that characterizes unsteady pool fires. Another example is the radiative energy blockage that is particularly important for large pool fires [7]. Both of these issues could be better understood through the determination of accurate velocity fields around the fire.

The second reason why obtaining adequate experimental characterizations of the velocity field is vital in the study of pool fires is their role in computing the HRR. The HRR can be estimated using methods that do not rely on velocity measurements, such as those based on the amount of oxygen burnt by the fire or entrainment methods based on the collection and subsequent analysis of the smoke produced by the fire [33]. Nonetheless these methods are limited, and a direct estimation of the HRR derived from temperature and velocity measurements is desirable.

Unfortunately, obtaining experimental measurements of pool fires is notoriously difficult [7, 33]. Early studies on pool fires focused on the determination of experimental correlations between the normalized flame height and the total heat release rate [18]. Later studies have attempted to characterized the basic structure of chemical pool fires (mean flame shape, convection of macroscale structures) using a wide range of experimental techniques including photographs [34], thermography [35, 36], spontaneous flame emission images [37], and infrared thermography [38].

Semi-quantitative measurements can be obtained using flame visualization. The relatively low velocities of pool fires and the presence of soot particles make the latter an interesting candidate to conduct flow visualization studies using soot as the tracer particle [18]. Additionally, fire researches have developed specific measurement techniques to perform fully quantitative measurements with some success [33]. These include the aforementioned entrainment technique to estimate the HRR and the use of a bidirectional velocity probe to perform point-wise velocimetry [18]. Attempts have been made at using non-intrusive measurement techniques such as Laser Doppler Velocimetry (LDV) to obtain large-scale velocity measurements. Particle tracking techniques have also been used to study pool fires experimentally [18, 33].

Nevertheless, despite our ability to obtain partial measurements on these flows, there is far less experimental data available in comparison with isothermal flows. And even in comparison with momentum-driven reacting flows, fire data is sparser still [18]. The problem lies in the difficulty and cost associated with instrumenting a fire due to the harsh environment they create and the presence of large velocity, temperature, and density gradients. As a consequence "the amount of data available is very small compared with isothermal or even reacting jet flows [and] the need for data to ensure growth in this field cannot be overemphasized" [18].

Of all flow states of interest, velocity is notoriously difficult to measure [7]. Even state-of-the-art experimental techniques such as Particle Image Velocimetry (PIV) or Particle Tracking Velocimetry (PTV) are still limited in the scale of experiments that can be performed [39, 40]. Even though this is an active area of research, the use state of the art PIV or PTV to perform velocimetry on the sort of experimental setups that have practical applications for the study of pool fires remains challenging. Although previous work has demonstrated the possibility of using PIV to obtain experimental characterizations of the velocity fields in flames, its application remains limited. The study of even medium-scale pool fires, which have a measurement volume of several cubic meters, is either not technically possible with current PIV/PTV techniques or prohibitively expensive and time consuming due to the complex experimental setup and expensive measurement equipment required. From a practical standpoint it is desirable to develop a robust reconstruction method that can reliably reconstruct the velocity fields from other data that is easier to obtain experimentally, such as temperature.

All in all, despite being the most common flow configuration for the study of non-premixed combustion, and despite the fact that its study dates back for almost 100 years [41], the experimental study of diffusion flames like pool fires is severely limited, specially considering setups that have practical relevance for industrial applications. In this context, flow field reconstruction presents itself as a promising tool capable of augmenting the quantity and/or quality of the available experimental datasets in pool fire flames.

2.4. The concept of the mixture fraction

Recall that diffusion flames are controlled by the mixing of fuel and oxidizer. Mixing is the slowest and therefore rate determining process in such flames [42]. As such, a quantity that measures the local state of mixing in the flame can be a very powerful tool to obtain information about the flow. The mixture fraction, denoted by ξ for the remainder of this thesis, is a passive scalar that quantifies the local state of mixing in a non-premixed flame by measuring the amount of local mass that originated from the fuel stream. The mixture fraction is constructed as a linear combination of species such that their chemical reaction terms cancel out. As such, the mixture fraction is a passive scalar that is conserved in chemical reactions.

The mixture fraction is an artificial tool built from theoretical developments. Nevertheless, one can attempt to formulate a physical interpretation for the mixture fraction. We can think of it as tracking the atomic mass fractions of certain elements, for example the mass fraction of oxygen atoms (Y_O), nitrogen atoms (Y_N) or hydrogen atoms (Y_H). These atoms do not disappear during combustion, they are simply rearranged from one chemical species into another. The mixture fraction can be thought of as a passive scalar that embeds information about this conservation process.

Assuming a two-stream non-premixed system where one portion of the boundary contains uniform properties (the fuel stream) and the rest of the domain has different uniform properties (the oxidizer stream) the mixture fraction is given by:

$$\xi = \frac{sY_F - Y_O + Y_O^0}{sY_F^0 + Y_O^0} \tag{2.2}$$

where Y_F and Y_O are the local fuel and oxidizer mass fractions, Y_F^0 is the fuel mass fraction at the fuel stream, Y_O^0 is the oxidizer mass fraction at the oxidizer stream, and s is the stoichiometric mass ratio, given by:

$$s = \left(\frac{Y_O^0}{Y_F^0}\right)_{st} \tag{2.3}$$

Using this formulation, the mixture fraction is a normalized scalar that is equal to unity at the fuel stream and zero at the oxidizer stream. The value of the mixture fraction lies in the fact that it results in a useful parametrization of a diffusion flame. In mixture fraction space all non-premixed flames have a similar structure. This so-called *flame structure* refers to the variations in temperature and composition (i.e. mass fractions of all species) of the flame. A schematic representation of the structure of a diffusion flame in physical space is shown in Figure 2.5.

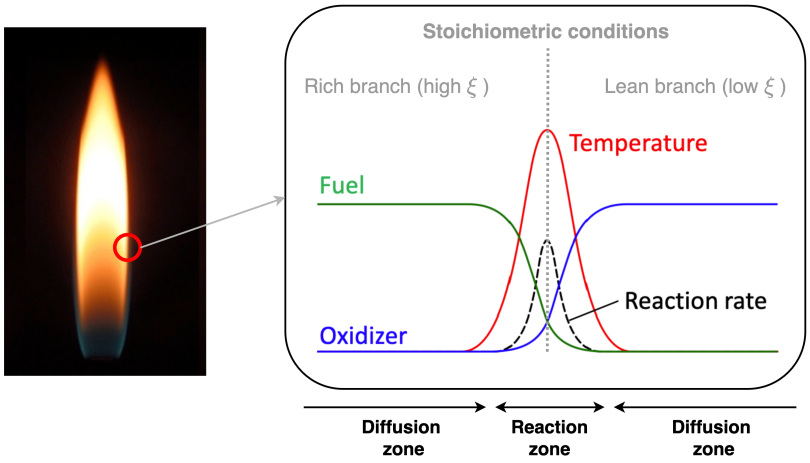


Figure 2.5: Structure of a diffusion flame in the region surrounding the flame front (schematic). The flame front separates a fuel-rich region (high ξ) from a fuel-lean region (low ξ). Adapted from [42].

To a first approximation, combustion is assumed to occur only when fuel and oxidizer mix at stoichiometric conditions. This happens at the flame front. Within the flame, the fuel mass fraction is high and the oxidizer mass fraction is low. The opposite is true outside of the flame. The maximum temperature (adiabatic flame temperature) occurs at the flame front, where the mixture fraction is at its stoichiometric value and the reaction rate is highest. The fuel and oxidizer diffuse towards the reaction zone, where they mix and combust. The flame structure shown on the right of Figure 2.5 is generally applicable to all non-premixed flames. Therefore, the mixture fraction is an interesting quantity to model within the flow field reconstruction algorithm. The transport equation for the mixture fraction is given by:

$$\frac{\partial(\rho\xi)}{\partial t} + \nabla \cdot (\rho \boldsymbol{V} \xi) = \nabla \cdot (\rho \mathcal{D} \nabla \xi) \tag{2.4}$$

where $\mathcal{D}$ is the mass diffusivity, which is assumed to be constant for all species. Note that Equation 2.4 has no source term. The removal of the chemical reaction terms that appear in the species conservation equations is also desirable from a numerical standpoint. Chemical reaction terms are generally stiff, and therefore may introduce complications when performing numerical simulations.

The mixture fraction is a useful tool in the computation of non-premixed combustion problems. It allows one to decompose such problems into two subproblems: a mixing problem and a flame structure problem. The former is a fluid dynamics problem that determines the evolution of the mixture fraction field as a function of space and time. The latter contains all the chemistry information to determine the flame structure, that is, the variation of temperature and composition in mixture fraction space. Figure 2.6 shows a schematic representation of how this decomposition can be used to construct a solver for diffusion flames. The figure implicitly assumes that the flame structure in mixture fraction space is constant in time and space.

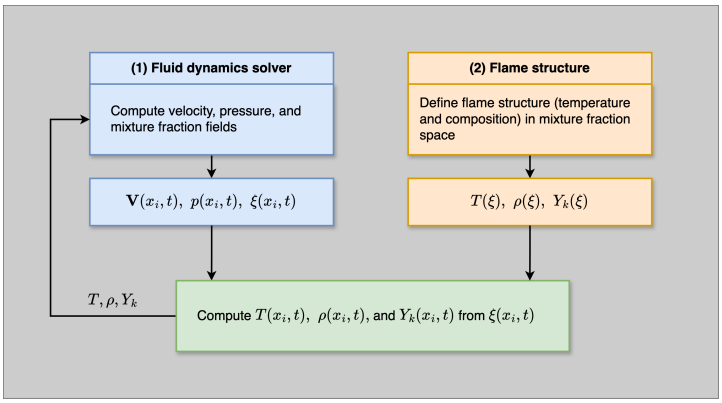


Figure 2.6: Schematic of an exemplary solver for a diffusion flame. The introduction of the mixture fraction allows separation of the fluid dynamics (left side, fluid dynamics solver) and the chemistry (right side, flame structure). The spatial coordinates are denoted by x_i, and the temporal coordinate by t. Y_k denotes the mass fraction of species k.

Several assumptions and models can be used to define the flame structure. In particular, one must make assumptions on the reversibility (or lack thereof) of the chemical reactions, in addition to their speed. A measure for the speed of the chemical reactions is the Damköhler number, defined as the ratio between the characteristic flow timescale and the chemical timescale:

$$\mathrm{Da} = \frac{\text{flow timescale}}{\text{chemical timescale}} = \frac{\tau_f}{\tau_c} \tag{2.5}$$

The present flame has a high Damköhler number and hence the assumption of infinitely fast chemistry is generally valid. Combined with the assumption of one-step irreversible chemistry, this leads to the so-called Burke-Schumann solution for the flame structure [8]. The use of the Burke-Schumann approximation to model the flame structure is described in more detail in Chapter 5 (Section 5.5). The key idea here is that the introduction of the mixture fraction allows the decomposition of a problem involving non-premixed flames into two independent subproblems, as shown in Figure 2.6. As a result, in the limit of infinitely fast chemistry, the mixture fraction acts as a universal coordinate transformation that maps all reacting scalars into the one-dimensional mixture fraction space, significantly simplifying the computation of non-premixed flames.

3

Flow field reconstruction methods

This chapter discusses flow field reconstruction methods. A brief motivation for their use is given in Section 3.1. Traditional reconstruction methods are then discussed in Section 3.2. Lastly, Section 3.3 presents novel methods for flow field reconstruction that augment the capabilities of traditional techniques by leveraging the power of deep learning.

3.1. The need for flow field reconstruction

Modern-day aerodynamicists have access to increasingly more powerful flow measurement techniques and an abundance of data from extensive numerical simulations. Being able to combine these sources of information to improve our ability to model complex fluid flows is vital in many applications. In such contexts engineers may wish to estimate a fluid flow field from limited and possibly corrupt measurements. This process is known as *flow field reconstruction*.

The available data is often experimental data, which can be noisy or incomplete. Alternatively, data from numerical simulations (which can contain errors and/or large uncertainties) may also be used. In most cases, flow field reconstruction makes use of a model that is capable of determining, given some partial information about the flow, what the rest of the flow field looks like. Being able to reconstruct coherent flow structures from noisy and incomplete data can be critically enabling for applications across many domains such as active flow control (relevant for wind turbine monitoring, for example), cardiac blood flow modeling, climate science, and geophysics, among many others [6].

3.2. Traditional methods

Many flow field reconstruction methods exist. Due to the increasing interest in machine learning a lot of new developments in the field of flow field reconstruction incorporate deep learning tools to perform data-driven reconstructions. These methods open new possibilities for flow field reconstruction — they are discussed in more detail in Section 3.3. Here, the focus is placed on traditional flow field reconstruction methods that do not use deep learning techniques. These methods have a solid track record that demonstrates their reconstruction capabilities in many different fluid problems. Additionally, they form the backbone of many of the more novel deep learning methods discussed in Section 3.3. A good understanding of the capabilities and drawbacks of traditional reconstruction methods is therefore crucial to contextualize the use of more novel techniques.

Callaham et al. [6] categorize traditional flow field reconstruction methods into 4 families, namely: stochastic estimation methods, model-based observers, library-based reconstructions, and sparse representations. The remainder of this section explores the key characteristics and limitations of these four families of methods.

Stochastic estimation methods are a statistical technique that, unlikely most of the other methods, does not rely on a modal decomposition of the flow field. Instead, conditional statistics are used to construct an estimation for some flow quantity of interest. The error in this estimation (the difference between the estimated quantity and some known measurement data) is modelled by expanding the estimation in the form of a Taylor series. The minimization of this error (generally the mean squared error) yields a functional dependence between the available measurement data and the estimated quantities. Effectively, stochastic estimation methods model the desired flow quantities as a conditional average given that the available measurements are true, and minimizes an error metric between the two datasets. Stochastic estimation has been successfully applied to study turbulent boundary layers [43, 44], open cavity flows [45, 46], asymmetric jets [47, 48], isotropic turbulence [49–51], and for the estimation of separated flows from wall pressure measurements [52], among others.

The second family of methods are *model-based observers*. These reconstruction methods use models of the flow field known as reduced-order models (ROM), or surrogate models, to construct an estimation of the flow quantities of interest. The available measurements are then combined with these surrogate models to improve the accuracy of the estimations. The motivation for the use of surrogate models is to obtain a simplified model that captures the most important characteristics of the full flow field in order to provide accurate estimations of the true flow in an efficient way. Many different types of models can be used for model-based flow field reconstruction. The choice of model plays a significant role in the accuracy of the reconstruction and the type of problems that can be tackled. A complete description of all possible surrogate models of fluid flows is well outside the scope of the present work — the interested reader is referred to reviews of modal-based analysis such as the one of Rowley and Dawson [53] for a complete overview of existing techniques. Here it suffices to briefly discuss some of the key models used for flow field reconstruction.

Most reduction models are based on the idea of modal decomposition, that is, the decomposition of a flow field into a set of basic low-dimensional modes. These modes can then be combined, generally through a linear combination of them, to reconstruct the full flow solution. One of the most common modal decomposition techniques is the so-called Proper Orthogonal Decomposition (POD). Using POD a flow field is modelled as a linear combination of spatial basis functions with time-dependent weights. For example, the POD decomposition of a velocity field $\mathbf{U}(\mathbf{x}, t)$ takes the form:

$$\mathbf{U}(\mathbf{x}, t) = \sum_i a_i(t) \cdot \phi_i(\mathbf{x}) \tag{3.1}$$

where $\phi_i(\mathbf{x})$ are the spatial basis functions (the modes), and $a_i(t)$ are the time-dependent weights corresponding to the basis functions. A linear combination of these functions and weights is taken over some index i. The POD representation is truncated by the upper limit of this index. The POD of a certain flow field is formulated as an eigenvalue problem — the spatial basis functions present the eigenmodes of the flow, while the weights are time-dependent eigenvalues that effectively describe how much energy is stored in any of those modes at a particular point in time. POD is one of the most well known modal decomposition methods. Other notable methods are dynamic mode decomposition (DMD) and the eigenvalue realization algorithm (ERA), among others [53].

A lot of model-based flow field reconstruction methods make use of such modal decomposition techniques. The resulting models can be linear (e.g. based on POD or DMD) or non-linear [6]. In some cases a model can also be obtained using model-identification techniques derived from control theory [54, 55]. The available measurements are used to update the estimates made by these models using techniques such as Kalman filtering [56]. The accuracy of model-based reconstruction methods depends on the quality of the model — for this reason, obtaining high accuracy reconstructions generally comes at the cost of high latency [6].

The third family of flow field reconstruction methods are *library-based reconstructions*. These methods are model free; instead of using a surrogate model they take advantage of large offline datasets (also known as libraries) to estimate quantities of interest. Generally, a flow field is discretized into a high-dimensional vector which is then approximated as a linear combination of modes in a library. The modes could be generic (e.g. wavelet basis functions) or flow-specific (e.g. obtained via a modal

decomposition such as POD or DMD). Other advanced algorithms exist for model construction like K-SVD (K-means Singular Value Decomposition) or GOBAL (Goal Oriented Basis Learning) [6].

A well-known library reconstruction method is Gappy POD. This method extends the standard POD to improve its tolerance against incomplete data. This is useful to perform modal decomposition based on PIV data, for example, since the resulting datasets can often have low signal to noise ratios. For this reason the data presents gaps — standard POD reconstructions treat the missing data as true data, negatively affecting the quality of the results. Gappy POD addresses this issue by iteratively guessing the values at the missing locations. Gappy POD has been shown to improve the reconstruction accuracy of standard POD in the presence of missing data. In particular Saini et al. [57] demonstrated this improvement in accuracy for the reconstruction of the velocity field of a gas turbine combustor from noisy PIV data.

Most of the reconstruction methods discussed until now use L_2 optimization. For this reason they suffer from the same limitations as least-squares parameter estimation, namely being prone to overfitting and sensitive to noise [6]. Additionally, although these approaches minimize the discrepancy in kinetic energy between the reference data and the estimated data, there is no guarantee that the estimates are globally optimal. In order to address these limitations, Callaham et al. [6] introduce *sparse representations*, the fourth and final family of traditional reconstruction methods considered in this section. The method is based on the idea of "trying to reconstruct a flow field by searching for a sparse representation in a library of example flow fields rather than minimum-energy solution in a modal library" [6].

A schematic representation of how the sparse representation method can be used for flow field reconstruction is shown in Figure 3.1. The method assumes that a some state vector $\mathbf{x}$ can be accurately reconstructed as a linear combination of library elements Ψ_j belonging to a library of example flow fields, denoted by Ψ. The existing training library Ψ is used to construct a measured library $C\Psi$ during offline training. Then, a sparse representation of some test data $\mathbf{y}$ is constructed by applying the same measurement operator C and noting that the measurement data can be corrupted by some non-zero noise η, that is, $\mathbf{y} = C\mathbf{x} + \eta$. This sparse representation is used to obtain the sparse coefficients $\hat{\mathbf{s}}$. The reconstructed flow field $\hat{\mathbf{x}}$, which is only an estimation of the true test data $\mathbf{x}$, can then be constructed by applying the sparse coefficients to the training library, that is, $\hat{\mathbf{x}} = \Psi\hat{\mathbf{s}}$.

Callaham et al. [6] demonstrate that sparse representations are capable of outperforming other model-based and library-based reconstruction methods for a variety of flow field reconstruction tasks of varying complexity. Their work includes the study of a mixing layer, the periodic vortex shedding over a 2D cylinder, and the estimation of the global sea surface temperature field and the vorticity field over the Gulf of Mexico using real-world data.

All in all, it is clear that there are many methods to perform flow field reconstruction, ranging from basic modal decompositions like POD or DMD, to stochastic estimation methods, to more complex library-based reconstructions and even sparse representations. All methods presented up until now have been successfully applied to a variety of fluid problems. Nonetheless, they all suffer from some key limitations. The methods considered so far require large training libraries to work effectively. This prohibits their application in cases where obtaining experimental data is difficult. Additionally, their performance is very strongly dependent on the location of measurements within the spatio-temporal domain. And lastly, all of the methods require at least *some* training data on the flow properties to be reconstructed. It is not possible to apply these methods to perform velocity field reconstruction, for example, in cases where no velocity data is available at all. These disadvantages limit the applicability of traditional flow field reconstruction methods for the study of reacting flows like pool fires, where one may wish to reconstruct flow quantities for which no experimental data is available at all. Recently, novel reconstruction methods that make use of machine learning techniques have been developed to extend the traditional method presented up until now. These novel reconstruction methods are briefly discussed in Section 3.3.

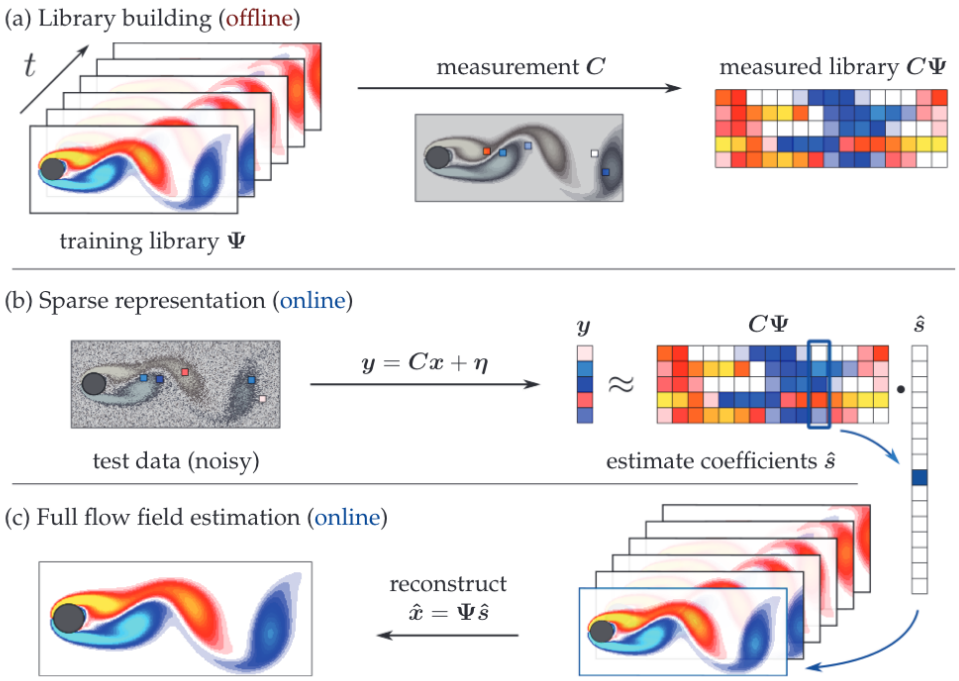


Figure 3.1: Flow field reconstruction using a sparse representation. Reprinted from [6].

3.3. The role of machine learning

Due to the increasing interest in the field of machine learning, traditional flow field reconstruction methods are being extended by leveraging the power of deep learning for data-driven applications. The state of the art in flow field reconstruction heavily relies on machine learning tools. A quick search on Google Scholar demonstrates this very clearly, as shown in Figure 3.2 (next page). Over the last 5 years the percentage of flow field reconstruction research that makes use of deep learning techniques has grown from less than 30% between 2018 and 2019 to almost 70% since 2022 until the present.

Deep learning methods are being used to extend and complement existing methods like modal decompositions and sparse representations. For example, Yu and Hesthaven [58] used an artificial neural network to estimate the POD coefficients needed for the development of a multidimensional aerodynamic database. Deep learning techniques have also been used to address well-known limitations of linear dimensionality reduction techniques like POD or DMD, which struggle to efficiently capture symmetries in the data (e.g. rotating structures) [6]. Nonlinear dimensionality reduction techniques are now being developed with the help of deep learning techniques [59–61].

More generally, deep learning techniques are widely used for flow field reconstruction. Fukami et al. demonstrated the use of deep learning methods to perform super-resolution on grossly under resolved turbulent flow fields [62]. The authors compare two deep learning techniques, namely a convolutional neural network (CNN) and a hybrid downsampled skip-connection/multi-scale (DSC/MS) model. They report remarkable accuracy in the reconstruction of a 2D cylinder wake. Other examples of the use of CNNs for flow field reconstruction can be found in the work of Hao et al. [63]. Erichson et al. [64] used a shallow neural network to perform flow field reconstruction from limited sensors, simulating the conditions of typical experimental campaigns. They tested their method on three examples from fluid mechanics and oceanography. Their results reproduced the reconstruction accuracy of traditional methods using much fewer sensors. Peng et al. [65] used a hybrid deep learning approach to reconstruct the unsteady periodic flow field behind a cylinder and the transonic flow around a NACA 0012 airfoil. The authors report a significant reduction in reconstruction error (70-85%) compared to traditional methods like POD and shallow decoders (SD).

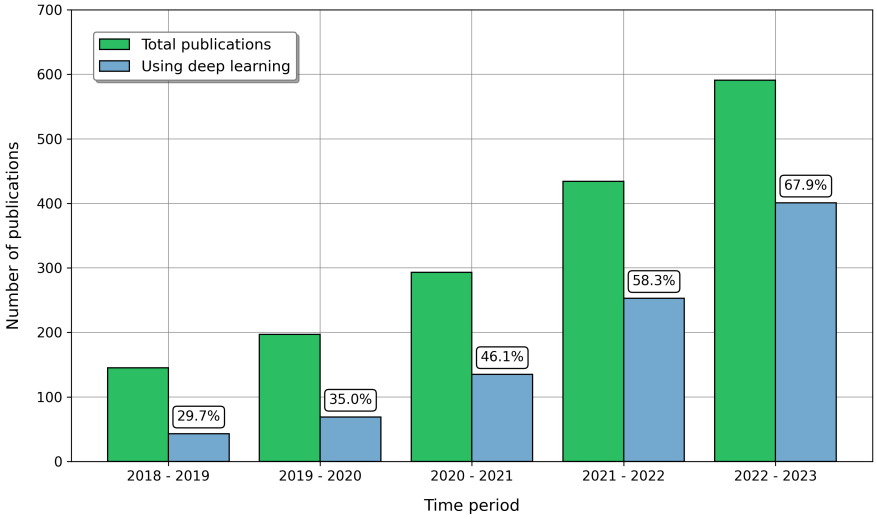


Figure 3.2: Growth of deep learning techniques for flow field reconstruction. The data has been gathered by noting the number of Google Scholar entries related to flow field reconstruction and filtering them to identify those that make use of deep learning techniques. The latter are the subset of publications that contain any of the keywords "deep learning", "neural network", or "machine learning" in their title.

Some work has also been done on the use of deep learning techniques to perform flow field reconstruction for reacting flows. Barwey et al. [66, 67] used CNNs to map Hydroxyl Planar Laser Induced Fluorescence (OH-PLIF) images to PIV in a swirl stabilized combustor. The authors compare two different types of CNN models and apply them to both attached and detached flames. Their results demonstrate the powerful ability of such models to perform flow field reconstruction, specially for attached flames. Additionally, Deng et al. [68] used deep learning to reconstruct the flow field in a supersonic combustor from Schlieren images. Finally, Li et al. [69] demonstrated the use of a dual graph neural network (DGNN) to perform flow field reconstruction and performance prediction of turbomachinery.

All in all, the existing body of literature contains a multitude of examples that report promising results. It is clear that the use of deep learning techniques can enhance existing flow field reconstruction methods or even create completely novel algorithms that are capable of accurately reconstructing flow fields for both traditional fluid dynamics applications and reacting flows. Nevertheless, these methods still exhibit one of the key limitations that is also prevalent in traditional reconstruction methods, as discussed in Section 3.2. All methods discussed so far are unable to reconstruct flow quantities for which no data is available. The methods need at least *some* data on the quantity of interest in order to perform the reconstruction. What happens in cases where one seeks to reconstruct a physical quantity for which no data is available, as in the case of velocity reconstruction in pool fires? The use of these methods is not possible in such cases.

Fortunately, novel tools in the field of scientific machine learning may provide a solution to this problem. The key is the use of physical priors in the training process of the neural networks. The embedding of a physics-based constraint into the neural network architecture may allow for the reconstruction of physical quantities for which no data is available. So-called physics-informed neural networks (PINNs) have recently emerged as a promising alternative to other flow field reconstruction methods. They have already been applied with some success to a variety of flow field reconstruction problems, but the limits of their applicability still remain unknown. These novel reconstruction algorithms are the focus of Chapter 4.

4

Physics-informed neural networks

This chapter discusses physics-informed neural networks (PINNs). Background on PINNs and motivation for their use is given in Section 4.1. A detailed description of PINNs and their key attributes and features is given in Section 4.2. An overview of relevant applications of PINNs is then given in Section 4.3, focusing on examples that demonstrate their use for data assimilation and flow field reconstruction. Lastly, a brief overview of some well-known pathologies of PINNs and ways to mitigate them is given in Section 4.4.

4.1. Background and motivation

Nowadays scientists and engineers across many disciplines of the natural sciences are capable of simulating a wide range of physical systems through the numerical discretization of partial differential equations (PDEs). Over the years many different methods to perform such discretizations have been developed. This includes finite difference methods, fine element methods based on weak formulations (e.g. Galerkin methods), finite volume methods, and spectral methods, among many others.

These methods have been applied across many disciplines of engineering with great success. Nevertheless, they also have limitations. Firstly, traditional numerical methods are generally unable to easily incorporate known data about the solution into the numerical scheme [1]. So-called *data assimilation* methods exist, but often they are model or problem specific, limiting their applicability. This issue is particularly relevant today; we live in a data-driven world where multi-fidelity data, that is, data obtained from different sources with different levels of spatio-temporal resolution, is readily available in many engineering applications. The obtained data can be noisy, gappy, or incomplete, which further increases the complexity of the data assimilation problem. Data driven methods have been developed to try to overcome these issues. Although they have been successful in many disciplines, they have a tendency to overfit the observations, hindering their ability to generalize beyond their training data [1]. Therefore, there is a need for a universal approach that is capable of seamlessly incorporating noisy real-life data to find solutions to problems governed by PDEs across the physical sciences and beyond.

In addition, while traditional numerical methods are well equipped to solve forward problems, they are generally unable to solve *inverse problems* [1]. Forward problems are well-defined problems in the traditional sense — the PDE or system of PDEs that governs the dynamics of the system are known, as are the boundary and initial conditions for the problem. In this case the system of PDEs can be discretized and propagated numerically in time from an initial condition to find the desired solution. Traditional numerical methods perform very well dealing with forward problems [3]. Inverse problems, on the other hand, pose a challenge for these methods. Inverse problems are those where a set of known observations (e.g. measured data) must be used to infer information about the causal factors that produced them (e.g. the system of PDEs and/or boundary and initial conditions for a specific problem). Data assimilation is therefore an inverse problem.

In this context, machine learning has recently emerged as a promising alternative to tackle the aforementioned issues by augmenting existing numerical methods or creating novel methods altogether [1, 3, 5]. One area of research that has received a lot of attention is the development of *physics-informed deep learning methods*. These methods are characterized by the integration of knowledge about the physical laws that govern a specific process into the learning process of a machine learning method, typically encoded as empirical or mathematical descriptions of the system under consideration. By doing so the user is effectively including physical priors which, alongside experimental data, can improve the ability of the algorithms to capture the complex (and often nonlinear) relationships present in the data. The addition of physical priors prevents the method from having a tendency to overfit the data and ensures that the results are consistent with the underlying physical laws. Additionally, it removes the need for large amounts of training data — physics-informed deep learning methods are capable of working with smaller datasets by exploiting the information embedded in the physical formulation of the problem [3].

The volume of literature on the use of physics-informed deep learning techniques for scientific computing is incredibly vast and growing at a fast pace. It is beyond the scope of the present work to give a comprehensive overview of the field — for that purpose readers are referred to recent reviews on the topic, see [1, 3, 5]. In the present work the focus is placed on a specific type of physics-informed deep learning algorithm known as a physics-informed neural network (PINN). A detailed description of PINNs and their applications is given in the following sections.

4.2. What are PINNs?

Physics-informed neural networks (PINNs) are a type of machine learning tool that incorporates information from governing physical laws in order to improve its ability to model a complex dynamical system. The physical laws are embedded as a soft constraint in the loss function of the neural network. In addition to the physical priors, PINNs are capable of seamlessly incorporating information from known data, as well as constraints from boundary conditions, initial conditions, or any other problem-specific knowledge. PINNs, first introduced by Raissi et al. in 2017 [70, 71], are characterized by a composite loss function which can be easily adapted to include additional constraints into the learning process of the neural network. In the most general case, the loss function of a standard PINN takes on the following form:

$$\mathcal{L}_{\text{total}} = \sum \omega_i \, \mathcal{L}_i = \omega_{\text{phys}} \, \mathcal{L}_{\text{phys}} + \omega_{\text{data}} \, \mathcal{L}_{\text{data}} + \omega_{\text{BC}} \, \mathcal{L}_{\text{BC}} + \omega_{\text{IC}} \, \mathcal{L}_{\text{IC}} + \omega_{\text{other}} \, \mathcal{L}_{\text{other}} \tag{4.1}$$

where the loss terms $\mathcal{L}_i$ represent the different components of the loss function, and the weights ω_i are used to modify the relative importance of each term to the overall loss of the network.

In the most general case, the loss function of a PINN will have a component derived from the residual of the PDEs that govern the system, which embeds the information about the physical priors ($\mathcal{L}_{\text{phys}}$), a component that fits the observations from some known data about the solution ($\mathcal{L}_{\text{data}}$), components to enforce boundary or initial conditions ($\mathcal{L}_{\text{BC}}$ and $\mathcal{L}_{\text{IC}}$), and additional terms that can encode other problem specific constraints ($\mathcal{L}_{\text{other}}$). Depending on the problem some of those terms may or may not be used — this modularity and flexiblity of the loss function is one of the key advantages of PINNs over traditional numerical methods.

For example, one could use a PINN to solve a traditionally well-posed forward problem by only including the loss term related to the PDE residuals ($\mathcal{L}_{\text{phys}}$, the physical prior) and the loss terms associated with boundary and initial condition ($\mathcal{L}_{\text{BC}}$ and $\mathcal{L}_{\text{IC}}$). Similarly, one could use a PINN to solve a traditionally ill-posed or overdetermined inverse problem by providing the loss function with a term to enforce some known data ($\mathcal{L}_{\text{data}}$), a term to include some information about physical priors ($\mathcal{L}_{\text{phys}}$, even if this is an incomplete description of the underlying system of PDEs), and some other problem-specific constraints ($\mathcal{L}_{\text{other}}$). The PINN architecture is agnostic to the type of problem under consideration — this is the source of its flexibility and adaptability to many different types of problems and represents a significant advantage compared to traditional numerical methods [72].

Let us now briefly discuss the typical architecture of a PINN. Following the example of Cai et al. [3] we consider the general parametrized PDE problem given by:

$$\begin{aligned} &\boldsymbol{f}\left(\boldsymbol{x},\, \hat{u},\, t,\, \partial_x \hat{u},\, \partial_t \hat{u}, \ldots,\, \boldsymbol{\lambda}\right) = 0, && \boldsymbol{x} \in \Omega,\ t \in [t_0, T], \\ &\hat{u}\left(\boldsymbol{x}, t_0\right) = g\left(\boldsymbol{x}\right), && \boldsymbol{x} \in \Omega, \\ &\hat{u}\left(\boldsymbol{x}, t\right) = h\left(t\right), && \boldsymbol{x} \in \delta\Omega,\ t \in [t_0, T] \end{aligned} \tag{4.2}$$

where $\boldsymbol{x} \in \mathbb{R}^d$ is the spatial coordinate, $\boldsymbol{f}$ is the residual of the nonlinear PDE governing the system, $\hat{u}$ is the solution to the PDE, $g\left(\boldsymbol{x}\right)$ is the initial condition, $h\left(t\right)$ is the boundary condition, and $\boldsymbol{\lambda}$ are some unknown parameters of the PDE. Ω and $\delta\Omega$ represent the spatial domain and boundaries, respectively. t_0 is the initial time and T the final time.

A PINN is then constructed as a fully-connected feedforward neural network with multiple hidden layers capable of finding an approximation $\tilde{u}$ to the real solution $\hat{u}$. As an example, consider the case where Equation 4.2 describes a 2D multiphysics problem where the inputs to the network are the two spatial coordinates x and y and the time t, while the outputs are the 2D velocity components (u, v), the pressure p, and a scalar field ϕ (which could represent the temperature, for example). A schematic representation of a typical PINN architecture for such a problem is shown in Figure 4.1.

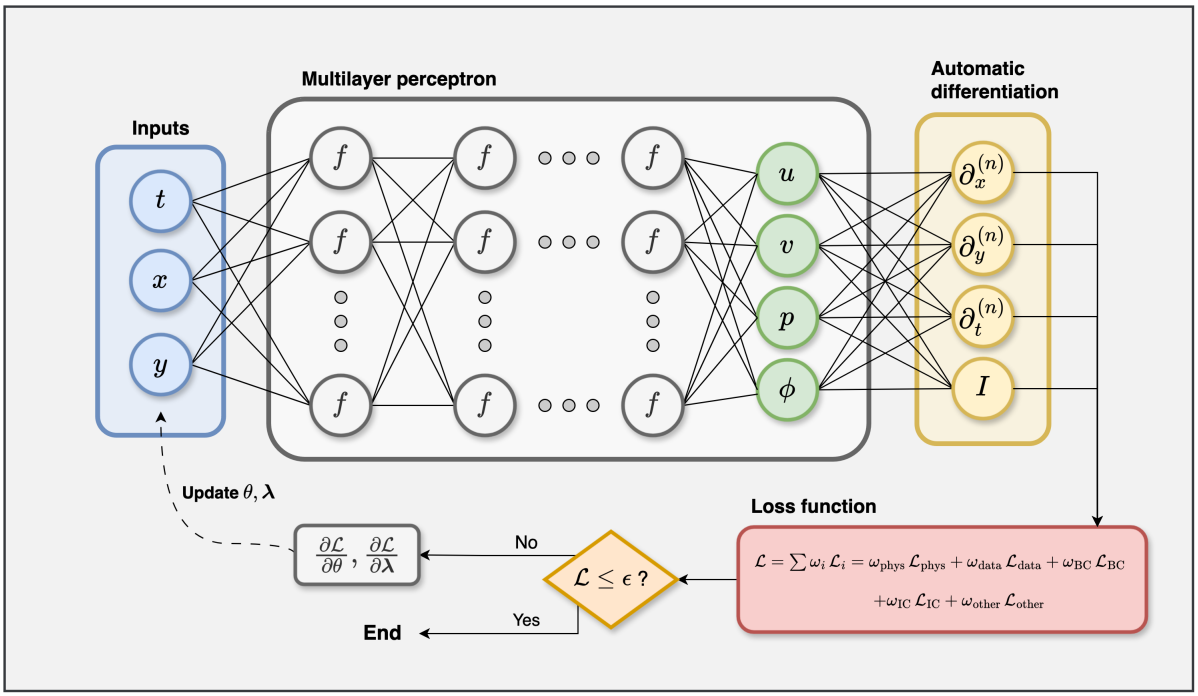


Figure 4.1: Typical network architecture of a standard PINN. The inputs are the spatiotemporal coordinates $[x, y, t]$. These are fed through the multilayer perceptron to produce the outputs $[u, v, p, \phi]$ apply a non-linear activation function f to the outpuf of each neuron. Automatic differentiation is used to compute the required derivatives of the approximate solution $\hat{u}$ with respect to the inputs. These are used to compute the residuals that make up the physics-informed loss $\mathcal{L}_{\text{phys}}$. The components of the composite loss function $\mathcal{L}$ are computed and added together with their respective weights ω_i. The parameters of the network θ (the weights and the biases) and any unknown parameters of the system of PDEs $\boldsymbol{\lambda}$ can be learned simultaneously by minimizing $\mathcal{L}$ through an optimization procedure, which is generally gradient-based.

If we denote the hidden variables of the k^{th} hidden layer by $\mathbf{z}^k$ then the inputs $[\boldsymbol{x}, t] = [x, y, t]$ are propagated through the feed-forward neural network by performing the following computations:

$$\begin{aligned} &\mathbf{z}^0 = [\boldsymbol{x}, t] \\ &\mathbf{z}^k = \sigma\left(\mathbf{W}^k \mathbf{z}^{k-1} + \mathbf{b}^k\right), && 1 \leq k \leq P-1 \\ &\mathbf{z}^k = \mathbf{W}^k \mathbf{z}^{k-1} + \mathbf{b}^k, && k = P \end{aligned} \tag{4.3}$$

where σ is an activation function, $\mathbf{W}^k$ and $\mathbf{b}^k$ are the matrix of weights and the biases for the k^{th} layer, and the networks has $P+1$ layers including the output layer. The values of the output layer are used to compute an approximation $\tilde{u}$ to the real solution $\hat{u}$. The combination of all network parameters (weights and biases) is denoted by θ.

Recall now the general form of the loss function, given by Equation 4.1. In order to compute the residuals of the PDE the network must be able to compute the derivatives of the approximate solution $\tilde{u}$ with respect to the inputs. This is done using a procedure known as *automatic differentiation* [3]. Using this procedure, the network is capable of computing all of the partial derivatives needed to evaluate the physics-informed term of the loss function. Automatic differentiation calculates the derivatives of the outputs with respect to the network inputs directly in the computational graph [3]. By calculating the derivatives with an explicit expression this method avoids the introduction of truncation or discretization errors [3].

The physics-informed term of the loss function is computed at specific points in the spatio-temporal domain. These points are known as *collocation points*, sometimes also referred to as residual points. The choice of collocation points can have a significant impact on the performance of the PINN, as discussed in Section 4.4. The data-driven term of the loss function ($\mathcal{L}_{data}$) is computed by calculating the difference between the approximate solution $\tilde{u}$ at the current iteration with some known data at specific points known as *measurement points*. In the case of synthetically generated data these measurement points may be chosen to be equal to the collocation points for the sake of simplicity, although this need not be the case. In the case of using experimental data the spatio-temporal resolution and location of the measurement points is determined by the experimental apparatus. The remaining terms of the loss function ($\mathcal{L}_{BC}, \mathcal{L}_{IC}, \mathcal{L}_{others}$) are also calculated and added to compute the total loss using the different weights for each component $\mathcal{L} = \sum \omega_i \mathcal{L}_i$.

If the loss is larger than some acceptable threshold ϵ, the network parameters θ and any unknown parameters of the PDE system $\boldsymbol{\lambda}$ can be updated simultaneously by minimizing the loss. This optimization procedure is generally done with a stochastic gradient-descent method [1, 3]. The choice of optimizer will affect the performance of the network. Common choices are the Adam optimizer and the L-BFGS optimizer [1].

Lastly, it should be noted that there is a key difference between PINNs and conventional "black-box" neural networks. The latter are normally trained on large datasets and then applied to a specific task through a testing dataset. In the case of PINNs the networks are not trained beforehand and then applied to a specific problem. Instead, they are trained on the problem itself by incorporating the known data and the physics priors into the loss function, and then trying to approximate the exact solution over the field. This difference removes the need for large volumes of training data. Nevertheless the use of PINNs still requires some data if the user decides to include a data-driven component in the loss function, in addition to some testing data that acts as ground truth to measure the final estimation error of the neural network.

4.3. Applications of PINNs

Due to their flexibility, PINNs have been applied to solve a wide range of problems across different domains of the physical sciences. The volume of literature on the applications of PINNs and their variants is vast and extends across many different domains. A complete description of all applications of PINNs is outside the scope of the present work. Instead, for the remainder of this section the focus is placed on the most relevant use case: using PINNs to capture complex model dynamics in the presence of imperfect data. Section 4.3.1 presents examples where PINNs are used for data assimilation tasks like flow field reconstruction. Then, Section 4.3.2 highlights examples where PINNs and other related physics-informed machine learning techniques are used to analyze reacting flows.

4.3.1. PINNs for data assimilation

One of the key advantages of PINNs over traditional numerical methods is their ability to perform data assimilation tasks in a quick, computationally inexpensive, and flexible way [1, 3, 5]. The existing body of literature contains a multitude of examples where PINNs are used to perform data assimilation tasks in different domains. The general consensus is that PINNs excel at incorporating noisy and scattered data while respecting the underlying physical laws, resulting in surrogate models for complex physical systems that outperform traditional numerical methods at solving inverse problems. This section presents and discusses some relevant examples found in the literature.

Let us begin by considering the applications of PINNs to perform data assimilation tasks in the field of fluid dynamics, also known as flow field reconstruction (see Chapter 3). Mao et al. [73] applied PINNs to solve multiple inverse problems for high-speed flows. Motivated by the Schlieren photography experimental technique commonly used for high-speed aerodynamics the authors combined density gradient and pressure data with conservation laws to construct a surrogate model for a high-speed flow configurations. They applied this model to problems with smooth solutions and to discontinuous Riemann problems (Sod and Lax problems), obtaining satisfactory results in both cases. Comparing PINNs to traditional numerical methods the authors claim that PINNs are "superior for inverse problems that cannot even be solved with standard techniques" [73].

Another example of the ability of PINNs to perform data assimilation in fluid dynamics is the work of Bai et al. [74]. The authors considered two different types of PINNs and applied them to a variety of problems. The numerical tests include a 2D lid-driven cavity flow, a 1D propagating wave, laminar flow passing through a porous media, and turbulent flows passing through a C-shaped channel. Their results show the ability of PINNs to successfully perform data assimilation in a variety of complex fluid problems. The authors conclude their paper claiming that "the learned results are not only comparably accurate as the traditional CFD techniques, but also indicate considerable capability to reconstruct missing flow dynamics, for both laminar flows and turbulent flows" [74].

The work of Raissi et al. [75, 76] has received a lot of attention in the community. Their work focuses on the use of PINNs to discover "hidden" quantities in a flow field for which no data is available. They perform data reconstruction on fluid problems using PINNs combined with scattered measurements of a passive scalar field obtained via flow visualization. The authors apply this technique to several flow problems, including the Kármán vortex street over a cylinder (in both two and three dimensions) and the study of two dimensional channel flow over an obstacle. Additionally, they apply the technique to a biomedical engineering problem, namely the study of 3D physiologic blood flow in a realistic intracranial aneurysm (ICA). The data for this latter problem were concentration fields generated using patient-specific boundary conditions (the aneurysm geometry). The authors observed very good agreement between the reference and regressed datasets for all problems considered. A graphical summary of the methodology and the results obtained for the ICA example are shown in Figure 4.2.

Wang et al. [77] used a PINN to perform dense velocity reconstruction from particle image velocimetry (PIV) / particle tracking velocimetry (PTV) data. They investigated the performance of the network using 2D Taylor decaying vortices, turbulent channel flow, and the reconstruction of the 3D velocity field in the wake of a hemisphere. The results show that the PINN is capable of reconstructing the velocity fields even in the presence of noisy data. The authors report that the use of PTV data is favorable for PINNs since they appear to struggle dealing with the spatially correlated data introduced by the cross-correlation algorithm of PIV data. The authors also obtained a reconstruction of the pressure field around the hemisphere. They report a pressure distribution that is qualitatively accurate, claiming that the technique can be used to estimate the force around the object. Nonetheless they obtain errors that are one order of magnitude larger than those of the velocity predictions (in the order of 10% for the pressure versus 1% for the velocity). Overall their results demonstrate the potential of PINNs to aid in data assimilation tasks in experimental fluid mechanics. Nonetheless, the authors highlight the slow convergence of the loss function and the limited accuracy as key areas for future research. This is a common theme in many applications of PINNs; the reasons for these difficulties in the training process are discussed in more detail in Section 4.4.

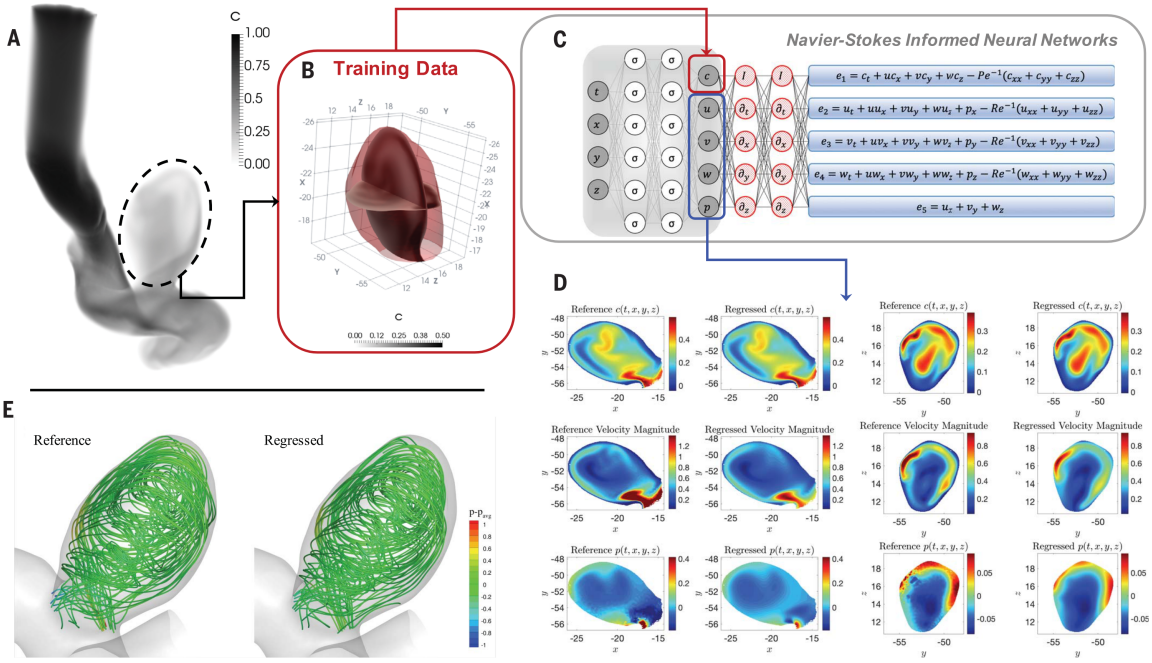


Figure 4.2: Inferring quantitative 3D hemodynamics using PINNs. Panel A shows the patient-specific aneurysm geometry used to generate the training data, shown in panel B. The architecture of the used PINN is shown in panel C. Comparisons between the reference and regressed data are shown in panel D (2D countours of concentration, velocity, and pressure) and panel E (3D flow streamlines). Reprinted from [76].

The work of Xu et al. [78] is another example of successfully applying PINNs to perform data reconstruction for fluid flows. Their study provides encouraging insight for the potential of PINNs as a data assimilation method in experimental fluid mechanics. They tested the data reconstruction capabilities of PINNs (for the pressure p and flow function ψ) in two cases: one where sparse velocity data was available, and another one when incomplete velocity data was available (a certain region of the spatial domain is blocked and therefore no data is gathered in that region). The authors implement an adaptive algorithm for the learning rate to improve the training performance of the network. Using this method they obtain reconstruction errors in the order of 10^{-2} to 10^{-3}. Their results are satisfactory, but their research is limited to incompressible flows with low Reynolds numbers.

Cai et al. [79] used a PINN to infer the 3D-velocity and pressure fields over an espresso cup using only temperature fields obtained via tomographic background-oriented Schlieren photography (Tomo-BOS). The authors first quantified the accuracy of the method using a 2D synthetically generated dataset for buoyancy-driven flow, and then applied it to the Tomo-BOS data. An independent PIV experiment was also performed to qualitatively validate the inference of the unsteady velocity fields at the center plane. The authors found that the reconstructed data exhibited the same general features and with the same magnitude that the data obtained from the PIV experiments.

Other examples of the use of PINNs to perform data assimilation in the field of fluid dynamics include the reconstruction of hydrofoil cavitation flow done by Ouyang et al. [80], the indoor airflow field reconstruction reported by Wei and Ooka [81], the reconstruction of the 3D flow field around a building model in a wind tunnel test of Rui et al. [82], and the use of PINNs to perform super-resolution (high-resolution data augmentation) of fluid flows done by Eivazi and Vinuesa [83].

The work of Kissas et al. [84] provides yet another example of the potential of PINNs for data assimilation tasks, in this case for a biomedical application. The authors combined non-invasive measurement technique known as 4D flow MRI with a PINN to create a surrogate model of the cardiovascular flow in human systemic arteries. This surrogate model was then fed with limited clinical data to obtain physically consistent measurements of quantities that cannot be easily obtained in a non-invasive manner.

The used dataset was made up of noisy, scattered measurements of blood velocity and wall displacement obtained by non-invasive 4D flow MRI. The obtained outputs are physically consistent predictions for velocity, pressure, and wall displacement pulse wave propagation, all without the need to employ conventional simulators, without the need to stipulate boundary and initial conditions, and without the need to generate a complex mesh for the geometry.

All examples presented above show that PINNs are capable of successfully performing data assimilation and data reconstruction tasks across different domains, even in cases where the used data is noisy and scattered. The consensus is clear — the existing body of literature strongly suggests that the use of PINNs to perform data assimilation is justified and can produce results that are in very good agreement with reference data.

4.3.2. PINNs for reacting flows

Reacting flows like pool fires are characterized by strong temperature and density gradients and exhibit a high degree of complexity due to the coupling between the momentum and scalar fields and the chemical reactions present in the flow, as described in Chapter 2. If PINNs are to be applied to perform data reconstruction on such flows, it is worth considering what previous work exists, if any, in the study of reacting flows using this technique.

Unfortunately, very limited work has been published on the matter. PINNs are a relatively new technique in the field of scientific computing and the scope of their application remains limited. As shown by the examples of the previous section, most of the applications of PINNs to perform data assimilation tasks consider flows that exhibit small density variations and can be described by a Boussinesq approximation. This approximation is not applicable for reacting flows like pool fires which contain large density variations and are buoyancy-driven.

Nevertheless, some work does exist on the use of PINNs to study reacting flows. Additionally, by slightly broadening the scope of the research one can also find examples of other related deep learning techniques (some purely data driven and some physics-informed) for the study of reacting flows. Some key examples are discussed hereinafter.

Tathawadekar et al. [85] explored the use of purely data-driven deep learning approach to perform data reconstruction in reacting flows. They consider the case of a Bunsen flame under acoustic excitation by a single-tone velocity perturbation. A neural network architecture composed of a series of CNNs is used to perform multilevel filtering of the input fields (in this case numerically generated velocity fields) and recombine them into an output field. The target of the reconstruction is the HRR field of the flame. The authors conclude that their network architecture is "able to reconstruct the HRR field from the velocity field, not only in terms of morphology and spatial features, but also in an integral sense" [85].

The work of Yadav et al. [86] demonstrates the use of a physics-informed neural network architecture to infer the linear and nonlinear response of a premixed laminar flame to incoming velocity perturbations. They make use of a physics-informed long short-term memory network (PI-LSTM) to construct a surrogate model that is capable of reconstructing the temporal variations in HRR. Similarly, Tathawadekar et al. [87] report the use of a physics-informed neural network to construct a surrogate model of laminar flames. They demonstrate the validity of the proposed approach on a planar and a Bunsen-type flame at various operating conditions. Their results demonstrate that the method is capable of accurately predicting the evolution of the flame interface and flame shape.

Chi et al. [88] used physics-informed machine learning for data-driven discovery of optimal HRR markers for NH_3/H_2/air flames. The found markers are capable of reconstructing both spatial and temporal evolution of HRR from limited measurements on species concentrations. Additionally, PINNs have also been used to successfully create surrogate models for chemical kinetics that are relevant to the study of reacting flows. Examples of this are the work of Nakazawa et al [89] and Ji et al. [90]. PINNs have also been proven to be effective at performing flow field reconstruction for combustion problems. Examples are the work of Wang et al. [91], who performed flow field reconstruction in a rotating

detonation combustor, and the work of Darlik and Peters [92], who applied a PINN to reconstruct the particle fields in a particle-fluid problem relevant for biomass combustion chambers.

The publications highlighted so far show the potential of PINNs to analyze reacting flows of various types. But the most relevant work for the proposed research is that of Sitte and Doan [26] and that of de Boer [93]. Both investigate the use of PINNs to perform data reconstruction for small-scale two-dimensional laminar pool fires. The goal of the authors is to obtain a reconstruction of unmeasured quantities from measured ones using a PINN. In this case the measured quantities are taken to be the temperature, density, and pressure fields, while the unmeasured quantities are the two velocity components. Numerical simulations are performed to obtain the required data. The DNS data on the measured quantities is used to train the neural network through the computation of the data-driven term of its loss function. The network is never introduced to any data on the unmeasured quantities — the latter is only used as "ground truth" to evaluate the performance of the network after training.

The numerically generated data is validated by comparing it to existing numerical and experimental data of Moreno-Boza et al. [28]. The DNS data is shown to replicate experimental data in terms of the flame length, critical point, and puffing frequency. Both a steady and an unsteady puffing case were analyzed. In the steady case the results show a very good agreement between the reference and reconstructed data for the axial velocity component (u) not only qualitatively but also quantitatively. The results are inferior for the radial velocity component (v) but still satisfactory. For the unsteady case the results show adequate qualitative agreement between the reference and reconstructed data for both velocity components.

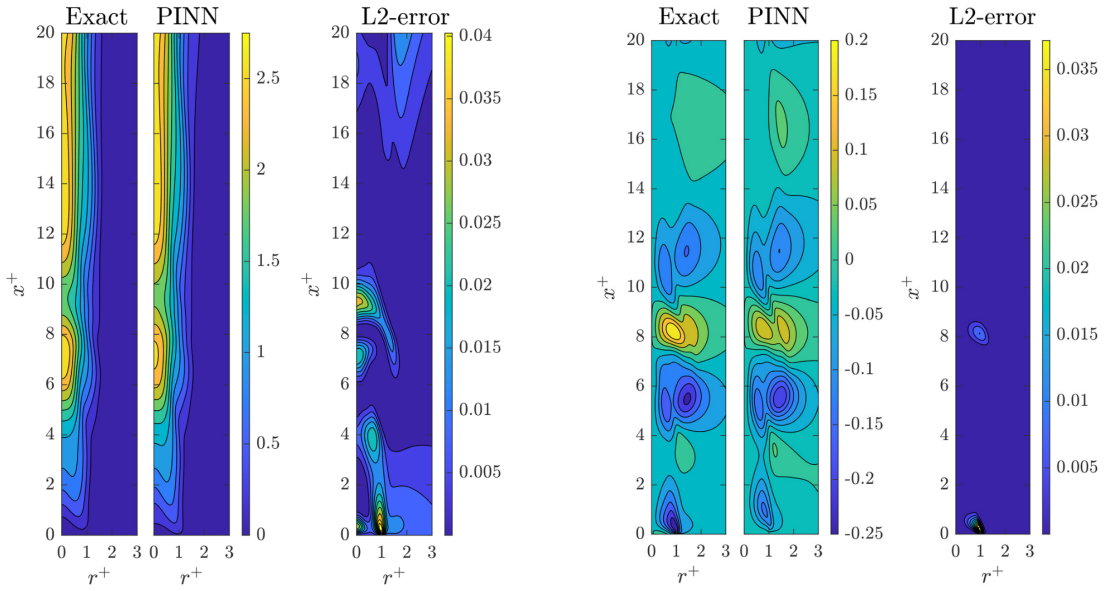
In general it can be concluded that PINNs are capable of successfully reconstructing the velocity profiles from the measured quantities, although further work is required to improve the reconstruction accuracy. A comparison between the reference and reconstructed data for one of the tested configurations is show in Figure 4.3. The obtained reconstruction accuracy is satisfactory, specially considering that the network has not seen any information on the velocity fields. Nevertheless, the network struggles to correctly reconstruct regions that exhibit large gradients. This is particularly true in the unsteady case — the unsteady puffing behavior introduces a temporal variation that further increases the gradients in regions that are already difficult to resolve in the steady case.

Additionally, the results show that better reconstruction accuracy is obtained for the axial velocity components than for the radial components. De Boer claims that a possible reason for this is the fact that the radial velocity is either zero (almost everywhere in the domain) or reaches extreme values with very large gradients (close to the flame). This makes the radial velocity field inherently more difficult to reconstruct for the PINN. De Boer further hypothesizes that the neural network architecture may not be complex enough to capture the high-frequency oscillations near the peaks — the locations of the extrema match well with the reference data, but their amplitudes are underestimated by the PINNs. Alternatively, this may simply be a manifestation of the well-known spectral bias of PINNs (see Section 4.4) and may be independent of the network size and architecture.

All in all, the literature presented in this section demonstrates that PINNs are a promising tool to perform data reconstruction for reacting flows like pool fires. Nonetheless, it is clear that in order to tackle pool fire configurations with practical relevance further research is required.

4.4. Pathologies and extensions

Despite the fact that PINNs have successfully been applied to tackle a wide range of problems, the reconstruction quality that can be obtained by the standard PINN architecture remains limited by a series of pathologies, both theoretical and practical, that are well-documented in the existing body of literature. This section provides an overview of these pathologies and the state-of-the-art mitigating strategies used to overcome such limitations. The goal is not to provide a comprehensive review of all of the complications associated with the use of PINNs; that falls well outside the scope of the preset work. Instead, a collection of well known issues is presented and discussed, alongside possible modifications that have been applied to the baseline PINN architecture to overcome such limitations. Understanding



(a) Axial velocity, u

(b) Radial velocity, v

Figure 4.3: Velocity reconstruction of a laminar pool 2D fire using PINNs. The reconstructed data is compared against the reference data. Subfigure (a) shows the axial velocity reconstruction; subfigure (b) shows to the radial velocity. Reprinted from [26].

these limitations is an important step to decide which modifications, if any, can be used to improve the performance of PINNs in the present work.

Training pathologies of PINNs

From a theoretical perspective little work has been done to provide a solid mathematical justification for the use of PINNs. The first mathematical description of PINNs is the work of Shin et al. [94], where the authors explore the convergence behavior of PINNs for linear second order elliptic and parabolic PDEs. The work of Shin et al. and other related work focuses on trying to answer a fundamental question — can PINNs find the solution to the underlying PDEs using gradient-based optimization algorithms? The answer remains unclear, and studying the convergence of PINNs and their error behavior remains an area of focus for future research [1].

The issue of convergence and uniqueness is a well-known open problem in the field of machine learning — due to the highly non-convex nature of typical objective functions, and considering the stochastic nature of the most commonly used gradient-descent optimization methods, there is no guarantee that a neural network will converge to a unique solution. The application of PINNs is no different. There is no theoretical guarantee that the standard PINN architecture will converge to the global minimum in the highly complex loss landscapes that characterize most problems of relevance. It is not uncommon for such networks to get stuck in local minima and/or arrive at solutions that are erroneous or physically inconsistent.

The reasons for this are well-documented in the literature. PINNs exhibit an inherent training pathology that results in highly complex loss-landscapes that are difficult to navigate for gradient-based optimization algorithms. As a consequence, the baseline PINN architecture may struggle to approximate PDEs even in simple cases. Wang et al. [95] identified this fundamental failure mode of PINNs and attributed it to an inherent stiffness in the gradient flow dynamics of the PINN. This stiffness leads to unbalanced backpropagation gradients during model training, ultimately causing poor training behavior and preventing the network from obtaining sufficient approximation accuracy. The authors explored a series of numerical examples (1D Poisson equation, 2D Helmholtz equation) and show that in all cases the PDE residual term dominates the total loss function. The discrepancy between the different terms of the total loss function is the cause of the unbalance in the backpropagation gradients. This ultimately leads to solutions that do not match the provided data. In this case the solutions did not respect the provided boundary and initial conditions, which effectively drives the network away from approximating

the correct solution, resulting in low reconstruction accuracy across the board.

In a subsequent publication, Wang et al. [96] studied this failure mode in more detail. They again state that PINNs suffer from a remarkable discrepancy of convergence rate in the different components of their loss function, which is responsible for their low reconstruction accuracy [96]. In order to explore the failure mode in more detail the authors studied the training dynamics of PINNs through the lens of their natural tangent kernel (NTK). They traced the aforementioned unbalance between the different terms of the loss function to a more fundamental difference between their corresponding kernel matrices. More specifically, the authors show that the matrix corresponding to the PDE residual term generally has eigenvalues of significantly larger magnitude than other components. As a consequence the PDE residual term dominates the total loss function, causing an unbalance in the training process that ultimately leads to poor network performance.

This inherent limitation of PINNs has been reported by other authors. Krishnapriyan et al. [97] also demonstrate how PINNs can fail to produce satisfactory results in simple settings. They also attribute this fact to the dominance of the PDE residual term in the loss function. They specifically consider a 1D convection equation and a 1D reaction-diffusion problem. In both cases the standard PINN fails to approximate the true solution to the underlying PDE. The authors analyzed the loss landscape of trained PINN models and find that adding the PDE-based residual term leads to a more complex loss landscape that is more difficult to optimize. The authors also investigate the changes to the loss function as the physics-based regularization parameter (which measures the relative importance of the PDE residual on the total loss) is changed. Their results demonstrate that reducing the effect of the PDE residual alleviates the complexity of the loss landscape. Evidently this comes at the cost of obtaining solutions that do not satisfy the PDE, which defeats the purpose of using PINNs in the first place.

Moreover, the work of Krishnapriyan et al. [97] demonstrates that the reason for this failure is not a lack of expressivity in the neural network architecture, but rather optimization difficulties that are inherent to the physics-based soft constraint used in PINNs. The work of Krishnapriyan et al. [97] and that of Zeng et al. [98] point to the differential operator used in the physics-based residual term as the key source of this training pathology. The use of such differential operators is very different from the more standard norm-based regularization used in most machine learning methods. Differential operators can be ill-conditioned, which can lead to numerical instabilities that complicate the training process. For example, the condition number for the regularization operator of the simple 1D diffusion problem considered in [97] scales with $\mathcal{O}(\nu N^2)^2$, where N is the grid size and ν the kinematic viscosity. Similarly, for the 1D reaction-diffusion problem the condition number scales with $\mathcal{O}(\beta N)^2$, where β is the convection coefficient. Both of these scaling relations lead to very high condition numbers. It is not surprising that this ill-conditioned property is introduced by the differential operators into the training of the PINN through the physics-based regularization. These result are in agreement with the aforementioned work of Wang et al. [95, 96] — they provide a theoretical explanation for the dominance of the PDE residual in the loss function, which ultimately leads to poor network performance.

The consensus is clear — PINNs have a well-known failure mode related to an unbalance between the different terms of their loss function. This makes their training process inherently ill-conditioned and limits the reconstruction accuracy that can be obtained by the standard PINN architecture. This pathology arises from their use of differential operators for the physics-based regularization. These differential operators can have high condition numbers, and the use of the L_2 norm to quantify the PDE error squares this condition number and further aggravates the issue. The result is a highly non-convex loss landscape full of local minima that is very difficult to navigate by gradient-based optimization algorithms. In addition to this failure mode, PINNs also suffer from spectral bias [1, 96]. This well-known phenomenon affects feed forward neural networks like the ones used in PINNs, making them more likely to prioritize low-frequency components of the solution while minimizing their loss function. This makes the networks unable to learn high-frequency functions or unable to correctly resolve high-frequency oscillations and regions with large gradients.

Mitigation strategies for improved performance

Thankfully, a series of modifications to the standard PINN have been proposed to address these limitations. One extension that has received a lot of attention are *loss balancing* techniques. These are algorithms that modify the relative importance of the different components of the total loss function by assigning a set of weights w_i. The weights can be dynamically modified at each training iteration (or a frequency specified by the user, e.g. every 10 iterations) to maintain a balance between the different terms of the loss function. The goal is to prevent any single term from dominating the overall loss function. Multiple different loss balancing techniques (sometimes referred to as dynamic weight optimization techniques) are reported in the literature. They demonstrate significant improvements in accuracy compared with the standard PINN. Examples are the work of Wang et al. [95, 96] and van der Meer et al. [99].

As mentioned in Section 4.2, the sampling of the collocation points can significantly affect the reconstruction accuracy of a PINN. For that reason adaptive methods for the sampling of the collocation points have also been proposed. One example are *importance sampling* techniques [100], where the location of the collocation points is changed at each iteration of the training process in a way that increases the density of collocation points in regions with large errors. Other adaptive methods for sampling the collocation points are reported in the literature.

Other extensions to PINNs include domain decomposition techniques (both in space and time) for improved efficiency, the use of novel activation functions, sequential training schemes that mimic time-marching methods, curriculum learning schemes where the complexity of the loss landscape is progressively increased, and causal training techniques. Lastly, it should be noted that the performance of PINNs is heavily dependent on the tuning of its hyperparameters, the choice of network architecture and size, and the choice of the optimization algorithm. The choice of learning rate is considered to be of particularly importance [101]. Xu et al. [78] propose the use of an adaptive learning rate algorithm to iteratively adapt the learning rate during the training process. This is motivated by the work of Loshchilov and Hutter [102] who demonstrated the advantages of using an adaptive learning rate algorithm with warm restarts to improve performance on gradient-descent optimization of ill-conditioned functions. In particular Xu et al. [78] propose a cosine annealing algorithm for the learning rate. Other authors use a fixed learning rate or propose their own learning rate schedules — Raissi et al. [76] used a simpler step-wise learning rate algorithm, while Loshchilov and Hutter propose and compare a series of learning rate schedules in [102].

Table 4.1: Extensions to the baseline PINN to mitigate training pathologies

Technique	Notes / Examples	Related work
Loss balancing	aka adaptive training or dynamic weight optimization	[82, 95, 96, 99]
Sampling of collocation points	e.g. batching techniques, importance sampling	[100, 103–105]
New activation functions	e.g. adaptive activation functions	[106, 107]
Domain decomposition	Particularly useful for high-performance computing	[108, 109]
Sequential learning	Similar to time domain decomposition (time-marching)	[97, 106]
Causal training	Used to enforce causality	[110]
Curriculum learning	Progressively increasing complexity of the loss landscape	[97]
Learning rate scheduling	e.g. cosine annealing algorithm	[75, 78]

In conclusion, despite the fact that PINNs exhibit some training pathologies and lack a complete mathematical theory to prove their convergence to the true PDE solution, several promising extensions exist to mitigate these issues and improve their performance. A summary of the extensions discussed in this section and the related publications is given in Table 4.1. These extensions have been proven to, at the very least, partially mitigate the training pathologies of standard PINNs. The improved performance and training efficiency that can be achieved with these modifications paves the way for the application of PINNs to more complex problems like the one considered in the present work. The extensions presented in this section, and the information regarding the well-known pathologies of PINNs, are used to guide the design of the PINN-based reconstruction algorithm discussed in Chapter 5 (Section 5.3).

Part II

Methodology & Results

5 Methodology

This chapter outlines the methodology employed in this study, beginning with a high-level conceptual framework presented in Section 5.1. Subsequent sections delve into various critical components in detail. Section 5.2 elaborates on the numerical simulations used to obtain ground truth data for the reconstructions. Following this, Section 5.3 provides a comprehensive description of the PINN-based reconstruction algorithm and the surrounding data processing pipeline. The formulation and verification of the physics-informed loss components, including the mixture fraction loss, are explored in Section 5.4. Section 5.5 then discusses how the structure of the flame, that is, its temperature and composition, is modeled in mixture fraction space. Lastly, the chapter concludes with Section 5.6, where the approach for estimating the Heat Release Rate (HRR) from reconstructed quantities is described.

5.1. General framework

A schematic representation of the pool fire flow field reconstruction process is shown in Figure 5.1. The goal of this section is to describe the interplay between the different components of the reconstruction process, establishing a general framework for the methodology and highlighting the tasks that need to be performed to carry out the necessary reconstructions.

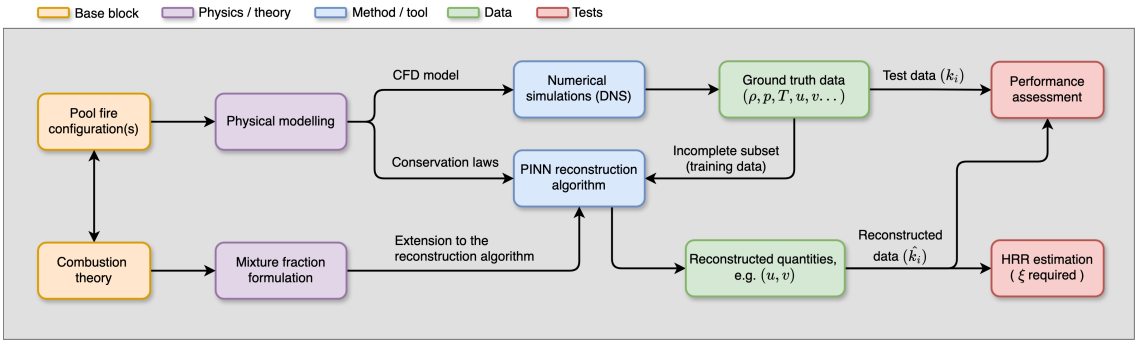


Figure 5.1: From pool fire configuration to reconstructed flow fields. Schematic representation of the flow field reconstruction process. Starting from the pool fire configurations, physical modeling is used to derive the relevant governing laws and setup a CFD model. The former is used to construct the physics-informed reconstruction algorithm. The latter is use to perform numerical simulations that serve as ground truth data, a subset of which is used to guide the reconstruction process. The remaining data is used to assess the performance of the reconstruction algorithm.

Based on the work of Sitte and Doan [26], a specific pool fire configuration has been selected as the target for the reconstruction. This configuration is the starting point for the reconstruction process, as shown on the top left of Figure 5.1. Two important components necessary to perform the reconstructions must be derived from this pool fire configuration. First, a dataset of the pool fire flames is required to guide the reconstruction algorithm and to use as "ground truth" to assess the performance of the

reconstructions. Secondly, the governing equations for the pool fire flames are needed to construct the physics-informed reconstruction algorithm. Both of these items are the result of a physical model that represents the pool fire configuration.

In principle, the flow field data can be derived from experimental data or from numerical simulations. In the present work the data is obtained via a Computational Fluid Dynamics (CFD) simulation of the chosen pool fire configuration. This is done for several reasons. Firstly, as discussed in Chapter 2, diffusion flames like pool fires are notoriously difficult to study experimentally — this is precisely the reason why they are a suitable target for flow field reconstruction methods in the first place. If experimental data could easily be obtained on all the quantities of interest, there would be no need to perform flowfield reconstructions. Numerical simulations are therefore used as a surrogate for experimental data to carefully study the performance of the reconstruction algorithm. The numerical simulations used in the present work are readily available thanks to the work of Sitte and Doan [26]. It is important to ensure that the numerical data has been validated experimentally to ensure the applicability of the reconstruction algorithm to real-life data. The datasets used in the present work have been validated by Sitte and Doan [26] against the experimental data of Moreno-Boza et al. [28], demonstrating that they are a suitable surrogate for experimental data for the problem under consideration.

Secondly, using numerical data allows one to have more control over the inputs of the reconstruction process. The boundary conditions are well defined, and the spatiotemporal resolution of the data can be fixed when performing the simulations. In addition numerical data is free from the uncertainties, noise, and possible measurement errors that can be present in experimental datasets. The numerical simulations used in this study are therefore a "clean" and experimentally validated surrogate for the flow field of a small laminar pool fire flame.

In addition to the numerical simulations, one must also define the set of conservation laws that are fed to the reconstruction algorithm as physical priors to guide the reconstruction process. These conservation laws are a natural result of the modeling that is required to perform the CFD simulations. However, it should be noted that the set of conservation laws used in the reconstruction need not be exactly the same as those used in the CFD model. Generally they should be consistent with one another, since they are trying to describe the same flow problem, but it is possible to select a subset of conservation laws to perform the PINN reconstructions that is different from the one used in the CFD simulations.

As shown in Figure 5.1, the PINN combines the chosen set of conservation laws with a limited subset of the ground truth data to reconstruct missing flow quantities. The numerical simulations used in the present work contain information about thermodynamic variables like pressure (p), density (ρ) and temperature (T), in addition to the velocity fields (u, v) and other flow variables related to the combustion process, like the mass fractions of the different chemical species. From this dataset, only a subset of flow variables are fed to the reconstruction algorithm during training. These are referred to as *training* data for the remainder of this thesis. The variables that are the target of the reconstruction are referred to as *reconstructed* quantities. No training data on the reconstructed quantities is seen by the PINN at any point during training. Instead, the network is tasked with the reconstruction of these unseen quantities from the knowledge of the training data and the relevant conservation laws.

In a typical reconstruction attempt, the training variables could be pressure, density and temperature, with the reconstructed quantities being the two velocity components (u, v). To reiterate, during such a reconstruction attempt the PINN is never fed any data on the velocity profiles, and it must learn to infer such data from the training data and the residuals of the governing equations. Ideally, the reconstruction algorithm should be modular, flexible, and robust, allowing the user to easily perform multiple reconstruction attempts with different choices for the training and reconstructed quantities. The implementation of the reconstruction algorithm is described in more detail in Section 5.3.

As shown in the bottom branch of Figure 5.1, the standard PINN algorithm is extended with the inclusion of the mixture fraction, denoted by ξ and introduced in Chapter 2 (Section 2.4). The intention is that by adding the mixture fraction transport equation into the physics-based regularization of the PINN, the network will learn to capture the structure that characterizes diffusion flames like pool fires and therefore improve the accuracy of its reconstructions. The formulation of the governing equation

for the mixture fraction and its addition to the reconstruction algorithm is described in more detail in Section 5.4.

5.1.1. Performance metrics

The outcome of the reconstruction attempts are the estimations of the neural network for the chosen reconstruction targets. As shown in Figure 5.1, these estimations are compared against the existing data from the numerical simulations to assess the performance of the reconstruction accuracy. Several error metrics are used to evaluate the reconstruction accuracy.

The first metric is the standard error, denoted by ε. This metric is calculated by simply computing the difference between the reconstructed field and the reference field over the entire domain. To highlight the fact that this error metric contains information about the error distribution throughout the domain it is referred to as the *error field*. For a quantity k the error field is defined as:

$$\varepsilon_k(r,z) = \hat{k}\,(r,z) - k\,(r,z) \tag{5.1}$$

where $\hat{k}\,(r,z)$ is the reconstructed field and $k\,(r,z)$ the reference field from the DNS data. Having a space-dependent error metric gives insight into the performance of the algorithm in different spatial regions. This is important since one generally wishes to understand how the algorithm is reconstructing regions of the domain that are intrinsically more relevant that others. In this case, the ability of the network to reconstruct the flow structures near and around the flame is more important than its ability to resolve the far field, which does not contain any interesting flow features. Additionally, studying the distribution of the error across the domain may provide insights into the performance of the network.

Although the error field by itself is a valuable performance metric, it does not easily allow for comparison between different flow variables. The magnitude of this error will be proportional to the magnitude of the flow quantity for which it is measured. Since, for example, temperature and axial velocity have different magnitudes, it is not easy to directly compare their error fields quantitatively. In order to do, another error metric is introduced, the *normalized error field*, denoted by ε_{norm}. As the name indicates this metric is computed by normalizing the error with some constant. The normalization constant is chosen to be the mean value of the DNS data of the flow quantity for which the error is measured. The normalized error field is then calculated as:

$$(\varepsilon_{norm})_k = \frac{\varepsilon_k(r,z)}{\mu_k} \tag{5.2}$$

where μ_k is the average value of quantity k in the DNS data. The normalized error field is a local error metric that allows one to compare the reconstruction accuracy for different flow quantities.

Global error metrics that quantify the overall reconstruction accuracy over the entire domain are also used. The first one is the *average absolute value of the normalized error field*. The absolute value of the normalized error field is taken to prevent regions with positive and negative errors from canceling each other out. The average of this absolute value is then computed, yielding a scalar metric that quantifies the overall reconstruction accuracy over the entire domain.

Additionally, the *mean-square error* (MSE) is also used as an additional global error metric. The MSE of some variable k is defined as the mean of the square of its error field:

$$\mathrm{MSE}_k = \frac{1}{N}\sum_{i=0}^{N}\left[\hat{k}\,(\,\boldsymbol{x_i}\,) - k\,(\boldsymbol{x_i})\right]^2, \text{ where } \boldsymbol{x_i} = (r_i, z_i) \in \mathbb{R}^2 \tag{5.3}$$

where i represents an index over the spatial grid points in the domain where the loss is computed, $k\,(\boldsymbol{x_i})$ is the true value of quantity k at the location $\boldsymbol{x_i}$, and $\hat{k}\,(\boldsymbol{x_i})$ is the reconstructed value given by the PINN at that same point. This yields another global metric that quantifies the error in the estimation of the reconstructed quantity k over the entire domain. It should be noted that the MSE is computed using

the standardized flow variables, unless stated otherwise. This is done to ensure a fair comparison between the MSE of different flow variables, removing the dependency on the absolute value of each quantity.

One last metric used to evaluate the quality of the reconstructions is the estimation of the Heat Release Rate (HRR), as shown on the bottom right of Figure 5.1. The HRR is a parameter of interest for practical studies of flames since it is a good proxy for the destructive power of a flame, allowing comparisons between different flame configurations. The HRR is not a direct output of the PINN algorithm, and hence it must be estimated from the available flow quantities. The process used to perform the HRR estimation is discussed in detail in Section 5.6. It should be noted that the estimation of the HRR requires the mixture fraction to be one of the outputs of the reconstruction process. For that reason, only tests that include the mixture fraction will have an estimated HRR value.

Lastly, it should be noted that all of these performance metrics are used in tandem to analyze any given reconstruction run and draw conclusions regarding its accuracy. Any of these metrics taken in isolation can paint an incomplete picture of the outcome of a reconstruction. The assessment process begins by visually inspecting the normalized error field. Global error metrics are then computed, like the MSE and the average absolute normalized error. These global metrics can be computed over the entire domain or within a specified region. Generally they are computed globally and also within the flame region, which is broadly defined as the region where the majority of interesting flow features are present. The limits of the flame region are defined to be:

$$\text{flame region: } \begin{cases} 0 < r^+ \leq 3 \\ 0 \leq z^+ \leq 15 \end{cases} \tag{5.4}$$

The performance of the network during training is also investigated to understand the evolution of the training process. This generally entails analyzing the evolution of the loss components and the reconstruction errors, defined as the standardized mean square errors for the reconstructed variables, as the training progresses. The evolution of the reconstructed fields themselves is also a valuable tool that is used to measure the performance of a reconstruction attempt and gain insight into the behavior of the PINN.

The remaining sections of this chapter describe the most relevant components of the reconstruction process shown in Figure 5.1, providing details about their development and implementation. These are the numerical simulations used to generate the ground truth data (Section 5.2), the PINN-based reconstruction algorithm and the surrounding data processing pipeline (Section 5.3), the formulation and verification of the physics-informed loss terms including the mixture fraction (Section 5.4), the modeling of the flame structure in mixture fraction space (Section 5.5), and finally the estimation of the HRR from reconstructed flow quantities (Section 5.6).

5.2. Numerical simulation using OpenFOAM

A numerical simulation of the pool fire flame is used as a surrogate for experimental data to perform the flow field reconstructions. This section presents details about the simulation, setup and conducted by Sitte and Doan [26]. The formulation and numerical settings used to obtain the datasets are discussed since they affect the quality and applicability of the obtained results. Additionally, it is important to keep in mind how the pool fire flame is modelled in the numerical simulations to make sure that the physical description embedded in the PINN is compatible with that of the numerical data.

The simulation has been performed using the computational fluid dynamics (CFD) toolbox OpenFOAM-7 [111]. A custom version of the transient solver for diffusion flames *fireFoam* is used, modified to fit the problem under consideration. The chosen configuration is constructed to match the one studied by Moreno-Boza et al. [28] to allow validation by comparison to experimental data.

The governing equations implemented in the *fireFoam* solver are briefly discussed hereinafter. Firstly, the continuity equation is used to enforce mass conservation. Using index notation it is given by:

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_i}{\partial x_i} = 0 \tag{5.5}$$

where ρ is the density of the fluid, x_i are the spatial coordinates, and u_i the velocity components. Momentum conservation is enforced through the Navier-Stokes equations. In the present case the only body force that is considered is that of gravity, which acts along the axial direction and for buoyancy driven flows plays an important role and cannot be neglected. This leads to the following formulation of the Navier-Stokes equations, again using index notation:

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial \rho u_i u_j}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} + \rho g_i \tag{5.6}$$

where τ_{ij} is the deviatoric stress tensor, and g_i represent the body forces acting along x_i. Additionally, conservation of total specific enthalpy (or alternatively, energy) must also be enforced. The corresponding equation, shown below, highlights the fact that enthalpy changes can only occur due to diffusion. The diffusion rate is controlled by $\alpha = \lambda / \rho C_p$, the thermal diffusivity.

$$\frac{\partial \rho h}{\partial t} + \frac{\partial \rho u_i h}{\partial x_i} = \frac{\partial}{\partial x_i}\left[\rho \alpha \frac{\partial h}{\partial x_i}\right] \tag{5.7}$$

Lastly, in the case of a reacting flow with multiple chemical species, the principle of conservation of mass requires not only global mass conservation (enforced by Equation 5.24), but also conservation of the mass fractions of each individual species k. The change in mass fraction of each species is governed by two mechanisms: diffusion, and the rate at which they are produced or consumed as a consequence of the chemical reactions taking place within the domain. The conservation of the species mass fractions is enforced through the following equations (one such equation for each species k):

$$\frac{\partial \rho Y_k}{\partial t} + \frac{\partial \rho u_i Y_k}{\partial x_i} = \frac{\partial}{\partial x_i}\left[\rho \mathcal{D}_k \frac{\partial Y_k}{\partial x_i}\right] + \rho \dot{\omega}_k \tag{5.8}$$

where Y_k are the species mass fractions, $\dot{\omega}_k$ their chemical reaction rates, and D_k is the molecular diffusivity. Equations 5.5 to 5.8 form the system of governing equations used to simulate the pool fire flame.

The studied configuration is a 2D axisymmetric pool fire with n-heptane fuel (C_7H_{16}), modelled at ambient conditions. The oxidizer is ambient air, with an oxygen mass fraction of $Y_{O_2} = 0.233$ and a nitrogen mass fraction of $Y_{N_2} = 0.767$ at a temperature of $T_0 = 300$ K and a base pressure of $p_0 = 100$ kPa. The spatial domain, shown schematically in Figure 5.2, has a size of $50a$ in the axial direction and $10a$ in the radial direction, where a is the pool radius. The domain is discretized into a grid with 601 points in the axial direction and 161 points in the radial direction. In the area of interest where the flame is located the pool radius a is discretized by 91 points, with the same number of points being used in the axial direction.

The baseplate where the flame sits is modelled as an isothermal wall, whereas the top and right boundary conditions are set as pressure outlets at the ambient pressure p_0. The left boundary uses a symmetry boundary condition, since it is the central axis of the axisymmetric flame. As shown in Figure 5.2, the fuel inlet is maintained slightly below the baseplate. This is done to match exactly the setup of Moreno-Boza et al. [28]. In their paper the authors state that the flame behavior is sensitive to dimensions of this offset, as it can cause fuel spillage from the pool into the baseplate. For this reason, the fuel inlet was maintained at the value reported by Moreno-Boza et al., 0.75 ± 0.25 mm.

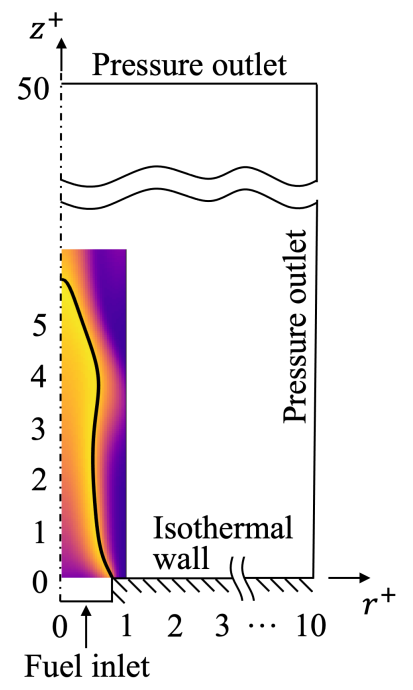


Figure 5.2: Representation of the domain used in the DNS simulations. Coordinates are normalized by the pool radius a. Adapted from [26].

The liquid surface at the fuel inlet is modelled as boundary condition with a temperature equal to the boiling temperature of the fuel, $T_B = 371.5$ K. The mass flow rate of fuel at the surface is determined by computing the evaporation rate that relates the conductive heat transfer from the flame to the fuel with the normal flow velocity at the surface u_n:

$$\rho\, u_n = \frac{1}{l_V}\,\lambda\,\frac{\partial T}{\partial n} \tag{5.9}$$

where

$$l_V = L_V + C_L(T_B - T_0) \tag{5.10}$$

with the latent heat $L_V = 360$ kJ/kg and the liquid heat capacity $C_L = 2.24$ kJ/(kg K). The boundary conditions for the species mass fractions at the interface, known as jump conditions, must also be stipulated. For the fuel mass fraction Y_F the jump condition is given by:

$$\rho\, u_n = \rho\, u_n\, Y_F - \rho\, \mathcal{D}_F\,\frac{\partial Y_F}{\partial n} \tag{5.11}$$

For all other species $k \neq F$ the flux through the flame surface must be zero, yielding:

$$0 = \rho\, u_n\, Y_k - \rho\, \mathcal{D}_k\,\frac{\partial Y_k}{\partial n} \tag{5.12}$$

Based on the experiments conducted by Moreno-Boza et al. [28] and the numerical investigations of Sitte and Doan [26], the critical pool diameter at which puffing occurs has been found to be $2a = 20.1$ mm. The chosen configuration has a pool diameter of 15.9 mm to prevent puffing and ensure that a steady-state solution can be obtained. The resulting spatial resolution within the flame region is approximately 0.088 mm. The combustion process is modelled as the irreversible single-step reaction given by:

$$\mathrm{C_7H_{16}} + 11\,\mathrm{O_2} \longrightarrow 7\,\mathrm{CO_2} + 8\,\mathrm{H_2O} \tag{5.13}$$

with a reaction rate given by the Arrhenius law:

$$K = BT^{\beta}\exp\left(-T_A/T\right) \tag{5.14}$$

where the model constants $\beta = 0$, $B = 5.5 \times 10^7$ m^3/(mol s), and $T_A = 12\,000$ K are chosen to best fit the laminar flame speed curve for heptane–air mixture with a single-step mechanism [26]. The molecular and thermal diffusivities are computed under the assumptions of unity Lewis number and constant Prandtl number:

$$\mathrm{Le} = \frac{\alpha}{D} = 1 \tag{5.15}$$

$$\mathrm{Pr} = \frac{\mu}{\rho\alpha} = 0.7 \tag{5.16}$$

where μ is the dynamic viscosity, computed using Sutherland's law independently of species composition:

$$\mu = \frac{A_s\sqrt{T}}{1 + T_s/T} \tag{5.17}$$

with $T_s = 170.67$ K and $A_s = 1.672 \times 10^{-6}$ kg / (m s K$^{1/2}$). The simulation was performed using second-order Gaussian schemes in space and a first-order Euler scheme in time. A constant time step of $\Delta t = 10^{-5}$ s is used, resulting in a Courant–Friedrichs–Lewy (CFL) number of < 0.1. The simulation is run until convergence of the solution is obtained.

The outputs of the DNS simulation are the two velocity components (u, v), density (ρ), pressure (p), temperature (T), mixture fraction (ξ), specific heat capacity at constant pressure (c_p), heat release rate (HRR), and the mass fractions of fuel (Y_F), oxidizer (Y_O), and reaction products (Y_{CO_2} and Y_{H_2O}). The results of the DNS simulation are shown in Figure 5.3 and Figure 5.4. Only the flow variables that play a role in the reconstructions are shown, namely the two velocity components, density, pressure, temperature, and mixture fraction. Note that the domain is cropped in the radial direction to highlight the interesting features near the flame.

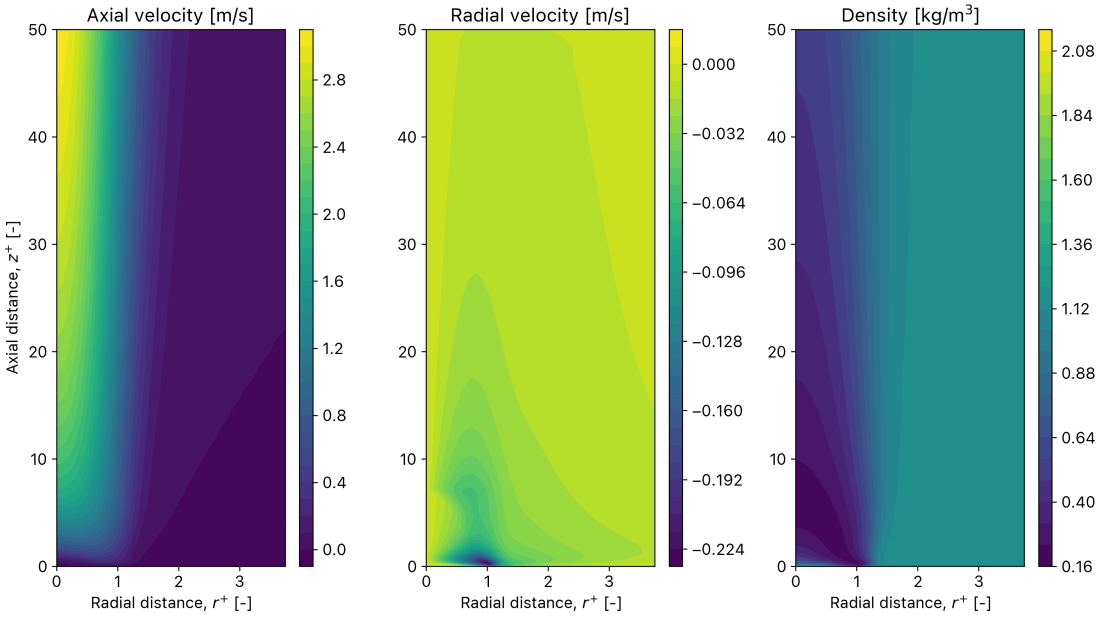


Figure 5.3: Results of the steady-state DNS simulation. From left to right the fields are axial velocity, radial velocity, and density. The domain is cropped in the radial direction to highlight the interesting features near the flame.

The axial velocity field (left side of Figure 5.3) is characterized by the presence of a buoyant plume on top of the flame. In the far field, and everywhere outside of the plume region, the axial velocity is zero. At the inlet the fuel is slowly moving upwards with a very low velocity. The fuel has very little initial momentum — its velocity as it evaporates from the pool and reaches $z^+ = 1$ is around 0.3 m/s. Within the flame region (roughly $z^+ < 8$) the temperature of the gas increases due to the chemical reactions taking place. As a consequence its density reduces, making the air rise due to buoyancy. The air continues to rise and accelerate, increasing its velocity as it moves upwards through the plume region. The highest axial velocity is reached along the centerline of the flame (corresponding to the left boundary of the domain) when the air has reached the top boundary. The maximum axial is velocity is approximately 3.1 m/s.

Since the pool fire is buoyancy driven, and buoyancy acts in the axial direction, the axial velocity dominates the total velocity field. The radial velocity varies from approximately zero in the far field to a maximum value absolute value of 0.225 m/s, that is, one order of magnitude smaller than the axial velocity. The radial velocity field is characterized by a small region of negative velocity close to the base of the flame. This is the air entrainment region — as discussed in Chapter 2, air is entrained into the flame through its base due to the presence of baroclinic vortices caused by the misalignment of the density and pressure gradients. The air entrainment region is characterized by large gradients and is the dominant feature of the radial velocity field.

The density, show on the right of Figure 5.3, is constant and equal to that of air at the ambient conditions in the far field. Close to the inlet the composition of the gas is made up of mostly fuel, and therefore the density is high, reaching a value of 2.15 kg/m^3. As the gas combusts within the flame as it approaches the flame boundary, its temperature increases significantly, decreasing its density. The plume, made up of hot air mixed with reaction products and unburnt fuel, is seen in the density field as a low density region close to the symmetry axis.

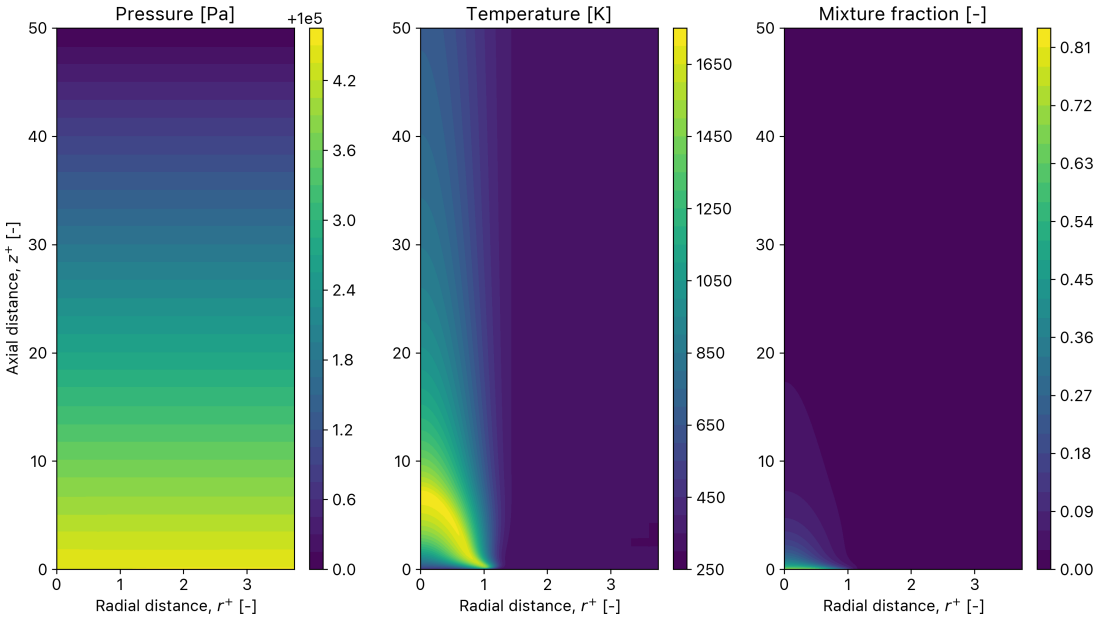


Figure 5.4: Results of the steady-state DNS simulation. From left to right the fields are pressure, temperature, and mixture fraction. The domain is cropped in the radial direction to highlight the interesting features near the flame.

The temperature field, shown in the middle of Figure 5.4, shows the characteristic shape of the flame. The temperature is maximum at the boundary of the flame, where oxidizer and fuel meet at stoichiometric conditions and combust, increasing the temperature of the gas to a maximum value of approximately 1740 K. Within the flame the mixture experiences an increase in temperature as it approaches the flame boundary and the combustion starts to take place. As the gas rises within the plume past the flame boundary, it begins to cool down. By the time it has reached the top boundary of the domain, its temperature has decreased to approximately 700 K. In the far field the temperature is equal to 300 K, the ambient temperature of air set in the simulations.

The pressure field, shown on the left of Figure 5.4, is approximately constant in the radial direction and decreases linearly in the axial direction. This is characteristic of vertical domains with non-zero gravity terms. The pressure at the top of the domain is equal to the ambient pressure of 100 kPa, and it slowly increases as z decreases due to the weight of the air above. The axial gradient of pressure is approximately constant, as predicted by the hydrostatic equation.

Lastly, the mixture fraction field is shown on the right of Figure 5.4. The mixture fraction is equal to zero in the far field, since no fuel is present in that region, and the gas is composed only of air. Very close to the inlet the gas is composed mostly of fuel, and therefore the mixture fraction is high in that region, reaching a maximum value of $\xi = 0.81$. Within the flame the mixture fraction decreases from its highest value at the inlet to the stoichiometric value of $\xi = 0.0622$ at the flame boundary (the calculation of this value is discussed in Section 5.5). Outside of the flame and within the lower portion of the plume the mixture fraction is low but nonzero ($\xi < 0.05$) due to the presence of a small amount of reaction products mixed into the air. Everywhere else in the domain, where no fuel is present, the mixture fraction is zero. The mixture fraction is therefore highly concentrated within the flame, as expected.

5.3. PINN-based reconstruction pipeline

The central element needed to perform the flow field reconstruction is the PINN-based reconstruction algorithm. As discussed in Section 5.1, this algorithm must be capable of reconstructing the desired flow variables using only a subset of the raw DNS data. The goal of this section is to describe the PINN-reconstruction pipeline that has been developed to perform the reconstructions. This novel reconstruction framework is one of the main contributions of the present work. The software is

written in Python using the `Pytorch` library to construct and train the physics-informed neural networks.

The developed reconstruction algorithm, designed for modularity and flexibility, supports running multiple independent reconstruction attempts in parallel. Its modularity makes it easy to extend or modify in the future, adapting it as needed to other physics-informed flow field reconstruction problems. The data processing pipeline enables seamless raw data importation, varied reconstruction testing with complete control over neural network configurations and training, organized result storage, and efficient performance assessment of different reconstructions.

The reconstruction process is broadly made up of three steps. First, the raw data from the DNS simulation must be pre-processed. The processed data is then fed into the main reconstruction model, which trains a PINN on the subset of that data stipulated by the user, alongside many other settings that determine the network's configuration and several important parameters of the training process. The resulting trained model must then be evaluated. This is done during the post-processing phase.

The pre-processing step transforms the raw DNS data into a data structure that is suitable to be fed into the neural network. The raw data is initially stored as an HDF5 binary data file. During pre-processing this single HDF5 file is imported and split into a series of `numpy` arrays that contain the flow data using single precision 32-bit floats. Single precision is chosen over double precision to speed up the computations and limit the memory use of the neural network. Additionally, the data set is processed to remove data points at $r = 0$ to prevent singularities in the computation of the physics-informed loss terms where the radial coordinate appears in the denominator. This is discussed in more detail in Section 5.4.

Lastly, during pre-processing the raw data is standardized such that the resulting arrays have a mean of zero and a standard deviation of one. This standardization process is common in machine learning applications. The different flow variables have values at different orders of magnitudes. For example, the axial velocity has a magnitude of $\mathcal{O}(10^0)$, while the temperature has a magnitude of $\mathcal{O}(10^3)$, the pressure $\mathcal{O}(10^5)$ and the mixture fraction $\mathcal{O}(10^{-1})$ to $\mathcal{O}(10^0)$. Standardizing the data prevents variables with larger absolute magnitudes from having a bigger impact on the neural network's training. The raw data for a variable x is converted to standardized data as follows:

$$x_{\text{stand}} = \frac{x - \mu_x}{\sigma_x} \tag{5.18}$$

where μ_x is the mean of variable x and σ_x is its standard deviation. The standardization process is beneficial for the training of the neural network, but it complicates the computation of the physics-informed loss terms. In order to get around this issue, the flow variables are de-standardized back to their original form before computing the physics-informed terms. This process is represented schematically in Figure 5.5.

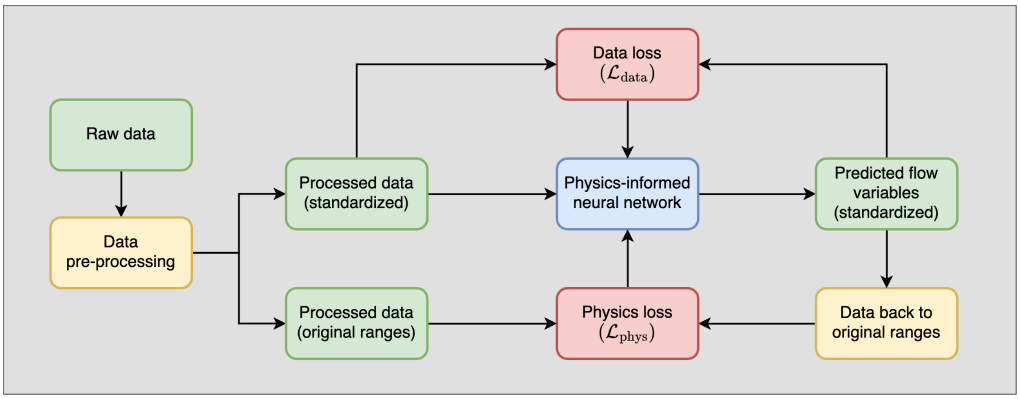


Figure 5.5: Standardization of the raw data and its role in computing loss terms. The estimated flow fields must be de-standardized back to their original ranges to accurately compute the physics-informed loss.

Each reconstruction attempt requires an independent Python file to be created where the user defines the settings for the simulation. The basic structure of these files is shown in Algorithm 1.

Algorithm 1: Basic structure of a PINN reconstruction model

 # Initialize script
– Load packages and initialize available torch device (CPU or CUDA)

 # Define settings
– Set pre-processing parameters
– Set NN hyperparameters
– Set weights of loss components
– Set simulation settings
– Set path to pre-trained model (if applicable)
– Set physical constants (if applicable)

 # Class and function definitions
– Define `Model()` class
– Define `data()` function
– Define sampling functions
– Define loss component functions
– Define `evaluate()` function

 # Preparing training
– Gather, pre-process, and load data by calling `data()`
– Initialize PINN by creating an instance of `Model()`
– Load pre-trained model state (if applicable)
– Initialize optimizer
– Create directories for logging

 # Training loop
– **For each training iteration:**
  • Take optimizer step by calling `evaluate()`, which calls the sampling and loss functions
  • Calculate L2 errors with respect to test data
  • Log iteration statistics
  • Save model state (when applicable)
  • Print progress update (when applicable)
  • Check termination conditions

The first part of the model file contains the definition of all of the necessary settings. This includes pre-processing settings to control how the raw data is processed, the hyperparameters to be used in the PINN (e.g. batch size, learning rate), the weights of the different loss components, and some simulation settings such as the total number of iterations and the frequency at which the model's state is saved. Additionally, the user has the ability to define the path to a pre-trained model to initialize the network's state. Physical constants must also be defined if they are needed to compute the physics-informed loss terms.

The second part of the model file deals with the definition of functions and classes needed during training. There are 5 types of objects that must be defined. The first object is the `Model()` class. This class fixes the structure and configuration of the PINN, which is created as an instance of this class. Then, the `data()` function is defined. This function gathers the raw data, pre-processes it with the appropriate settings, loads the data as tensors into `Pytorch`, and splits the data into a training and testing dataset. Then, two sampling functions are defined to decide how to sample the points at which the data and physics loss are computed (the measurement and collocation points, respectively) at each iteration. By default these are sampled randomly according to the batch-size, but a different sampling routine can also be specified within the sampling function. The loss functions are then defined. Generally one function per loss component is defined. The data loss is computed as the mean square error of the difference between the predicted flow states and the training data evaluated at the measurement

points. The evaluation of the physics-informed loss components is more involved and is discussed detail in Section 5.4.

Finally, the `evaluate()` function is defined. This function plays a special role since it is used in the main training loop to take an optimizer step. Within the evaluate function all of the individual loss component functions are evaluated and the total loss function is composed by taking the weighted average of these terms according to their weights, stipulated by the user. The evaluate function also takes care of performing the backpropagation that updates the network's parameters. This happens in the main training loop.

After completing the object definition, several steps are taken to evaluate these functions and prepare the training of the PINN. First, the `data()` function is called to gather, pre-process, and load the data into the `Pytorch` device. The `Pytorch` device is a context manager of the `Pytorch` library that selects the device used to load the tensors into memory. The reconstruction algorithm automatically detects whether the local machine supports computing on a graphical processing unit (GPU). If this option is not available, it uses the machine's central processing unit (CPU) as the `Pytorch` device.

Then, the PINN is created by evaluating an instance of the `Model()` class. If a pre-trained model has been defined by the user the initial state of the PINN is updated accordingly. Note that the structure of the pre-trained model must be compatible with that of the PINN defined in the `Model()` class. The optimizer is then initialized, and several directories are created to store training logs and results.

Finally, the training of the PINN takes place by looping over the total number of iterations defined by the user. During each iteration, the `evaluate()` function is called. This function first calls the sampling functions to determine the location of the measurement and collocation points, which are sampled randomly at every iteration. Using these points the individual loss components are evaluated. The collocation point are used to compute the physics-informed loss terms. A detailed explanation of how the formulation and verification of these loss terms is given in Section 5.4. The measurement points are used to compute the data loss. This term measures the difference between the estimated flow fields and the DNS data for the training variables, evaluated at the measurement points. Different reconstruction attempts can use a different set of training variables. The standard set of training variables are density, pressure, and temperature, with the two velocity components (the targets of the reconstruction) being excluded from the data loss. In that case, the data loss term is computed by computing the MSE of the difference of each training variable with the DNS data, and then adding each contribution:

$$\text{Density contribution:} \quad \Delta\rho = \rho_{\text{estimated}} - \rho_{\text{DNS}} \tag{5.19}$$

$$\text{Pressure contribution:} \quad \Delta p = p_{\text{estimated}} - p_{\text{DNS}} \tag{5.20}$$

$$\text{Temperature contribution:} \quad \Delta T = T_{\text{estimated}} - T_{\text{DNS}} \tag{5.21}$$

$$\textbf{Data loss:} \quad \mathcal{L}_{\text{data}} = \sum_i \text{MSE}\left(\Delta_i\right) = \text{MSE}\left(\Delta\rho\right) + \text{MSE}\left(\Delta p\right) + \text{MSE}\left(\Delta T\right) \tag{5.22}$$

After evaluating the different loss terms the total loss is computed as a weighted average of the individual terms according to their respective weights, which are set by the user. A backpropagation step is then taken to update the network parameters. The `evaluate()` function zeroes the gradients in the optimizer at the beginning of each iteration to prevent erroneous gradient accumulation between iterations with different batches of data points. A schematic representation of the interplay between the different components of the reconstruction model discussed so far is shown in Figure 5.6.

After the optimizer step is taken, the errors with respect to the training data are computed and stored. A log of the iteration results is appended to a log file, including information about the loss values, the errors with respect to the training data, and the training duration. Additionally, the state of the model's parameters is saved at a specific frequency, in addition to every time the total loss reaches a new minimum value. By doing so the results of the simulation include not only the state of the model at the end of training, but also a multitude of checkpoints that can be imported to evaluate the performance of the model at different stages of the training process. Finally, the script checks certain termination conditions to determine whether to continue with the training process. This could be a specific threshold

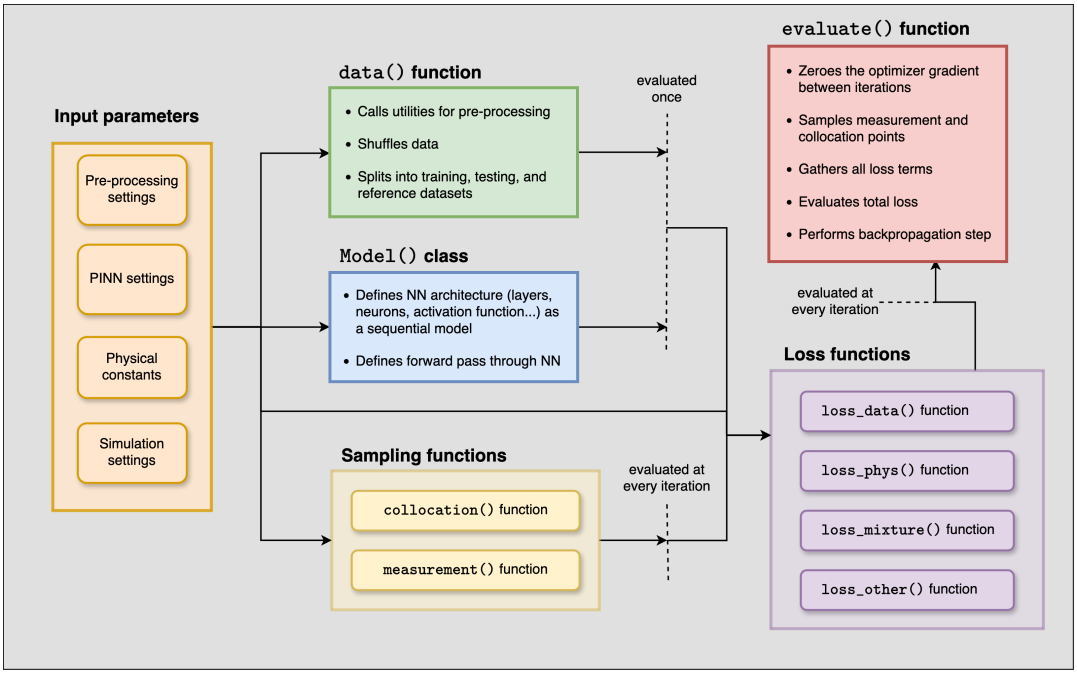


Figure 5.6: Interplay between the components of the PINN-based reconstruction model. Note that only the `evaluate()` and sampling functions are evaluated at every training iteration.

for the loss function or a maximum number of iterations.

The most important settings while running reconstructions and typical values for each of them are summarized in Table 5.1. The result of each reconstruction run is a log file that contains the statistics of the network's performance at each iteration. This includes the values of all loss components, the errors with respect to the DNS data, and the state of the model parameters at the specified checkpoints. Generally between 100 to 200 checkpoints for the model state are saved during a run. All of this information is contained in a Pickle file (.p) of around 300-500 MB in size.

Table 5.1: Typical settings for the PINN reconstruction model

Parameter	**Typical value**
Input variables	Spatial coordinates (r, z)
Output variables (standard tests)	u, v, ρ, p, T
Output variables (mixture fraction tests)	u, v, ρ, p, T, ξ
Training variables	Typically ρ, p, T
Reconstructed variables	Typically u, v
Hidden layers	15
Neurons per layer	75 to 90 (15 per output variable)
Activation function	Sigmoid linear unit (SiLU)
Training iterations	100,000 to 200,000
Weight of loss components	10^2 to 10^4 (defined w.r.t the physics loss)
Batch size	10^4
Optimizer	Adam
Learning rate	10^{-3} to 10^{-6}
Number of saved model states	100 to 200

The final step of the reconstruction process is the post-processing of the results. The post-processing is done with dedicated scripts that locate the results file for a specific run, import and process the data, and generate various plots to investigate the performance during training. These plots generally contain

the evolution of the loss components during training, the residuals of the governing laws used to compute the physics-informed losses, or the evolution of the errors with respect to the DNS data during training.

Additional scripts have been created to select one of the checkpoints at which the model state is available and save that as a separate pickle file. This allows the user to save a specific model state and use it as the pre-trained model to initialize another reconstruction. This is particularly useful to perform separate training runs with different loss functions. For example, it is typical to first train the PINN using a data loss on the subset of flow variables that are available for training, and then follow that with a second training phase where the physics-informed loss is included. Although these two training phases could be combined into one, separating them in two gives the user more flexibility to choose the most suitable data-trained model to use as the initial state for the physics-informed training phase. As such, a full reconstruction run is typically made up of the following steps:

- **Create data training model**. A reconstruction model is created only using a data-loss measured on the subset of flow quantities that are available as training data. No physics-informed loss terms are included.
- **Run and post-process the data-trained model**. The data training is conducted and its training statistics and the resulting flow field estimations are investigated. From the available checkpoints the best performing model state is selected. This generally corresponds to the checkpoint with the lowest loss, although this need not be the case. This model state is then saved into a pickle file to be used as the starting point for the physics-informed training phase.
- **Create physics-informed model (PINN)**. A physics-informed reconstruction model is created. This is generally done by copying the file for the data training model and adapting it to include the physics-informed loss terms. Nevertheless, it is also possible to construct an entirely new reconstruction model from scratch. The data-trained model is now imported and used as the initial state for the physics-informed model. A requirement is therefore that the network architecture of the physics-informed model matches that of the previously trained model. Otherwise the saved state cannot be imported into the PINN, and physics-informed training must be done from scratch.
- **Run and post-process physics-informed model**. The PINN is trained and its results are analyzed to evaluate the performance of the reconstruction attempt by comparison with the DNS data.

5.4. Physics-informed loss terms

As discussed in Chapter 4, the physics-informed loss is the key component that characterizes physics-informed neural networks. PINNs have a composite loss function comprised of a data term (which ensures that the provided data is respected by the PINN) and a physics-informed term that ensures compliance with the governing equations. The latter is constructed by evaluating the residuals of the governing equations given the current prediction of the flow fields. By minimizing these residuals the network is guided towards a solution that is consistent with the physical laws applicable to the problem under consideration. In the present work, the total loss of the PINN takes on the following form:

$$\mathcal{L} = \sum_i \omega_i \cdot \mathcal{L}_i = \omega_{\text{data}}\,\mathcal{L}_{\text{data}} + \omega_{\text{phys}}\,\mathcal{L}_{\text{phys}} + \omega_{\text{other}}\,\mathcal{L}_{\text{other}} \tag{5.23}$$

where $\mathcal{L}_{\text{data}}$ is the data loss, $\mathcal{L}_{\text{phys}}$ is the physics-informed loss, and $\mathcal{L}_{\text{other}}$ are any additional loss components such as those corresponding to boundary conditions. These components are weighted by their respective weights ω_i to compute the total loss $\mathcal{L}$.

In the present study the physics loss is, in almost all cases, the dominant term of the loss function. For that reason its weight is set to $\omega_{\text{phys}} = 1$. The remaining loss terms, which are generally smaller than $\mathcal{L}_{\text{phys}}$, can now be adjusted through their respective weights. The selection of appropriate weights during training has a significant effect on the performance of the network. This is discussed in more detail in Chapter 6 (Section 6.1).

The formulation of the physics-informed loss is problem dependent. In the present study there are two components that make up this loss. Initially, the physics loss is made up of a single component which enforces compliance with the governing fluid equations, namely the continuity equation and the Navier-Stokes equations. This term is referred to as the *momentum loss* for the remainder of this thesis. Additionally, the extension of the reconstruction algorithm with the mixture fraction requires an additional physics-informed term to be added to the loss function. This second component of the physics loss is referred to as the *mixture loss*, and it enforces compliance with the mixture fraction transport equation.

The remainder of this section discusses how these two physics-informed loss components are formulated through the derivation of the corresponding conservation laws. Additionally, the verification of these loss components is also presented and discussed.

5.4.1. Formulation of the momentum loss

The momentum loss is the main component of the physics-informed loss function that characterizes the PINN reconstruction algorithm. It is made up of the residuals from three governing equations, namely the continuity equation (which enforces mass conservation) and the Navier-Stokes equations along the two axes (which enforce momentum conservation).

To derive the formulation of the momentum loss it is important to clearly define the spatial domain and coordinate system to be used. The DNS data, and therefore all of the reconstruction attempts, use a 2D spatial domain, as shown in Figure 5.2. For the derivation of the momentum loss it is key to recall that the flow under consideration is actually three dimensional and cylindrical. The axial symmetry of the flow allows the reduction from three cylindrical coordinates (r, θ, z) to two axisymmetric coordinates (r, z), allowing the flow to be represented in 2D space. These two coordinates correspond to the radial (r) and axial (z) directions, respectively. The coordinate system used in the derivation of the momentum loss and throughout the remainder of this text is shown in Figure 5.7. The velocity components along the (r, θ, z) axes are denoted by (v, w, u).

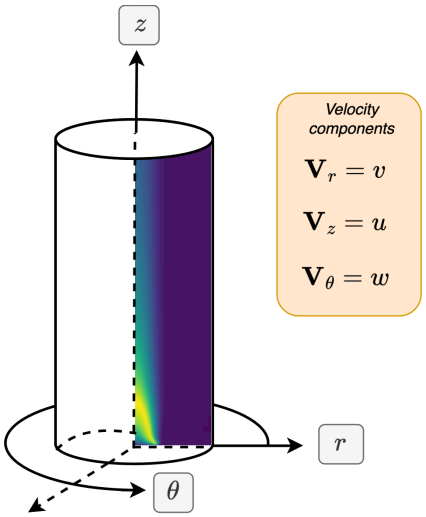


Figure 5.7: Coordinate system used to derive the momentum loss. The flame is represented by the temperature field. The velocity components along each axis are shown on the top right.

The general conservation equations are shown below. These are the continuity equation (Equation 5.24) and the Cauchy momentum equation (Equation 5.25), from which the Navier-Stokes equations are derived [112]:

General conservation laws

$$\text{Continuity equation:} \quad \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{V}) = 0 \tag{5.24}$$

$$\text{Cauchy momentum equation:} \quad \rho \frac{D\boldsymbol{V}}{Dt} = \nabla \cdot \boldsymbol{\sigma} + \rho \boldsymbol{g} = -\nabla p + \nabla \cdot \boldsymbol{\tau} + \rho \boldsymbol{g} \tag{5.25}$$

where $\boldsymbol{V}$ is the velocity vector, ρ is the density, D/Dt is the material derivative operator, p is the pressure, $\boldsymbol{g}$ is the vector of body forces, $\boldsymbol{\sigma}$ is the true stress tensor, and $\boldsymbol{\tau}$ is the deviatoric stress tensor. Several assumptions are made to derive the necessary formulation of these conservation laws to be used in the momentum loss. These assumptions are listed in Table 5.2.

Table 5.2: Assumptions made in the derivation of the momentum loss

Assumption	**Effect**
Steady flow	Time derivatives are zero
Compressible flow	Density gradient is not neglected
Axisymmetric flow	Tangential derivatives are zero
Zero tangential velocity	Tangential velocity is zero
Only body force is gravity	No other body forces
Non-uniform shear viscosity	Viscosity computed from Sutherland's law
Newtonian fluid	Provides a constitutive relation for the stress tensor

The assumptions shown in Table 5.2 are now applied to the governing equations (Equation 5.24 and 5.25) to formulate the momentum loss. A detailed derivation is provided in Appendix A. The results of this derivation are the formulation of the residuals of the continuity and Navier-Stokes equations in cylindrical coordinates, shown below. These residuals are defined everywhere in the domain and are therefore functions of the spatial coordinates (r, z). To compute the momentum loss the mean square errors of the three residuals are calculated and then added together:

Formulation of the momentum loss

$$\text{Continuity residual:} \quad \mathcal{E}_{\text{cont}} = \rho \left[\frac{\partial v}{\partial r} + \frac{\partial u}{\partial z} + \frac{v}{r} \right] + v \frac{\partial \rho}{\partial r} + u \frac{\partial \rho}{\partial z} \tag{5.26}$$

$$R \text{ momentum residual:} \quad \mathcal{E}_{\text{r-mom}} = \rho \left[v \frac{\partial v}{\partial r} + u \frac{\partial v}{\partial z} \right] + \frac{\partial p}{\partial r} + \frac{1}{r}(\tau_{\theta\theta} - \tau_{rr}) - \frac{\partial \tau_{rr}}{\partial r} - \frac{\partial \tau_{zr}}{\partial z} \tag{5.27}$$

$$Z \text{ momentum residual:} \quad \mathcal{E}_{\text{z-mom}} = \rho \left[v \frac{\partial u}{\partial r} + u \frac{\partial u}{\partial z} \right] + \frac{\partial p}{\partial z} - \frac{\tau_{rz}}{r} - \frac{\partial \tau_{rz}}{\partial r} - \frac{\partial \tau_{zz}}{\partial z} - \rho g_z \tag{5.28}$$

$$\textbf{Momentum loss:} \quad \mathcal{L}_{mom} = \text{MSE}(\mathcal{E}_{\text{cont}}) + \text{MSE}(\mathcal{E}_{\text{r-mom}}) + \text{MSE}(\mathcal{E}_{\text{z-mom}}) \tag{5.29}$$

Note that the residuals of the Navier-Stokes equations and that of the continuity equation require the computation of derivatives of several flow variables (velocity components, pressure, density, stress tensors...) with respect to both r and z. As mentioned in Chapter 4 this is done using automatic differentiation. It is important to note that both the inputs (r, z coordinates) and the outputs (u, v, ρ, p, T...) of the neural network are standardized. For this reason, the derivatives that can be computed with automatic differentiation are relative to the standardized coordinates, and cannot be directly applied to compute the momentum loss. The chain rule is used to express the true derivatives of the flow variables (those with respect to the original, non-standardized coordinates) as products of derivatives with respect to standardized coordinates. For example, the derivative of the axial velocity u in the z direction is

given by:

$$\frac{\partial u}{\partial z} = \left(\frac{\partial u}{\partial z_{\text{standardized}}}\right) \cdot \left(\frac{\partial z_{\text{standardized}}}{\partial z}\right) = \left(\frac{\partial u}{\partial z_{\text{standardized}}}\right) \cdot \left(\frac{\partial z}{\partial z_{\text{standardized}}}\right)^{-1} \tag{5.30}$$

Since all derivatives on the right side of Equation 5.30 are taken with respect to the standardized coordinates, they can be computed using automatic differentiation. The chain rule is applied in this way to compute all necessary derivatives needed to evaluate the momentum loss.

5.4.2. Verification of the momentum loss

The formulation of the momentum loss derived in Section 5.4.1 is complex, and its evaluation could be prone to numerical errors due to the presence of $1/r$ factors in the residual equations. Despite the fact that the raw data is pre-processed to remove the singularity at $r = 0$, as discussed in Section 5.3, small values of r may still be problematic and cause numerical instabilities. Additionally, the formulation has been derived from scratch starting from first principles, and therefore it could contain errors. For these reasons it is necessary to verify the formulation before it can be applied in any reconstruction attempts.

To verify the momentum loss it is important to recall that its evaluation relies on the computation of derivatives using automatic differentiation. Automatic differentiation allows one to directly compute derivatives between nodes of the computational graph of the network by exploiting the fact that all operations inside a neural network are essentially a sequence of arithmetic operations (matrix multiplications, additions, etc...). As such, the derivatives of any variable in the computational graph with respect to another variable can be computed by sequentially applying the chain rule through the computational graph.

Because of the fact that automatic differentiation requires a computational graph to compute derivatives, the evaluation of the momentum loss is always coupled with the neural network where it is applied. In other words, the momentum loss cannot be evaluated independently from a PINN since it must always take place within the computational graph of a neural network. This fact constraints the way in which the verification of the loss can be performed. The verification is composed of two steps:

1. **Verification of the gradient evaluation with automatic differentiation**. A synthetic dataset is used to verify that the calculation of gradients using automatic differentiation is being performed correctly. To do so, a PINN is trained on the synthetic dataset and the gradients of the flow variables are then computed using automatic differentiation. These gradients are compared to their exact values, which are known by computing the gradients of the flow data in the synthetic dataset analytically.
2. **Verification of the momentum loss formulation using DNS data**. A PINN is then trained on the entire DNS dataset. The momentum loss is evaluated at every training iteration, but it is never used to train the network. The PINN is trained only on the data loss, and the evolution of the momentum loss as the PINN learns (i.e. overfits) the DNS can be analyzed. If the momentum loss is formulated correctly it should decrease as the PINN learns to reproduce the DNS dataset.

Verification of gradient evaluation with automatic differentiation

A synthetic dataset has been constructed to verify the calculation of derivatives with automatic differentiation. The synthetic dataset is comprised of velocity, density, temperature, and pressure fields computed as a function of (r, z) using the relations shown below. These relations have been constructed, somewhat arbitrarily, by trying to simulate the complexity, shape, and magnitude of the corresponding

fields in the DNS data.

$$\text{Axial velocity:} \quad u = \sin(r)\cos(z) \tag{5.31}$$

$$\text{Radial velocity:} \quad v = 5rz + r^2 \tag{5.32}$$

$$\text{Density:} \quad \rho = 1.25 + \sqrt{r} \tag{5.33}$$

$$\text{Pressure:} \quad p = 10^5 + 100z \tag{5.34}$$

$$\text{Temperature:} \quad T = 500 - \frac{1}{4}r \tag{5.35}$$

The spatial domain has dimensions $0 < r \leq 10$ and $0 \leq z \leq 50$ to match the size of the domain normalized by the pool radius. The domain is discretized into a mesh with 10 points in the radial direction and 100 points in the axial direction. The resulting mesh spacing is ≈ 0.2 units in the radial direction and ≈ 0.1 in the axial direction. The low spatial resolution was chosen in order to reduce the computational cost of the data training and speed up the verification process. The synthetically generated data fields are shown in Figure 5.8. A PINN is trained on the synthetically generated data until a sufficiently low value of the data loss is obtained. The settings for this data training run are shown in Table 5.3. The resulting fields estimated by the PINN after data training are shown in Figure 5.9. By comparison with Figure 5.8, it can be seen that the network is capable of learning (i.e. overfitting) the synthetically generated very well.

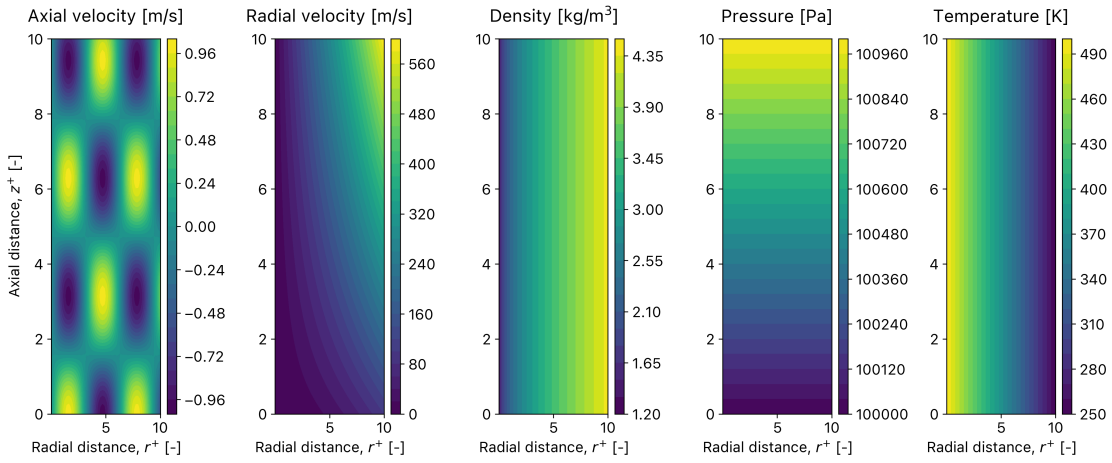


Figure 5.8: Synthetically generated dataset for momentum loss verification.

Table 5.3: Simulation settings for the verification of gradients on synthetically generated data

Parameter	**Value**
Input variables	(r, z)
Ouput variables	(u, v, ρ, p, T)
Number of hidden layers	15
Number of neurons per layer	75 (15 per output variable)
Activation function	Hyperbolic tangent
Weight normalization	No
Training iterations	$150,000$
Batch size	10^4
Optimizer	Adam
Learning rate	10^{-3}

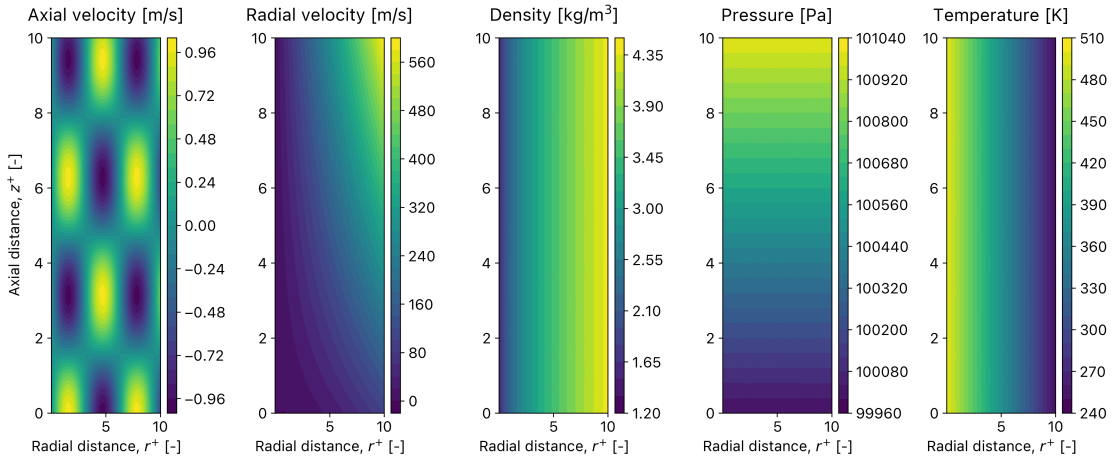


Figure 5.9: Resulting PINN-estimated fields after training on the synthetically generated data.

After the PINN has trained on the synthetic data and is capable of accurately reproducing the flow fields, their gradients along the r and z direction are evaluated using automatic differentiation. The true gradients of each flow variable are known exactly by taking the partial derivatives of the analytical equations Equation 5.31 to 5.35 with respect to r and z. A comparison between the exact gradients and those computed by the pre-trained PINN is shown in Figure 5.10 to Figure 5.14, with each figure corresponding to one of the flow field variables u, v, ρ, p, T.

Consider first the comparison of the axial velocity gradients, shown in Figure 5.10. It can be seen that the PINN is capable of estimating both the radial and axial gradients very well, reproducing them essentially perfectly. However, consider now the radial velocity gradients, shown in Figure 5.11. It can be seen that the gradients computed using automatic differentiation have the correct shape but show some significant erroneous oscillations. This is problematic and must be addressed. The same oscillations can be observed in the gradients of density, pressure, and temperature (Figure 5.12 to Figure 5.14). Due to the way the synthetic data fields are constructed, they all have a constant gradient in one of the two spatial directions. This makes the oscillations that appear when using automatic differentiation clearly visible.

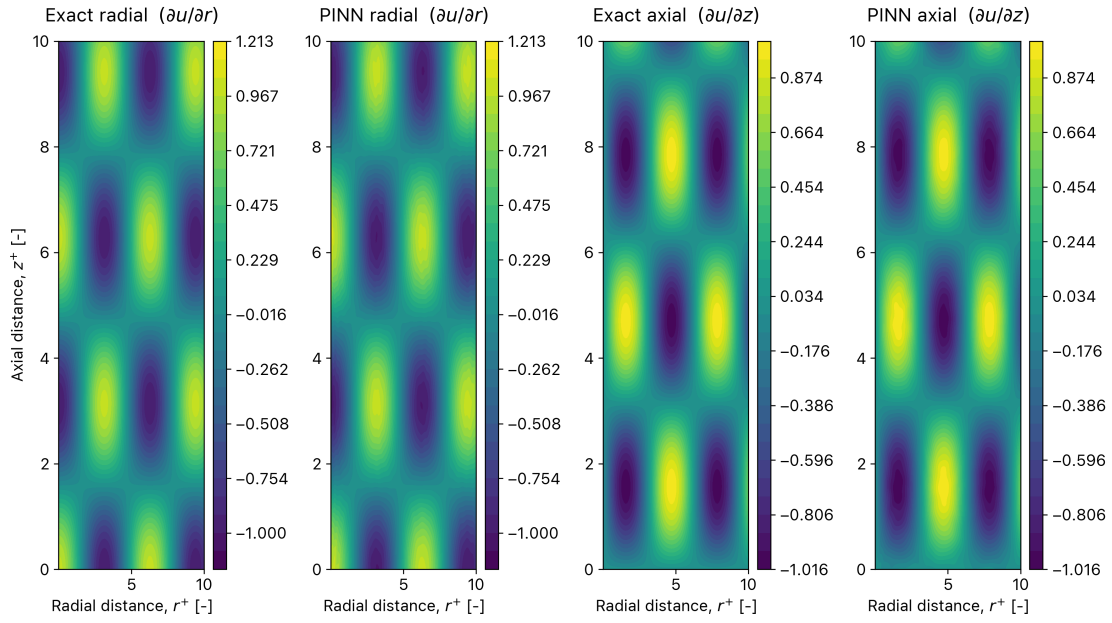


Figure 5.10: Comparison of exact and PINN gradients of axial velocity (synthetic data)

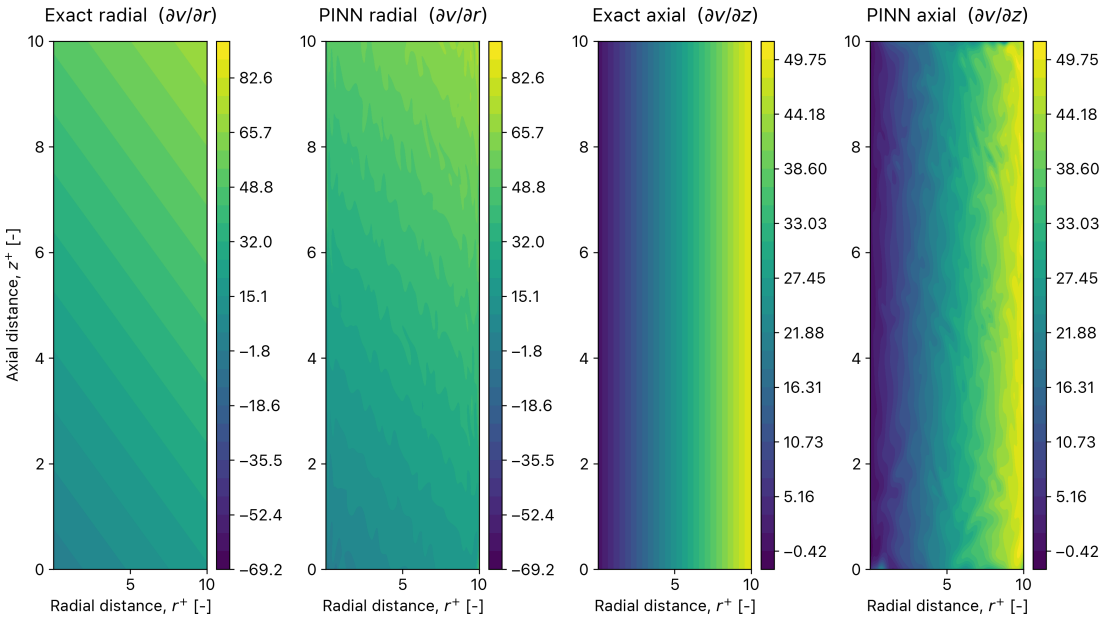


Figure 5.11: Comparison of exact and PINN gradients of radial velocity (synthetic data)

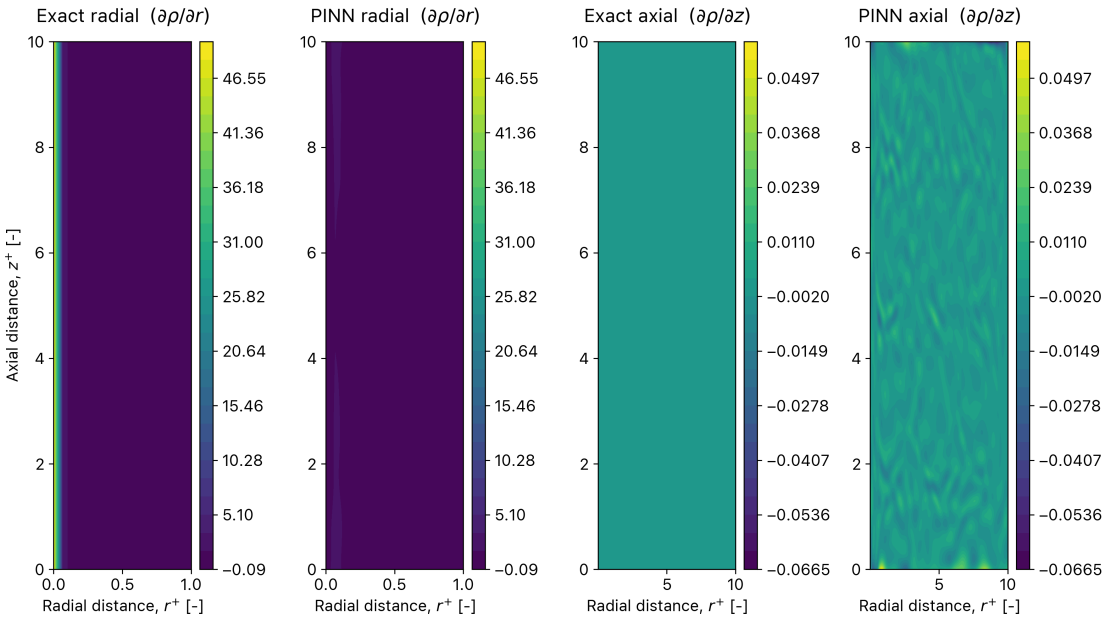


Figure 5.12: Comparison of exact and PINN gradients of density (synthetic data)

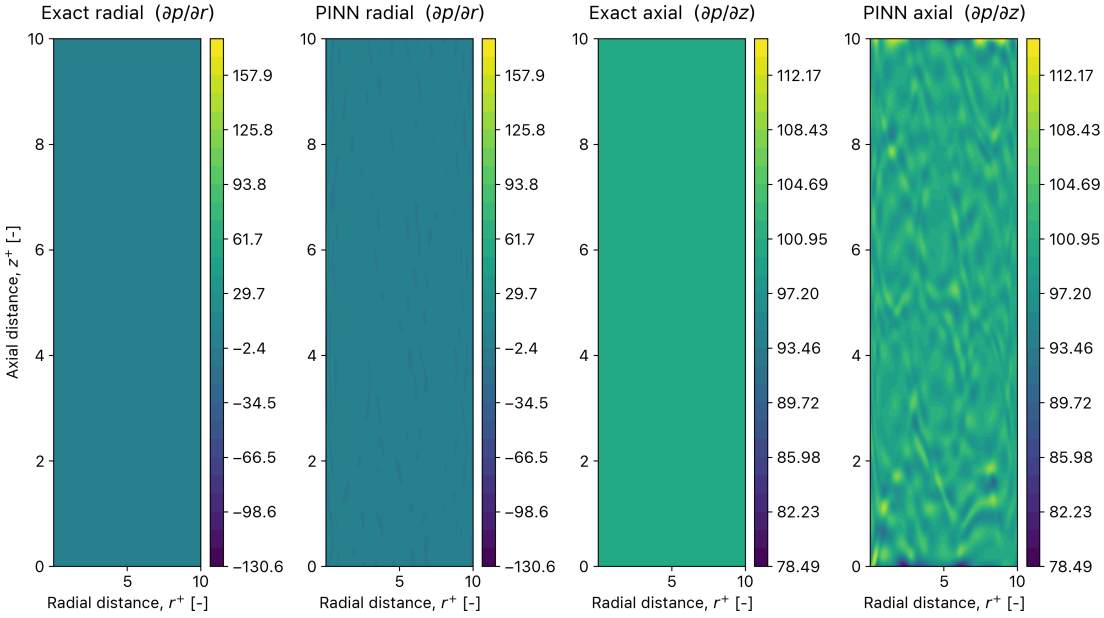


Figure 5.13: Comparison of exact and PINN gradients of pressure (synthetic data)

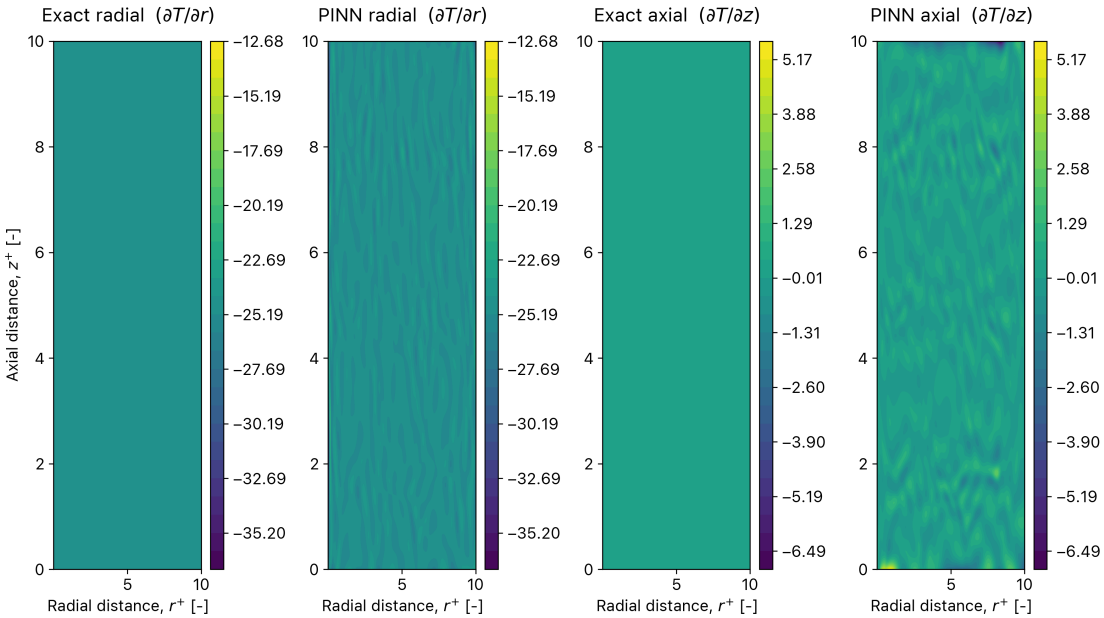


Figure 5.14: Comparison of exact and PINN gradients of temperature (synthetic data)

After some investigation these oscillations were traced back to suboptimal choices for the neural networks parameters. Two modifications are made to address the issue. Firstly, following the work of Sitte and Doan [26] and Raissi et al. [76], the activation function is changed from a hyperbolic tangent to a sigmoid linear unit (SiLU), also known as a swish function. Additionally, the network is modified to include weight normalization. By doing so the magnitude of the network's weights are decoupled from their direction, reparametrizing the weights and preventing any singular weight from having too large of an effect in the network. This technique is known to improve the conditioning of the optimization problem and to speed up convergence of stochastic gradient descent [113]. The updated simulation settings are shown in Table 5.4

Table 5.4: Updated simulation settings to remove erroneous gradient oscillations

Parameter	Value
Input variables	(r, z)
Ouput variables	(u, v, ρ, p, T)
Number of hidden layers	15
Number of neurons per layer	75 (15 per output variable)
Activation function	Sigmoid linear unit (SiLU)
Weight normalization	Yes
Training iterations	$150,000$
Batch size	10^4
Optimizer	Adam
Learning rate	10^{-3}

The new estimated gradients computed by the updated PINN are shown in Figure 5.15 to Figure 5.19. In general, it can be seen that thanks to the modifications made, the oscillations are now much milder. In the case of axial velocity (Figure 5.15) the predictions are just as good as before making the modifications. In the case of radial velocity (Figure 5.16) the oscillations that were present prior to making the modifications have now disappeared. Some small oscillations can still be observed in for the estimated radial velocity gradient in the top right corner of the domain. Nevertheless, these are much milder than before. For the pressure and temperature (Figure 5.18 and Figure 5.19) the oscillations have significantly decreased (note the scale of the vertical axis in the figures). Finally, for the density (Figure 5.17) the oscillations on the axial gradient have completely disappeared. Nevertheless, the radial velocity gradient shows a significant discrepancy with the exact solution at the left boundary of the domain. This discrepancy was also present before making the modifications, see Figure 5.13. The PINNs cannot capture the high-gradient region of the radial derivative field near the left boundary. This is attributed to the low spatial resolution of the synthetic mesh in the radial direction in comparison with the large gradient of the radial derivative of density, which results in aliasing near the left boundary. The exact solution decreases as $1/2\sqrt{r}$; this results in a fast decrease near the left boundary that cannot be accurately resolved with the chosen mesh resolution.

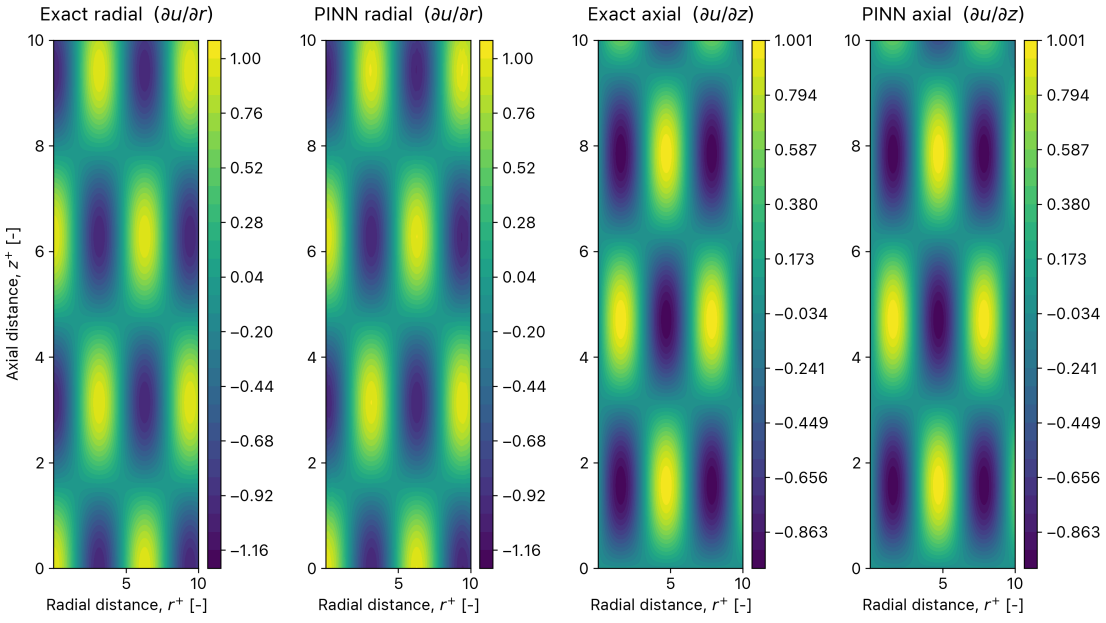


Figure 5.15: Comparison of exact and updated PINN derivatives of axial velocity (synthetic data)

Overall, the results shown in Figure 5.15 to Figure 5.19 show that the PINN is capable of accurately

computing the gradients of flow variables in both the radial and axial direction using automatic differentiation. This verifies the calculation of derivatives used to evaluate the momentum loss. Still, it is worth noting that oscillations may still occur when the grid spacing is not small enough to accurately resolve regions with large gradients. Additionally, oscillations may also appear close to the domain boundaries — the network does not have access to information of the flow variables outside of the domain and therefore struggles to accurately estimate the gradients in those regions, resulting in erroneous oscillations.

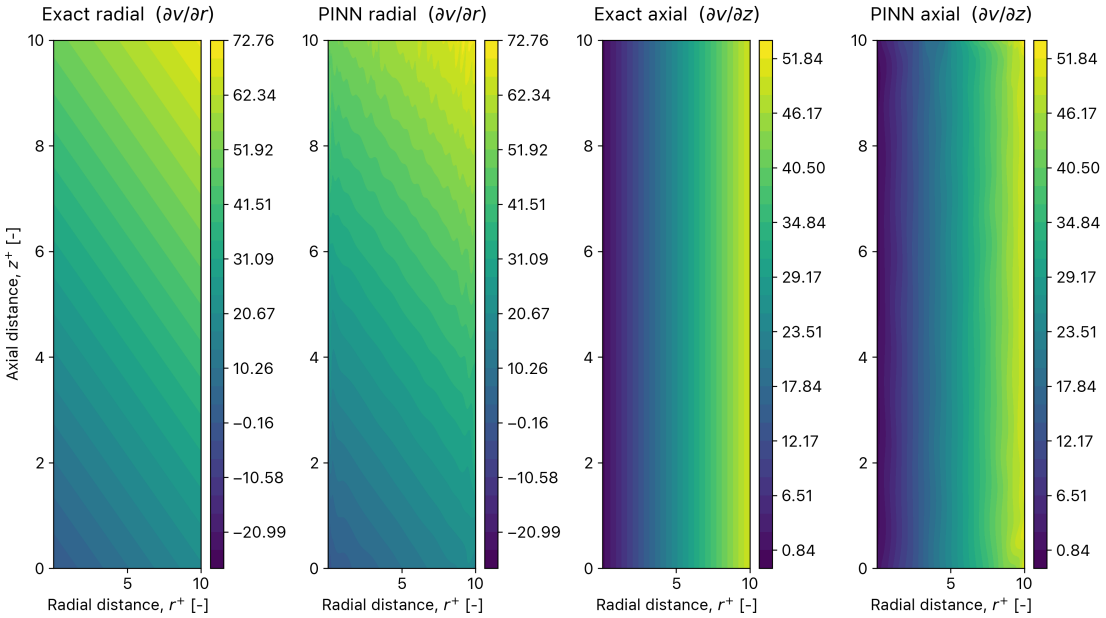


Figure 5.16: Comparison of exact and updated PINN derivatives of radial velocity (synthetic data)

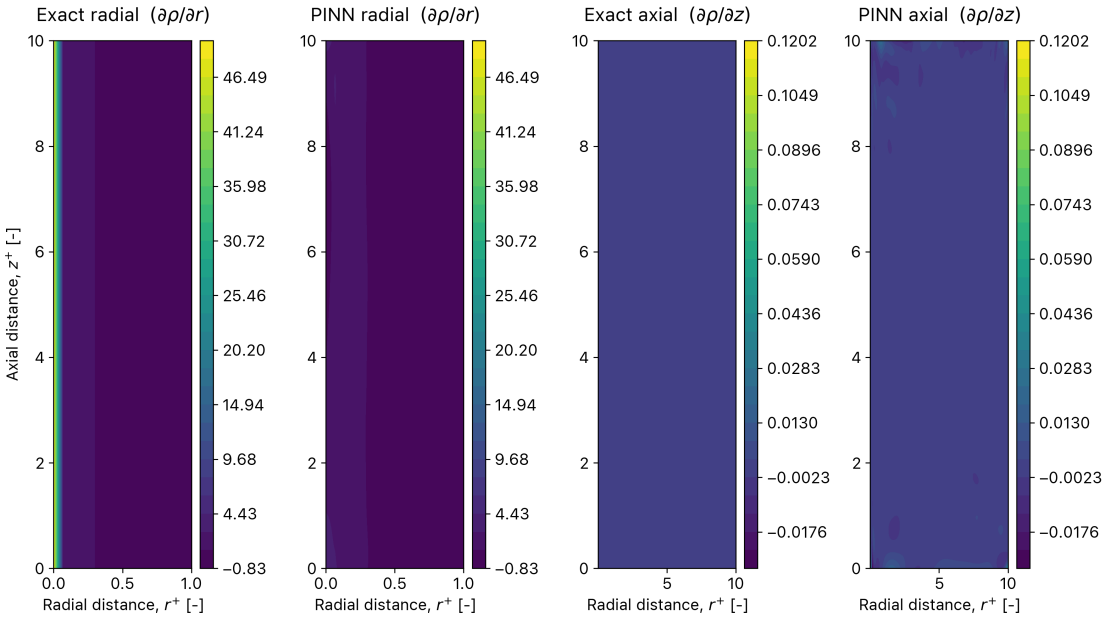


Figure 5.17: Comparison of exact and updated PINN derivatives of density (synthetic data)

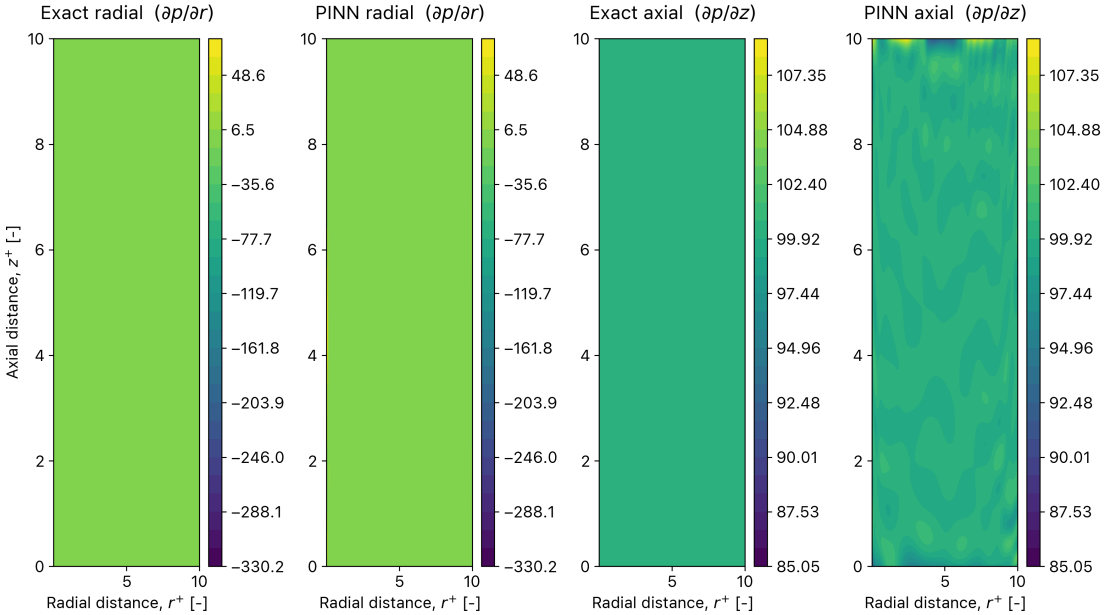


Figure 5.18: Comparison of exact and updated PINN derivatives of pressure (synthetic data)

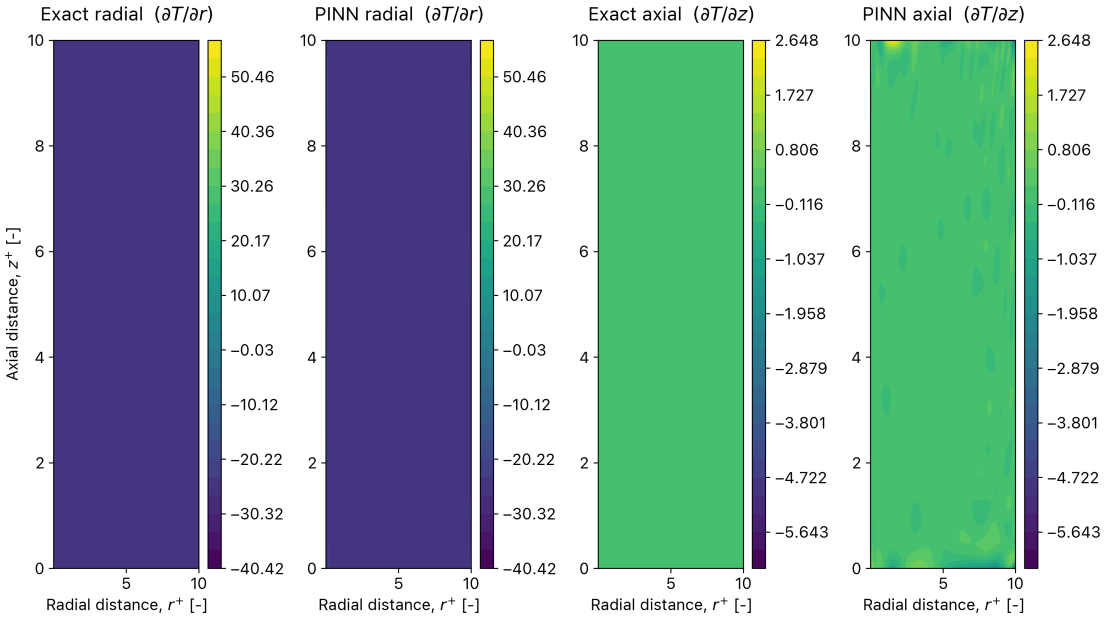


Figure 5.19: Comparison of exact and updated PINN derivatives of temperature (synthetic data)

Verification of the momentum loss formulation using DNS data

With the verification of the gradients completed, the formulation of the momentum loss can now be verified. The goal is to ensure that the derived formulation of the continuity and Navier-Stokes residuals, shown in Appendix A, is correct. In order to do so, a PINN is trained using a data loss measured on the entire DNS dataset. During training, the momentum loss is evaluated and its value stored, but it is not included in the loss function of the network. Therefore, it does not influence the training process — the PINN is only training on the DNS data loss. The evolution of the two loss components during training is

shown in Figure 5.20. It can be seen that as the data loss decreases and the estimation of the flow fields improve, the momentum loss also decreases. This verifies that the momentum loss is compatible with the data loss — an improvement in the flow field estimations leads to a reduction in the momentum loss, as desired.

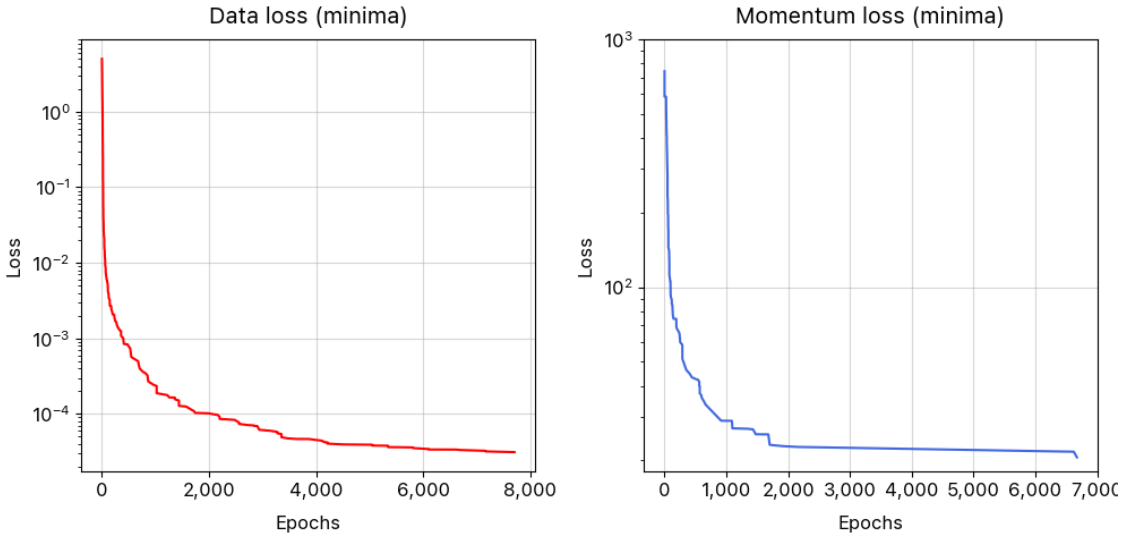


Figure 5.20: Verification of the momentum loss formulation. A PINN is trained using only a data loss on the entire DNS dataset. The momentum loss is evaluated at every iteration but not included in the network's loss function. As the data loss decreases so does the momentum loss — an improvement in the predicted fields leads to a decrease in the momentum loss.

Figure 5.20 demonstrates that an improvement on the data fields yields a significant decrease in the momentum loss, hence verifying its formulation. As the predicted fields move away from the random initial predictions towards the correct solution the data loss decreases rapidly to 10^{-4} in the first 30 thousand iterations. During this time the momentum loss also experiences a very sharp decrease, highlighting the compatibility of the two loss terms and indicating that its formulation is consistent with the DNS data.

Lastly, it is worth noting that the momentum loss remains several orders of magnitude higher than the data loss throughout training. On the one hand this could be expected since the momentum loss has not been included in the network's loss function, and therefore it has not been actively targeted by the optimization process. Nevertheless, the prediction of the flow fields at the end of data training is very accurate. Figure 5.21 shows a comparison between the DNS data and the PINN prediction for the axial velocity. The remaining flow fields are not shown for the sake of brevity, but equivalent levels of accuracy are obtained in all fields. This begs the question — why does the momentum loss remain higher than the data loss when evaluated on fields that match those from the DNS almost exactly?

To answer this question it is important to first focus on the initial loss evaluations at the start of training. At that point the parameters of the network are randomly initialized and the predictions for the flow fields are all completely erroneous and random. Even then, the data loss initially takes a value of just under 10^1 while the momentum loss has a value of approximately 10^3. This partially explains the relative high value of the momentum loss with respect to the data loss at the end of training — the discrepancy is present from the beginning of training and is inherent to the way these losses are calculated. The gap between the two losses does increase during training, with the data loss decreasing by almost five orders of magnitude while the momentum loss only decreases by two. This can be partially attributed to the fact that the momentum loss is not included in the network's training, while the data loss is. As such the latter is being directly targeted by the optimization process, whereas the decrease in momentum loss is purely a consequence of an improvement in the data fields.

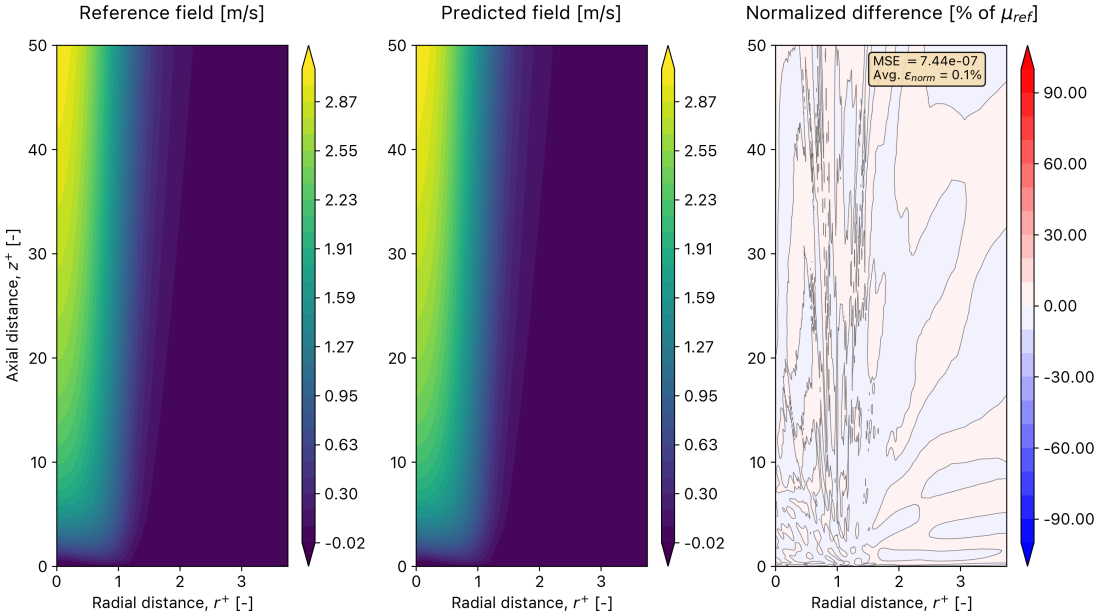


Figure 5.21: Comparison between DNS and PINN-estimated axial velocity after training on DNS data. The PINN is capable of very accurately reproducing the flow fields, as expected. Same levels of accuracy are obtained accross all other variables.

Even then, there seems to be an inherent discrepancy in the range of the two loss terms. The data loss ranges from 10^1 for completely erroneous fields to at low as 10^{-4} after data training. The physics loss however can be as high as 10^4 for erroneous fields and only decreases to 10^1 during data training. This discrepancy is in agreement with the findings of previous work [93], where similar discrepancies were found between the data and momentum loss terms. To investigate this further, the data-trained network is used to evaluate the residuals that make up the momentum loss at the end of training. The results are shown in Figure 5.22 and Figure 5.23.

Figure 5.22 shows that the residuals are generally low throughout the domain. The continuity residuals is high only near the bottom boundary of the domain and low everywhere else, with an average value in the order of 10^{-2}. The radial momentum residual also has an average value in the order of 10^{-2}, but it shows regions with oscillating positive and negative residuals that take on higher values in the plume region above the flame. The same is true for the axial momentum residual, which has an average value of 10^{-1} and shows two thin regions of positive and negative residuals in the plume region.

To get a better understanding of the magnitude of the residuals Figure 5.23 shows their absolute magnitude in a logarithmic scale. Indeed, all three momentum fields are generally low almost everywhere in the domain, ranging from 10^{-1} to 10^{-4}. Nevertheless, the boundaries of the domain and the plume region are problematic. In these regions the residuals show oscillations that spike their values up to 10^2 to 10^3. This oscillations are likely related to the oscillations that appear in the computation of gradients using automatic differentiation. To confirm this, the derivatives of the velocity components, density, and pressure are shown in Figure 5.24 to Figure 5.27.

In the plume region the derivatives of axial velocity and density are high due to the flow features present in that region. The bottom boundary exhibits sharp variations in the computed gradients — this is particularly visible in the axial derivative of the radial velocity component, Figure 5.25. In addition, the radial gradient of the pressure shows some oscillations, specially in the plume region.

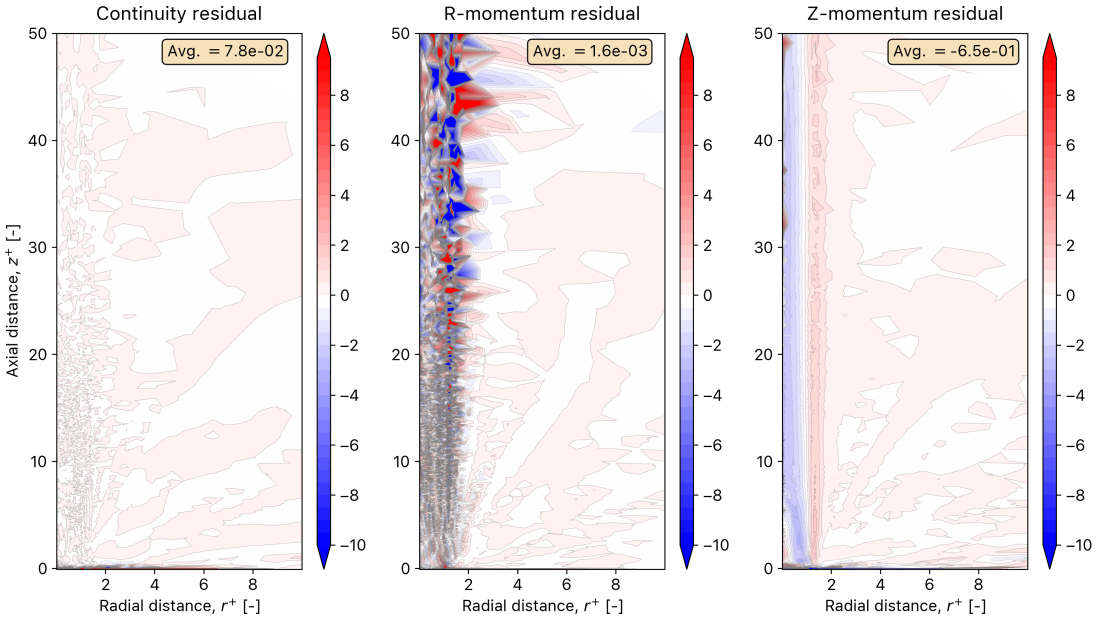


Figure 5.22: Continuity and momentum residuals after training on DNS data. The continuity residual is low everywhere except near the bottom boundary. The momentum residuals show spikes in the plume region above the flame.

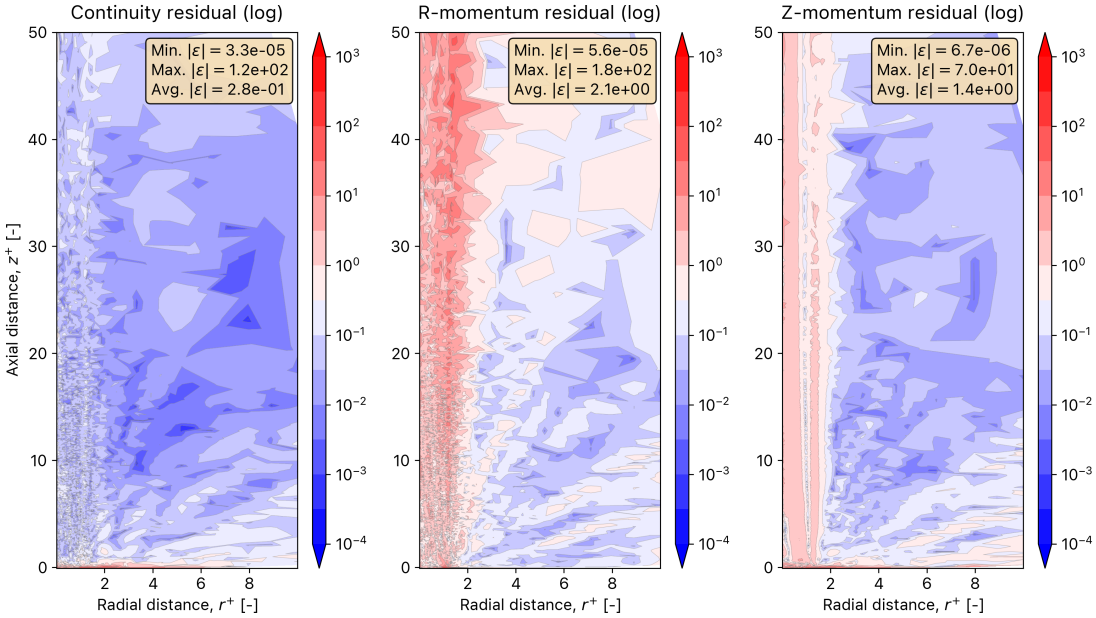


Figure 5.23: Absolute magnitude of continuity and momentum residuals after training on DNS data. The absolute magnitude of the residuals is shown in logarithmic scale. The minimum, maximum, and average absolute values of each residuals is shown on the boxes at the top right of each figure.

To summarize, computing the gradients of flow variables using automatic differentiation leads to small regions with erroneous oscillations, particularly in the plume region above the flame and near the bottom boundary. These oscillations disturb the residual fields, creating regions with large spikes in the absolute magnitude of such residuals. This is particularly true for the residual of the axial momentum equation. Because of the fact that the momentum loss depends on the mean-square error of the residuals, these small regions of localized large magnitude oscillations prevent the momentum loss from decreasing

further. This contributes to the difference in the range of the data and momentum loss terms.

Despite these oscillations, the average absolute value of the residuals is in the order of 10^{-1}, with most of the domain having values well below that in the order of 10^{-2} to 10^{-5}. Furthermore, the momentum loss has been shown to be compatible with the data loss. As the network in trained on the DNS data and the data loss decreases, so does the momentum loss. The momentum loss successfully decreases by 2 order of magnitude during data training. This decrease is not a direct consequence of the training of the network, since the momentum loss is not part of the loss function minimized by the optimizer. Instead, it is a consequence of the improvements in the PINN estimations of the flow fields. As such, it is concluded that the momentum loss has been correctly implemented and that its formulation is consistent with the DNS data.

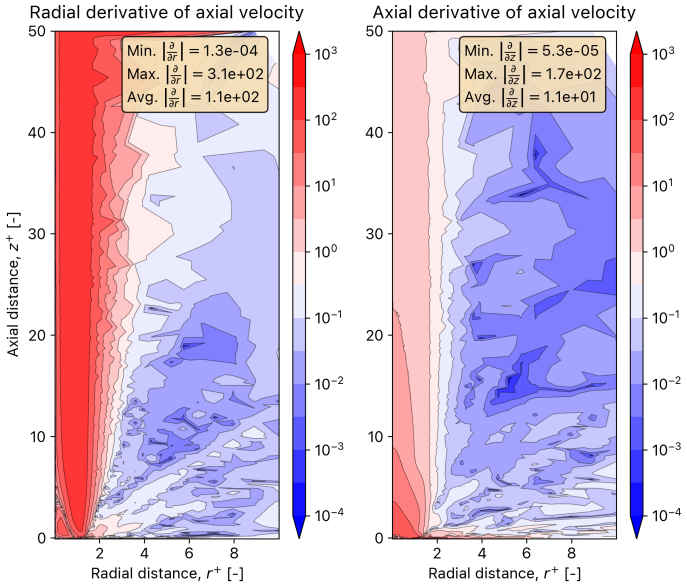


Figure 5.24: Absolute magnitude of axial velocity gradients after training on DNS data.

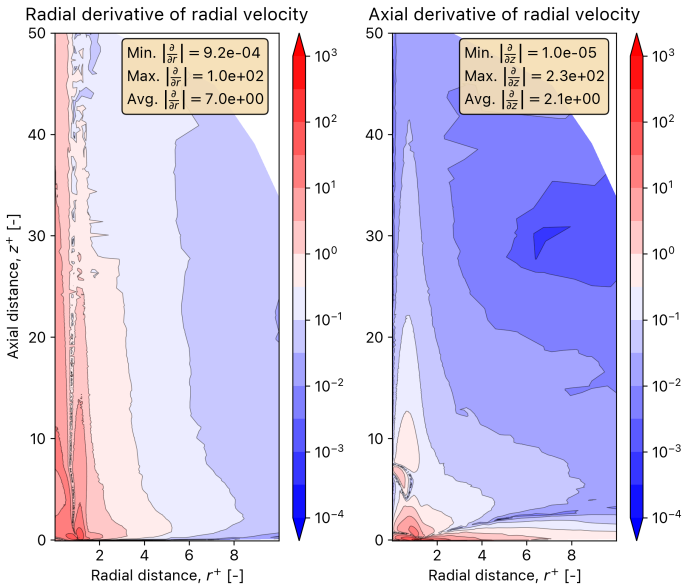


Figure 5.25: Absolute magnitude of radial velocity gradients after training on DNS data.

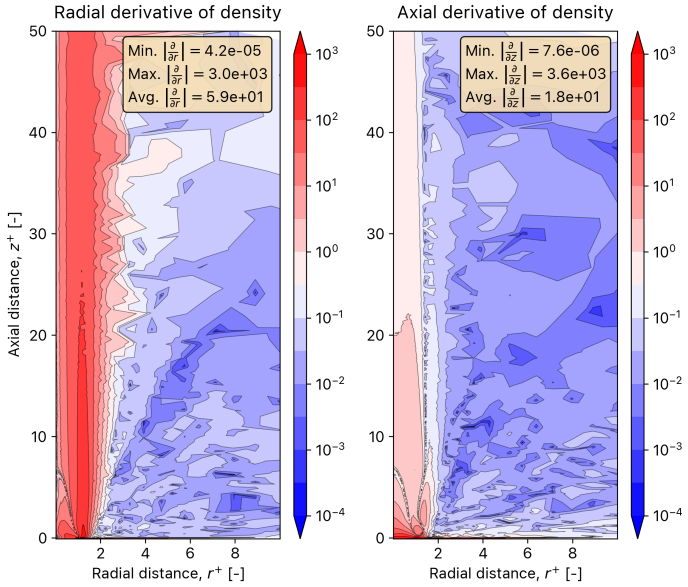


Figure 5.26: Absolute magnitude of density gradients after training on DNS data.

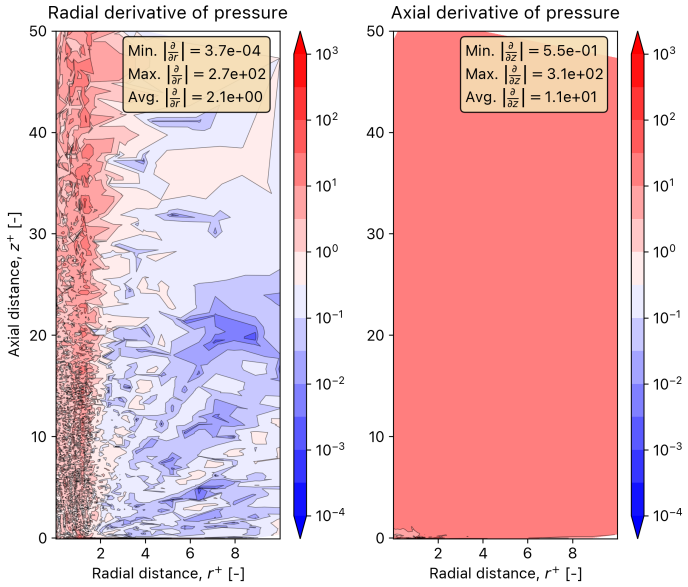


Figure 5.27: Absolute magnitude of pressure gradients after training on DNS data.

5.4.3. Adding the mixture fraction

The addition of the mixture fraction to the reconstruction algorithm requires the construction of a second physics-informed loss term that can enforce the corresponding governing law. As discussed in Chapter 2 the mixture fraction, denoted by ξ, is a scalar quantity governed by the following transport equation [114]:

Mixture fraction transport equation

$$\frac{\partial(\rho\xi)}{\partial t} + \nabla \cdot (\rho \boldsymbol{V} \xi) = \nabla \cdot (\rho \mathcal{D} \nabla \xi) \tag{5.36}$$

where $\mathcal{D}$ is the mass diffusivity. Equation 5.36 must be implemented in the PINN in order to ensure compliance with the passive transport of the mixture fraction through the domain. Following the procedure of Sitte and Doan [26] the mass diffusivity is calculated under the assumption of unity Lewis number and constant Prandtl number (and hence constant Schmidt number):

$$\mathrm{Le} = \frac{\alpha}{\mathcal{D}} = 1 \tag{5.37}$$

$$\mathrm{Pr} = \frac{\mu}{\rho\alpha} = 0.7 \tag{5.38}$$

$$\mathcal{D} = \frac{\mu}{0.7\rho} \tag{5.39}$$

where α is the thermal diffusivity and μ is the dynamic viscosity, which is computed as a function of temperature using Sutherland's law, as done for the momentum loss. The same assumptions made to derive the momentum loss (summarized in Table 5.2) are used here. The residual of the mixture fraction transport equation in steady state can be formulated as follows:

$$\mathcal{E}_{\text{mixture}} = \nabla \cdot (\rho \boldsymbol{V} \xi) - \nabla \cdot (\rho \mathcal{D} \nabla \xi) \tag{5.40}$$

Applying the rules for the divergence operator yields:

$$\mathcal{E}_{\text{mixture}} = \rho\xi(\nabla \cdot \boldsymbol{V}) + \nabla(\rho\xi) \cdot \boldsymbol{V} - \nabla(\rho\mathcal{D}) \cdot \nabla\xi - \rho\mathcal{D}\,(\nabla \cdot \nabla\xi) \tag{5.41}$$

Expanding the terms yields:

$$\begin{aligned} \mathcal{E}_{\text{mixture}} = \; & \rho\xi\left[\frac{1}{r}v + \frac{\partial v}{\partial r} + \frac{\partial u}{\partial z}\right] + \frac{\partial(\rho\xi)}{\partial r}v + \frac{\partial(\rho\xi)}{\partial z}u + \\ & - \rho\mathcal{D}\left(\frac{1}{r}\frac{\partial\xi}{\partial r} + \frac{\partial^2\xi}{\partial r^2} + \frac{\partial^2\xi}{\partial z^2}\right) - \frac{\partial(\rho\mathcal{D})}{\partial r}\frac{\partial\xi}{\partial r} - \frac{\partial(\rho\mathcal{D})}{\partial z}\frac{\partial\xi}{\partial z} \end{aligned} \tag{5.42}$$

The mixture fraction loss can then be computed by taking the MSE of the residual given by Equation 5.42. Hence, to summarize, the mixture fraction loss is computed as follows:

Formulation of the mixture loss

Residual:

$$\begin{aligned} \mathcal{E}_{\text{mixture}} = \; & \rho\xi\left[\frac{1}{r}v + \frac{\partial v}{\partial r} + \frac{\partial u}{\partial z}\right] + \frac{\partial(\rho\xi)}{\partial r}v + \frac{\partial(\rho\xi)}{\partial z}u + \\ & - \rho\mathcal{D}\left(\frac{1}{r}\frac{\partial\xi}{\partial r} + \frac{\partial^2\xi}{\partial r^2} + \frac{\partial^2\xi}{\partial z^2}\right) - \frac{\partial(\rho\mathcal{D})}{\partial r}\frac{\partial\xi}{\partial r} - \frac{\partial(\rho\mathcal{D})}{\partial z}\frac{\partial\xi}{\partial z} \end{aligned} \tag{5.43}$$

Mixture loss:

$$\mathcal{L}_{mixture} = \mathrm{MSE}(\mathcal{E}_{mixture}) \tag{5.44}$$

5.4.4. Verification of the mixture loss

The formulation of the mixture loss is verified following a similar procedure as the one used for the momentum loss, discussed in Section 5.4.2. A neural network with the settings shown in Table 5.5 is trained using all of the DNS data, including data on the mixture fraction. During the training process the mixture loss is evaluated and its value stored, but it is not included in the loss function of the neural network, which is only affected by the data loss.

Table 5.5: Simulation settings for the verification of the mixture loss formulation

Parameter	**Value**
Input variables	(r, z)
Ouput variables	(u, v, ρ, p, T, ξ)
Number of hidden layers	15
Number of neurons per layer	90
Activation function	Sigmoid linear unit (SiLU)
Weight normalization	Yes
Training duration	$100,000$ iterations ($\approx$ 5,000 epochs)
Batch size	10^4
Optimizer	Adam
Learning rate	10^{-3}

The evolution of the loss components during data training is shown in Figure 5.28. As expected, the data loss steadily decreases as the training progresses. Both loss components show oscillations — this is attributed to the use of batch sampling during the training process. During the initial phase of training it can be seen that the mixture loss decreases steadily alongside the data loss. Then, after around 1,000 epochs, the mixture fraction stabilizes while the data loss continues to decrease. Recall that the network is only training on the data loss. Thus, it is expected for this loss component to decrease steadily as the training progresses. On the other hand the mixture loss is not included in the network's loss function and is not directly affected by the optimization process that trains the neural network. Its evolution is only a consequence of the changes in the estimated flow fields.

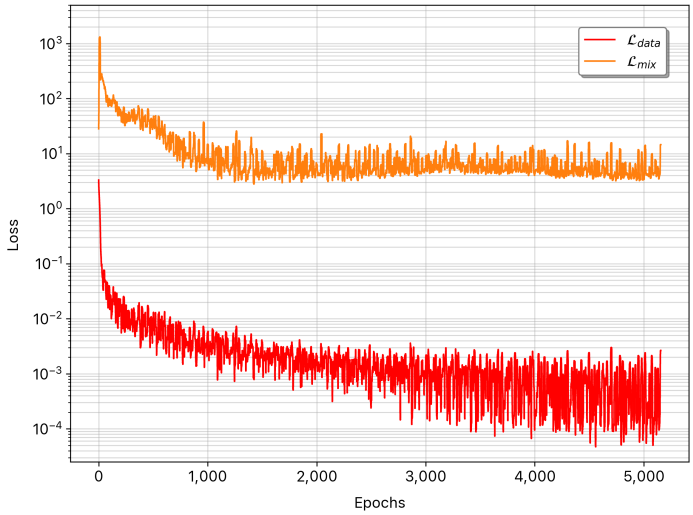


Figure 5.28: Verification of the mixture loss formulation. A PINN is trained using a data loss based on the DNS data, including the mixture fraction. The mixture fraction loss is evaluated during training but not included in the network's loss function.

The estimated fields improve very quickly. Starting from random initial guesses at the beginning of training, within the first 1,000 epochs the fields match the DNS data almost exactly (see Figure 5.29).

The mixture loss first increases very rapidly as the fields change from random and approximately constant values to something that starts to resemble the true flow fields but does not yet satisfy the governing equations due to significant misrepresentations of structures in the flow. As the estimates improve, the mixture loss quickly decreases in parallel with the data loss. The mixture loss decreases by two orders of magnitude within the first 1,000 training epochs as the flow fields are corrected by the data loss.

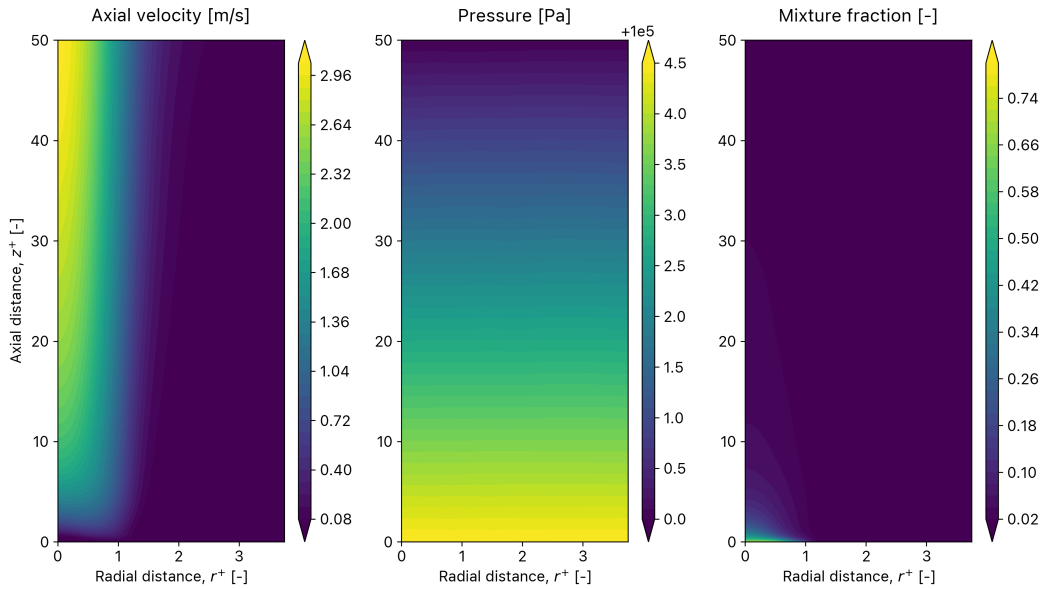


Figure 5.29: Flow field estimates after 20,000 iterations of mixture loss verification. They fields appear identical to the DNS data, shown in Figure 5.3 and Figure 5.4. Only axial velocity, pressure, and mixture fraction fields are shown, but the same can be observed for the remaining quantities.

The reason why the mixture loss become stagnant after around 1,000 epochs is attributed to the fact that, beyond that point, the improvements on the flow fields are almost negligible. The flow fields remain constant throughout the remainder of the training process. No visible large scale changes can be observed in the structures present in any of the six measured flow fields. During this second stage of training the network is making small modifications to decrease the errors at the provided data points, which ensures that the data loss continues to decrease. Nevertheless, these small changes do not significantly affect the flow field estimations in any visible way. The lack of any meaningful changes in the flow fields prevents the mixture fraction from decreasing any further. The fact that the mixture loss decreases significantly and in parallel with the data loss during the initial phase of training is deemed sufficient to verify its formulation. This demonstrates that the mixture loss is compatible with the DNS data. As the estimates of the velocity, density, and mixture fraction improve, the mixture loss decreases steadily, indicating that it has been formulated correctly.

5.5. Modeling the flame structure

At this point we have derived a formulation for the governing equation of the mixture fraction (its transport equation), verified it, and implemented it within the reconstruction framework. To make the most out of the addition of the mixture fraction, we now wish to describe the structure of the flame in mixture fraction space. By doing so we can exploit the mixture fraction data in the reconstructions to obtain information about other thermochemical quantities like temperature, composition, and density.

This was already discussed in Chapter 2, where we showed that the mixture fraction allows us to decompose diffusion flame computations into two separate and independent problems: a mixing problem and a flame structure problem. The former is concerned with the evolution of the mixture fraction field as a function of the spatial coordinates (and time, for unsteady problems). It is a fluid dynamics problem that depends on the geometry of the domain, the boundary and initial conditions, and the behavior of the flow. The latter is a chemical problem where we establish relations for the temperature and composition (i.e. mass fractions of all species) as a function of the mixture fraction.

Thanks to the mixture fraction these two problems can be decoupled and solved independently. In our case, the mixing problem is taken care of by the PINN algorithm — we either known the spatial distribution of the mixture fraction in the domain *a priori* (when the mixture fraction is used as a data source) or alternatively we obtain it as an outcome of the reconstruction. However, we have yet to establish how to deal with the flame structure problem. That is the purpose of this section. The idea is to precompute a representative structure for our flame to create enhanced initializations for the missing flow fields when searching for minimal data solutions (done in Section 6.5), and also for the computation of the HRR, which requires that we estimate the enthalpy of the flow from the mixture fraction (discussed in Section 5.6 and Section 6.6).

Our goal in this section is to find relations of the form $T(\xi)$ and $Y_k(\xi)$ for all species k. To do so we must make assumptions regarding the reversibility and speed of the chemical process. In terms of its speed, a chemical reaction can be assumed to take place at a finite rate or infinitely fast. The latter assumption is only applicable when the characteristic chemical time is much smaller than all other characteristic times of the flow (diffusion and flow time scales). This is the case in flows with a high Damköhler number, generally acceptable in fires.

In terms of its reversibility, a chemical reaction can be said to be reversible or irreversible. Additionally, a case of interest is that of *frozen* chemistry, which refers to a situation in which the oxidizer and fuel mix but do not react. The chemistry is then said to be frozen because no chemical reactions take place. As discussed in Section 5.2, the DNS simulations assume irreversible fixed rate chemistry, where the reaction rate depends on temperature and is given by the Arrhenius law. Here we take a simpler approach to model the flame structure. We assume infinitely fast chemistry, noting that the flame is laminar and has a high Damköhler number. The combination of irreversibility with infinitely fast chemistry is the so-called Burke-Schumann approximation [8].

As a result of the Burke-Schumann approximation fuel and oxidizer cannot coexist anywhere in the domain, with the exception of the flame front, where they mix at stoichiometric conditions and combust infinitely fast, creating reaction products. The reason for this is straightforward. If fuel and oxidizer were to coexist somewhere in the domain at non-stoichiometric conditions, they would react immediately. This reaction would consume fuel and oxidizer at stoichiometric proportions until only one of them is left. This means that the flame front, where the mixture fraction is equal to its stoichiometric value ($\xi = \xi_{st}$), separates the domain into a rich region inside of the flame (which cannot contain any oxidizer) and a lean region outside of the flame (which cannot contain any fuel). This simplification is very useful to define the flame structure in mixture fraction space.

To define the flame structure we first compute the stoichiometric value of the mixture fraction. To do so, we compute the equivalence ratio as:

$$\phi = s\frac{Y_F^0}{Y_O^O} \tag{5.45}$$

where Y_F^0 and Y_0^0 are the initial mass fractions of fuel and oxidizer in their respective streams, and s is the mass stoichiometric ratio, which for a single reaction of a C_mH_n hydrocarbon with the oxygen in air is given by [115]:

$$s = \frac{32\left(m + \frac{n}{4}\right)}{12m + n} \tag{5.46}$$

Note that fuel and oxidizer come into the domain through independent streams. Hence, the equivalence ratio given in Equation 5.45 represents the value that would be obtained if the same amount of mass from the fuel and oxidizer streams was mixed. It therefore does not give any information about the local state of mixing, but it characterizes the local structure of the flame that forms when the two streams interact. The initial mass fraction of fuel is $Y_F^0 = 1$ since we assume that the fuel is not diluted, that is, that the fuel stream does not contain any oxidizer or other species. The initial mass fraction of oxidizer in the oxidizer stream is $Y_O^0 = 0.233$, the mass fraction of oxygen in air at standard conditions, with nitrogen making up the remaining mass $\left(Y_{N_2}^0 = 0.767\right)$. The stoichiometric value of the mixture fraction is then given by:

$$\xi_{st} = \frac{1}{1 + \phi} \approx 0.0622 \tag{5.47}$$

Let us now compute the composition of the flame in mixture fraction space, that is, to find relations of the form $Y_k(\xi)$ for all species k. The species that are present in the flame are oxygen (O_2), n-heptane (C_7H_{16}), the reaction products (CO_2 and H_2O), and nitrogen (N_2), which is inert and does not take part in the chemical reactions. The mass fractions of fuel and oxidizer are computed first. This is done by drawing straight mixing lines in mixture fraction space connecting the three points at which the mass fractions are known. These are a mixture fraction of zero (where $Y_F = 0$ and $Y_O = Y_O^0$), a mixture fraction of one (where $Y_F = Y_F^0$ and $Y_O = 0$), and the stoichiometric mixture fraction (where $Y_F = Y_O = 0$). Additionally, from the theory of the Burke-Schumann approximation we can also derive the temperature profile as a function of the mixture fraction [8]. The resulting relations are:

$$\textbf{On the oxidizer side } (\xi < \xi_{st}): \begin{cases} Y_F = 0 \\ Y_O = Y_O^0\left(1 - \frac{\xi}{\xi_{st}}\right) \\ T = \xi T_F^0 + (1 - \xi)T_O^0 + \frac{QY_F^0}{C_p}\xi \end{cases} \tag{5.48}$$

$$\textbf{On the fuel side } (\xi > \xi_{st}): \quad \begin{cases} Y_F = Y_F^0 \frac{\xi - \xi_{st}}{1 - \xi_{st}} \\ Y_O = 0 \\ T = \xi T_F^0 + (1 - \xi)T_O^0 + \frac{QY_F^0}{C_p}\xi_{st}\frac{1-\xi}{1-\xi_{st}} \end{cases} \tag{5.49}$$

where T_F^0 is the initial temperature of the fuel, which is assumed to be the boiling temperature of n-heptane, $T_B = 371.5$ K. The initial temperature of the oxidizer is the ambient temperature $T_O^0 = T_0 = 300$ K. Additionally, Q is the heating value of the fuel and C_p the mass specific isobaric heat capacity of the mixture. The latter is a function of temperature and composition. It is computed as:

$$C_{p,\text{mix}} = \sum_k C_{p,k}Y_k = \sum_k C_{p,k}^m \frac{Y_k}{W_k} = f(T, Y_k) \tag{5.50}$$

where W_k is the molar mass of species k, $C_{p,k}$ is the isobaric mass heat capacity, and $C_{p,k}^m$ the isobaric molar heat capacity. The latter two are are functions of temperature. NASA polynomials are used to compute the variation of $C_{p,k}^m$ as a function of temperature for all species, using the approach of Sitte and Doan [26]. The mixture fraction is used to determine the composition of the flame, and the resulting mass fractions for each species are used to compute the specific heat capacity of the mixture using Equation 5.50.

The variation of temperature, fuel, and oxidizer mass fraction as a function of mixture fraction is plotted against the reference DNS data to check how far away the data is from the Burke-Schumann solution (Figure 5.30). Note that here the specific heat capacity is assumed to be constant, resulting in a linear relation for $T(\xi)$. It can be seen that generally the DNS data is not far off from the Burke-Schumann approximation. The reason for this is that the pool fire flame is a laminar flame with high Damköhler number, which justifies the assumption of infinitely fast chemistry. The deviation in

the rich branch (high ξ) of the temperature plot are caused by the assumption of constant C_p. Note that the constant C_p assumption is only used when constructing initializations for the missing flow fields in the search for minimal data solutions (see Section 6.5), while the actual specific heat capacity of the mixture $C_{p,\text{mix}}$ is calculated using Equation 5.50 to compute the HRR (see Equation 5.6 and 6.6).

The remaining deviations from the Burke-Schumann solution are caused by the finite rate chemistry used in the DNS simulations. For a given value of the mixture fraction different values of the thermochemical properties are possible due to the finite reaction rate. The higher the reaction rate, the closer the flow properties get to the infinitely fast chemistry limit.

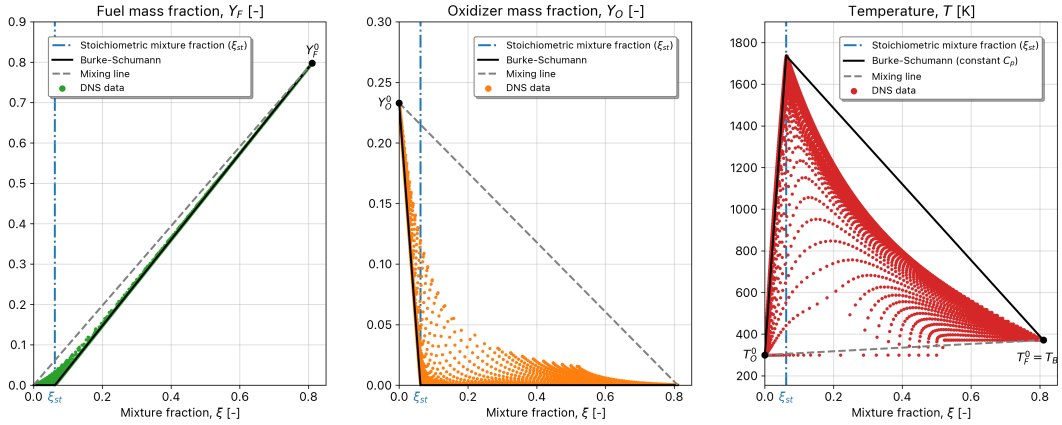


Figure 5.30: Comparison of DNS data against the Burke-Schumann approximation. Fuel mass fraction (left), oxidizer mass fraction (middle) and temperature data (right) is shown. T_{ad} is the adiabatic flame temperature, reached at stoichiometric conditions. The dashed-dotted blue lines indicate the stoichiometric value of the mixture fraction. The dashed gray lines show the pure mixing solution — they represent the flame structure under the assumption of frozen chemistry. The solid black lines show the Burke-Schumann solution.

To finalize the modeling of the flame structure we have yet to compute the mass fractions of the remaining species, namely oxygen, carbon dioxide, and water vapor. These are needed to calculate the specific heat capacity of the mixture in Equation 5.50, and hence the temperature. Additionally they are also required to calculate the enthalpy of the flow, which is needed to estimate the HRR of the flame. These mass fractions are computed using the same approach as those for the fuel and oxidizer: their values are known at the three reference points in mixture fraction space ($\xi = 0$, $\xi = \xi_{st}$, and $\xi = 1$), and straight mixing lines are drawn to connect these points and obtain piecewise linear functions of the form $Y_k(\xi)$ for $k = CO_2$, H_2O and N_2.

The values of the species mass fractions at the three reference points are shown in Table 5.6. The mass fractions of the reaction products at stoichiometric conditions are calculated using the stoichiometric coefficients of the reaction equation. Note that carbon dioxide and water vapor have a mass fraction of zero at both $\xi = 0$ and $\xi = 1$. Nitrogen on the other hand is only equal to zero at $\xi = 1$. At $\xi = 0$ it is present in the air alongside the oxidizer, and therefore assumes a mass fraction of $Y^0_{N_2} = 0.767$.

Table 5.6: Mass fractions of all species at the three reference points in mixture fraction space

Species	**Chemical formula**	**Mass fraction**		
		Oxidizer only ($\xi = 0$)	Stoichiometric conditions ($\xi = \xi_{st}$)	Fuel only ($\xi = 1$)
Heptane	C_7H_{16}	0	0	1
Oxygen	O_2	0.233	0	0
Carbon dioxide	CO_2	0	0.1921	0
Water vapor	H_2O	0	0.0899	0
Nitrogen	N_2	0.767	0.7180	0

Using the data of Table 5.6 the composition of the mixture is known everywhere in mixture fraction

space. In order to verify this computation and to check the validity of our assumptions, the DNS mixture fraction data is used to plot the resulting mass fraction fields of all species computed using the Burke-Schumann flame structure. The resulting fields are compared against those from the DNS simulations. Figure 5.31 shows this comparison for the fuel mass fraction, and Figure 5.32 does the same for the water vapor. The remaining species are not shown for brevity.

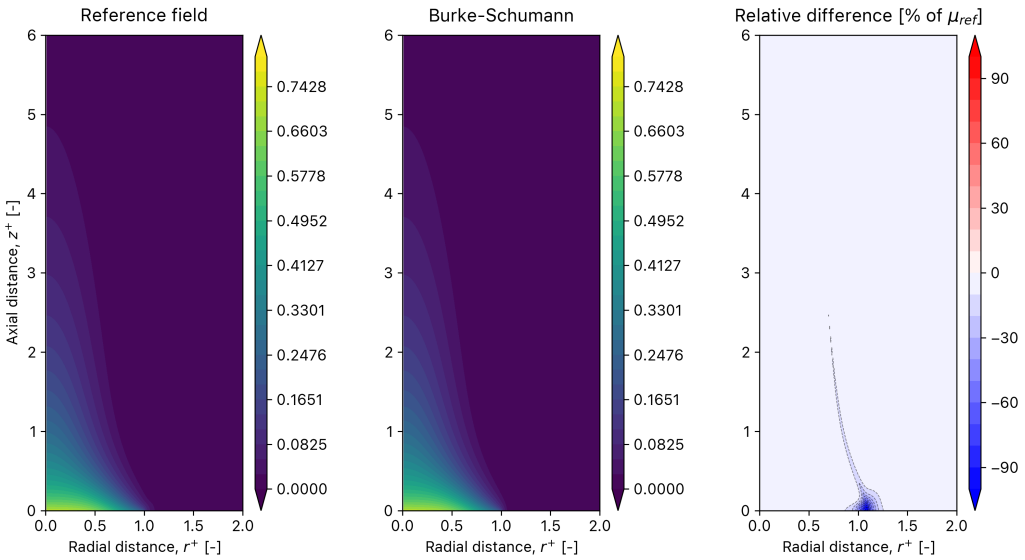


Figure 5.31: Fuel concentration using the Burke-Schumann approximation. Comparison against DNS data. The domain has been cropped to more clearly show the concentration of fuel within the flame.

The fuel concentration matches the DNS data very well. It is high within the flame and equal to zero outside of it. The fuel concentration slowly decreases from the inlet as the gas approaches the flame front and begins to react. The relative difference between the DNS data and the Burke-Schumann approximation (right side of Figure 5.31) shows a small region where errors are visible. This occurs near at the anchoring point near the bottom right side of the flame front. In that region there are discrepancies with respect to the DNS data due to finite-rate effects. The oxidizer concentration, not shown for the sake of brevity, shows the same behavior. It matches the DNS data very well except for the small area near the anchoring point.

The concentration of water vapor shown in Figure 5.32 also shows this behavior. The concentration is zero at the inlet and at the far field, as expected. The concentration of water vapor progressively increases as the gas approaches the flame front and reactions start to take place. The water vapor (alongside carbon dioxide, the other product of the reaction) coexist with the fuel within the flame region. The peak concentration of reaction products occurs at the flame front. The resulting field has a similar structure as the temperature field — this is expected, since the concentration of reaction products is proportional to the reaction rate used in the simulations, which is a function of temperature. The reaction products convect upwards with the plume, where they mix with the ambient air as they travel upwards through the domain. The same erroneous region near the anchoring point in Figure 5.31 can also be observed in Figure 5.32. The concentration of carbon dioxide shows the exact same behavior as that of water vapor. Both species are reaction products and hence are expected to behave in the same way. The computed concentration fields confirm this.

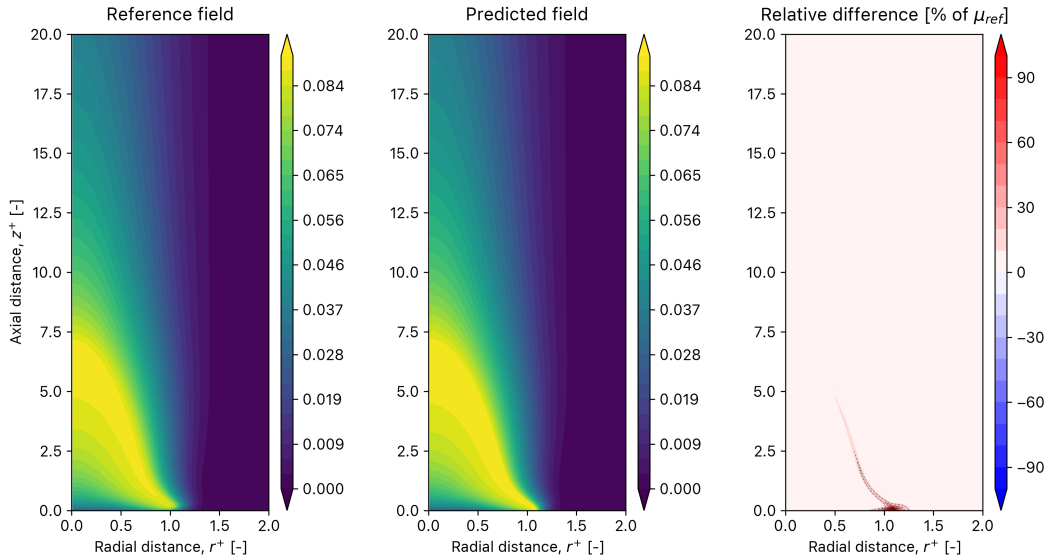


Figure 5.32: H_2O concentration using the Burke-Schumann approximation. Comparison against DNS data. The domain is cropped to depict the change of concentration within the flame and in the plume region.

The data presented until this point confirms that the Burke-Schumann approximation yields an accurate estimation of the flame structure in mixture fraction space. The temperature profile is closely approximated, as are the mass fractions of all species. As a final step to conclude this discussion we wish to use our knowledge of the composition to calculate the specific gas constant R as a function of the mixture fraction. This is used in Section 6.5 to calculate a mixture fraction enhanced initialization of the density field while searching for new minimal data solutions to the reconstruction problem. To obtain a relation of the form $R = f(\xi)$ we apply the same approach as done for the mass fractions: the value of R is known at the three reference points in mixture fraction space, and linear mixing lines are drawn connecting these points to obtain a piecewise linear function. The value of the specific gas constant at the reference points is calculated by dividing the universal gas constant $\mathcal{R}$ by molecular weight of the mixture (M_{gas}), which can be derived from its composition:

$$R = \frac{\mathcal{R}}{M_{\text{gas}}} \tag{5.51}$$

where $\mathcal{R} = 8.314$ J/mol K. Using this procedure, the following three values of the specific gas constant are computed:

$$\text{Oxidizer only:} \quad \begin{cases} \xi = 0 \\ R = 287.05 \text{ J/kg K} \end{cases} \tag{5.52}$$

$$\text{Stoichiometric conditions:} \quad \begin{cases} \xi = \xi_{st} = 0.0622 \\ R = 290.9 \text{ J/kg K} \end{cases} \tag{5.53}$$

$$\text{Fuel only:} \quad \begin{cases} \xi = 1 \\ R = 82.98 \text{ J/kg K} \end{cases} \tag{5.54}$$

Linear mixing lines are drawn connecting these points in mixture fraction space. The resulting piecewise linear function is shown in Figure 5.33.

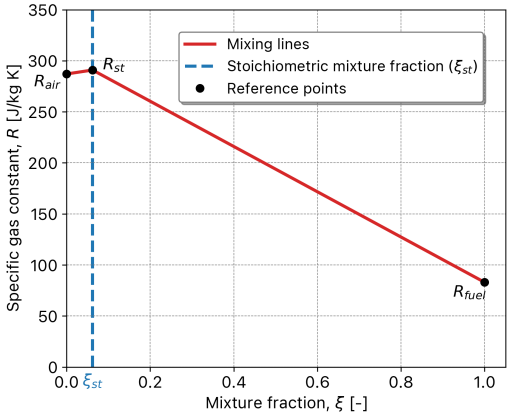


Figure 5.33: Specific gas constant in mixture fraction space. Linear mixing lines are drawn to connect the three known values of R at $\xi = 0$, $\xi = \xi_{st}$, and $\xi = 1$.

5.6. Estimating the HRR

As discussed in Section 5.1 we wish to evaluate the ability of the reconstruction framework to estimate the Heat Release Rate (HRR) of the flame. The HRR is a measure of the rate at which a flame releases energy. Introduced in Chapter 2, the HRR is a key parameter for practical studies of combustion flows since it is a good proxy for the destructive power of a flame. Two flames with an equivalent HRR can be expected to release the same amount of energy to their environment, irrespective of the flame shape or the specific configuration. For this reason the HRR is a parameter of interest, and it is desirable for the reconstructed flow fields to not only accurately reproduce the velocity but also be able to yield a reasonable estimate for the HRR.

We wish to extend the reconstruction method with the ability to estimate the HRR of the flame. Then, the estimated HRR can be used as an additional performance metric to evaluate the quality of the reconstructions. Since the HRR is not a direct output of the reconstructions, it must be computed from the available flow quantities. This section presents the method used to perform this computation.

Calculation of the HRR

The basic idea to calculate the HRR is that it can be computed by performing a macroscopic energy balance on a control volume that encloses the flame. We perform a control volume analysis at constant pressure with no work exchanged. By balancing the enthalpy flowing through the boundaries of the control volume one can estimate the HRR. A schematic representation of the control volume enclosing the flame is shown in Figure 5.34. There are three contributions to the enthalpy balance over the control volume. These are the fuel flowing into the domain through the inlet (bottom boundary), the air entrainment flowing in through the outer boundary (right boundary), and the outflux of exhaust gases (reaction products mixed with air) through the top boundary. Note that there can be no mass flux across the left boundary due to the symmetry boundary condition. At each of these boundaries the local conditions of the flow must be taken into account to accurately compute the enthalpy fluxes in and out of the domain.

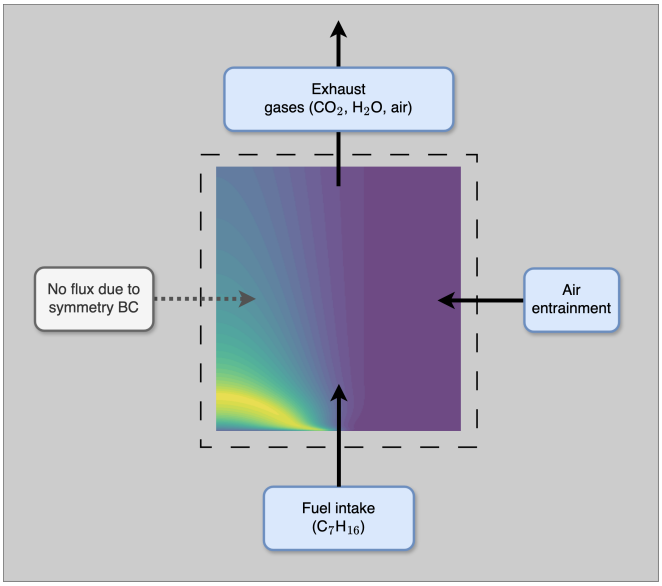


Figure 5.34: Estimating the HRR as a macroscopic energy balance. The three contributions to the balance are highlighted in blue. No flux can enter the control volume through the left boundary due to the symmetry boundary condition (shown in gray).

To perform the enthalpy balance we invoke the first law of thermodynamics, noting that the sum of enthalpy fluxes across the boundaries of the control volume is equal to the heat source $\dot{Q}$ and the technical work $\dot{W}_T$, neglecting kinetic and potential energy:

$$\sum_i h_i \, \dot{m}_i = \dot{Q} + \dot{W}_T \tag{5.55}$$

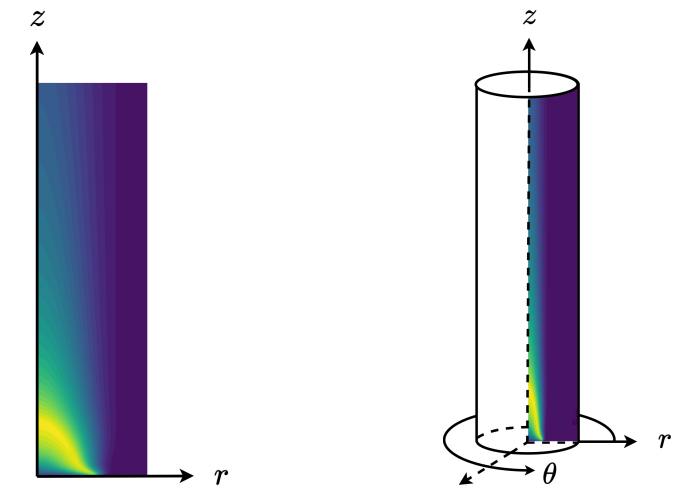
where i is an index over the boundaries, h_i is the specific enthalpy at each boundary, and $\dot{m}_i$ the mass flow through the boundary. In the present case there is no technical work applied to the system, and hence $W_T = 0$. Furthermore, radiation is not explicitly considered, following the approach of Sitte and Doan [26], and hence $\dot{Q} = 0$. Hence, in the present case, the energy balance of the control volume simplifies to::

$$\sum_i h_i \, \dot{m}_i = 0 \tag{5.56}$$

In the present case the properties of the flow vary along the boundaries of the domain. Hence, the enthalpy and mass flow are not constant along the boundary. For this reason, we evaluate the enthalpy flux through each boundary by integrating the product of the local mass flow normal to the boundary with the specific enthalpy over the corresponding area:

$$\text{enthalpy flux} = \dot{H} = \iint_{\boldsymbol{S}} h \, \dot{\boldsymbol{m}} \cdot d\boldsymbol{S} \tag{5.57}$$

Note that here the mass flow $\dot{\boldsymbol{m}}$ is a vector. Additionally, although the spatial domain used in the DNS simulation and the PINN reconstructions is two-dimensional, the computation of the HRR must take into account the rotational symmetry of the flow and calculate the HRR of the entire three-dimensional flame. A schematic representation of how the two-dimensional domain is extended to account for the rotational symmetry of the flow is shown in Figure 5.35.



(a) 2D view of the domain **(b)** 3D view of the domain

Figure 5.35: Visualizing the domain in two and three dimensions. The HRR computation takes into account the rotational symmetry of the flow.

The mass flow can be directly obtained from the outputs of the reconstruction by multiplying the density with the velocity vector:

$$\dot{\boldsymbol{m}} = \rho \mathbf{V} \tag{5.58}$$

The specific enthalpy h is the sum of the enthalpy of formation Δh_f and the sensible enthalpy h_s:

$$h = \Delta h_f + h_s \tag{5.59}$$

Each of these terms is calculated as a weighted average of the contributions of each species depending on the local composition, which is a function of the mixture fraction. The enthalpy of formation of the mixture is therefore given by:

$$\Delta h_f = \sum_k \Delta h_{f,k} \frac{Y_k}{W_k} \tag{5.60}$$

where $\Delta h_{f,k}$ is the molar based enthalpy of formation of species k. The same is is done for the sensible enthalpy:

$$h_s = \sum_k h_{s,k} \frac{Y_k}{W_k} \tag{5.61}$$

The molar based sensible enthalpy $h_{s,k}$ is calculated for each species by integrating its molar specific heat capacities as a function of temperature:

$$h_{s,k} = \int_{T_{\text{ref}}}^{T} C_{p,k}(T)\, dT \tag{5.62}$$

where $T_{\text{ref}} = 298.15$ K is the reference temperature at which the formation enthalpy of the species are measured, and T is the temperature of the mixture at the point where the enthalpy is evaluated. The molar specific heat capacities and the molar enthalpy of each species k can be computed as a function of temperature using NASA polynomials:

$$C_{p,k} = \mathcal{R} \cdot \left(a_1 + a_2 T + a_3 T^2 + a_4 T^3 + a_5 T^4\right) \tag{5.63}$$

$$h_k = \mathcal{R} \cdot \left(a_1 T + \frac{a_2}{2} T^2 + \frac{a_3}{3} T^3 + \frac{a_4}{4} T^4 + \frac{a_5}{5} T^5 + a_6\right) \tag{5.64}$$

where $\mathcal{R}$ is the universal gas constant. Energy is conserved during the chemical reaction. The combustion process within the control volume rearranges the chemical bonds in the reactants to form the products, transforming formation enthalpy into sensible energy. It follows that the HRR can be computed from the difference of the sensible enthalpy in the incoming and outgoing fluxes. Summing all fluxes using the convention of positive outflux yields:

$$\text{HRR} = \sum_i \dot{H}_{s,i} \tag{5.65}$$

Verification and baseline HRR estimation

The reference DNS data is used to verify aforementioned method to estimate the HRR based on the energy balance over the control volume enclosing the flame. The DNS simulation contains, as one of its outputs, the HRR field, with units W/m^3. This field is integrated numerically, taking into account the rotational symmetry of the flow, to verify the present method. Additionally, by applying the control volume approach to the DNS data, a baseline for the reconstructed HRR is established. This baseline is used in Section 6.6 to evaluate the ability of the PINN to reconstruct the HRR from reconstructed quantities.

If the present method is implemented correctly, one would expect a constant estimate of the HRR independently of the size of the control volume, provided that it completely encompasses the flame. To test this, the present method is applied multiple times changing the location of the top and outer (right) boundaries of the domain. The bottom boundary is kept constant at $z^+ = 0$, and the left boundary is kept constant at the symmetry axis $r^+ = 0$. The resulting estimates for the HRR, alongside that obtained by direct integration of the DNS HRR data, are shown in Table 5.7.

Table 5.7: Verification of the HRR estimation by comparison with the DNS data. The present method agrees with the DNS data provided the control volume completely encloses the flame (highlighted in gray).

Method	Outer boundary	Top boundary	Enclosed flame?	Estimated HRR [W]	Error [%]
Volume integration from DNS data	–	–	–	**102.7**	–
Energy balance over control volume	$r^+ = 1$	$z^+ = 1$	No	42.8	-58.29
Energy balance over control volume	$r^+ = 2$	$z^+ = 2$	No	57.9	-43.62
Energy balance over control volume	$r^+ = 2$	$z^+ = 5$	Partially	90.6	-11.76
Energy balance over control volume	$r^+ \geq 3$	$z^+ \geq 10$	Yes	96.2	-6.33

The HRR estimate of the present method improves as the control volume grows, remaining constant past the point where the volume encloses the flame. This is indicative of the fact that the method has been implemented correctly and behaves as expected. Provided that the control volume completely encloses the flame, the present method is in agreement with direct integration of the DNS HRR data, yielding an HRR estimate within $\sim 6\%$ of that of the DNS (last row of Table 5.7). This is deemed sufficient, and the present method is therefore considered to be an a suitable one to estimate the HRR of the flame and to be verified by the DNS data.

6

Results & Analysis

The developed PINN reconstruction framework is used to perform a wide range of reconstruction attempts to tackle the research objectives of this thesis. This chapter presents and discusses the outcome of these reconstructions. The chapter is structured such that each section tackles a particular aspect of the research questions introduced in Chapter 1. Along the way, the outcomes of the reconstruction attempts are analyzed and conclusions are drawn. Chapter 7 then summarizes and integrates these conclusions to answer the research questions of this thesis.

Outline of the results

Before analyzing the performance of the extended PINN model, adapted with the inclusion of the mixture fraction, a set of standard momentum-based reconstructions are performed. Through these reconstructions the newly developed PINN reconstruction framework introduced in Chapter 5 is verified. A series of tests are performed to analyze the sensitivity and robustness of the reconstruction algorithm. These preliminary tests are discussed in Section 6.1. Beyond verifying that the reconstruction algorithm works as intended, they serve a dual purpose. First, they provide insight into the limits of applicability of PINNs for the reconstruction of pool fire flow fields. They show the effect of various parameters on the reconstruction accuracy and the performance of the network during training. This paints a more complete picture of the strengths and limitations of PINNs as a flow field reconstruction method for pool fire flames. Previous work lacks such an analysis, and therefore this is a novel contribution of the present work.

In addition, the tests presented in Section 6.1 provide insights into the optimal combination of hyperparameters and simulation settings to perform successful reconstructions. It should be noted that the focus of this thesis lies on the extension of the PINN reconstruction algorithm through the implementation of additional physics, i.e. the mixture fraction. As such, the analysis on the optimal simulation settings is limited. The basic setup for the PINNs is derived from previous work [26, 93], and it is beyond the scope of the present work to consider the effect of making significant modifications to the reconstruction algorithm in terms of the neural network design. Nevertheless, within the constraints imposed by the chosen PINN model, there is still room to perform a sensitivity analysis on the network hyperparameters and the simulation settings. The results of this analysis are discussed in Section 6.1.

Using the optimal simulation setup found in Section 6.1, a baseline momentum-based reconstruction is presented in Section 6.2. These results represent the best velocity reconstruction that has been obtained using the standard version of the PINN algorithm, prior to its extension with the mixture fraction. They serve as a benchmark to evaluate the effects of the mixture fraction as an additional physics-informed constraint in the training process.

Following the momentum-based reconstructions of Section 6.1 and Section 6.2, a wide range of tests are conducted to evaluate the effect of the mixture fraction. First, Section 6.3 presents a series of tests where the mixture fraction is used an additional reconstruction target alongside the two velocity components. These tests verify that the extended PINN model works as intended and assess its ability to reconstruct the mixture fraction field. Additionally, the effect on the accuracy of the velocity

reconstructions is also discussed to determine if the extended PINN model is still capable of obtaining adequate velocity reconstruction despite the increase in model complexity.

Then, Section 6.4 discusses the effect of the mixture fraction when used as a data source instead of a reconstruction target. It is desirable to understand if using a passive scalar like the mixture fraction as a data source for the reconstructions has a significant effect on the reconstruction accuracy that can be obtained. This could open up new avenues for future research where experimental measurements on the mixture fraction can be used as an alternative to other flow quantities in order to perform velocity reconstructions.

This analysis is then taken a step further in Section 6.5 by considering the role of the mixture fraction in the search for minimal data solutions to the reconstruction problem. Minimal data solutions are defined as successful reconstructions that rely on a reduced amount of flow variables as an input to the reconstruction algorithm. From a practical standpoint it is desirable to reduce the amount of flow variables that are required to obtain successful velocity reconstructions since some of this data, specifically that of density and pressure, is generally difficult to obtain experimentally. Previous work has identified the advantages of initializing the reconstruction process with appropriate guesses for the missing data fields in the search for minimal data solutions [26, 93]. The performance of PINN reconstruction algorithms has been shown to improve when these guesses are used to initialized the training process [93].

The present work explores the use of the mixture fraction as a tool to improve the initialization of the network and, in doing so, search for new minimal data solutions. The mixture fraction is used to construct improved guesses to initialize the network by exploiting its relation to other flow variables such as temperature, as discussed in Section 5.5. This improved initialization is combined with the fact that, thanks to the presence of the mixture fraction as a training variable, the extended PINN algorithm can use the mixture loss as an additional way to constraint the reconstruction of the velocity fields. This is the focus of Section 6.5 — exploring the new avenues that open up in the search for minimal data solutions thanks to the inclusion of the mixture fraction in the reconstruction algorithm.

Finally, Section 6.6 assesses the efficacy of the extended PINN model at reconstructing the Heat Release Rate (HRR) of the flame using the method presented in Section 5.6. As discussed previously, the HRR is a good proxy for the destructive power of a fire, and therefore it is key parameter of interest in the study of pool fires and other reacting flows. For that reason, it is desirable to obtain reconstructions that not only are capable of accurately reproducing the velocity fields but also yield accurate estimates of the HRR.

Procedure for analysis

Many reconstruction attempts are discussed in the following sections. The same approach is applied to analyze all test results to draw conclusions in a consistent way. The analysis generally begins by inspecting the performance of the network during training. The evolution of the loss components and the reconstruction errors (standardized MSEs) is visualized to get a general sense of the quality of the reconstruction. This is followed by a visual inspection of the reconstructed flow fields to qualitatively understand the results of the reconstructions. Finally, key performance metrics (defined in Section 5.1.1) are evaluated to obtain a quantitative assessment of the reconstruction accuracy.

All reconstruction attempts make use of an initial pre-training phase where the PINN is trained only on the data loss, the composition of which varies between reconstructions. During the pre-training phase the network learns to reproduce the training data before the introduction of the physics-informed loss terms. The performance of the network during this phase is often omitted for the sake of brevity and due to the fact that it rarely shows any interesting features. The PINN is almost always capable of minimizing the data loss and accurately reproducing the training data. The focus is therefore placed on the performance of the network and the evolution of the reconstructed flow fields during physics-informed training. The training behavior of the network during this phase (i.e. the evolution of its loss components and the reconstruction errors) is generally similar between runs. For this reason, only the preliminary tests of Section 6.1 and the baseline reconstruction of Section 6.2 show this data. For the remaining tests, the training performance is omitted unless is shows significant deviations from the baseline.

6.1. Preliminary tests

A series of preliminary tests have been conducted before attempting full reconstructions or analyzing the effect of the mixture fraction. These preliminary tests, discussed in this section, serve three main purposes. Firstly, they are used to verify that the reconstruction algorithm is working as intended. As discussed in Chapter 5 the PINN-based reconstruction framework has been developed from scratch, and therefore it is important to test that the algorithm and the surrounding data processing pipeline is working as intended before conducting complex reconstructions.

The second purpose of these tests is to assess the sensitivity and robustness of the reconstruction method to different settings and network hyperparameters. This has not been done in previous work attempting velocity reconstructions in pool fire flames using PINNs. Generally, a specific network configuration that has been shown to work is reported, but no information is available on the effect of hyperparameters on the network performance. This is not ideal, since the optimal configuration is generally problem specific [1]. The tests described in this section try to address this and provide a more comprehensive overview of the behavior of the reconstruction algorithm. In doing so, several failures modes are identified. These affect the behavior of the network during training, and understanding them is important in order to perform successful reconstructions.

The third and final objective of these tests is to identify the most promising settings to perform the reconstructions. The optimal settings are used in Section 6.2 to conduct a baseline momentum reconstruction that serves as a reference point to evaluate the effect of the mixture fraction in the reconstruction process.

The setup used to perform these preliminary tests, which serves as the starting point for the present work, is taken from previous work [26, 93]. This includes the type of neural network to construct the PINN, the architecture of the network and the choice of optimizer, among others parameters. The standard settings, which are kept constant throughout all of the reconstructions attempted in this thesis, are summarized in Table 6.1. This setup has been proven to work for the problem under consideration in previous work, motivating its use in this thesis. Evidently, the parameters shown in Table 6.1 are design choices that can affect the quality of the reconstructions. It falls outside of the scope of the present work to perform a comprehensive analysis on the effect of significantly modifying the basic setup shown in Table 6.1. Instead, as discussed in Chapter 1, the focus of this thesis is placed on improving the reconstruction accuracy by embedding additional physics into the PINN through the mixture fraction. Nonetheless, within the constraints of the chosen network configuration there is room to analyze the effect of various hyperparameters and settings in order to obtain the best performing setup to perform the reconstructions.

Table 6.1: Chosen model configuration

Parameter	Selection
Network type	Standard PINN, multi-layer perceptron (MLP)
Network architecture	20 hidden layers of 15 neurons per output variable
Activation function	Sigmoid linear unit (SiLU)
Weight normalization	Yes
Weight initialization	Default `Pytorch` initialization
Sampling of collocation points	Random, with a fixed batch size
Sampling of measurement points	Random ($\neq$ collocation points), with a fixed batch sized
Loss function norm	L2 (mean squared error)
Optimizer	Adam

In particular, the effect of three different parameters has been investigated. These are the initial learning rate of the Adam optimizer, the weight of the data loss during momentum training, and the batch size. Additionally, the effect of adding velocity boundary conditions during training has also been explored. All tests presented in this section make use of a PINN model that has been pre-trained on the standard training variables, namely pressure, density, and temperature. After data training, the PINN

is put through a physics-informed training phase where the momentum loss is added to the network's loss function. This is done to simulate the actual momentum and mixture fraction reconstructions, all of which make use of a data pre-training phase to speed up the training process and improve the network's performance [26]. The effect of the aforementioned parameters is therefore measured relative to their impact on the physics-informed training phase.

Effect of the learning rate

The effect of the learning rate is tested first. The chosen value fixes the initial learning rate used by the Adam optimizer which, by construction, adapts the learning rate iteratively during training, improving its convergence compared with standard stochastic gradient descent [116]. During data training the learning rate is set to 10^{-3} since that value provides a good trade-off between stability and convergence time for the data loss, and is the value used in previous work [26]. The effect of the learning rate during physics-informed training is tested here. Following previous work the learning rate during physics training is taken to be smaller than that of data training, since the loss landscape is more complex and a smaller learning rate may prevent the loss function from undergoing strong oscillations during the early stages of training. Four different values of the learning rate are tested, namely 10^{-3}, 10^{-4}, 10^{-5}, and 10^{-6}. During these tests the models are trained on both the momentum and the data loss terms, simulating the setup of full momentum reconstructions. The weight of the data loss is set to 10^2, the batch size is set to 10^4, and no velocity boundary conditions are used, following previous work [26, 93].

The learning rate is found to strongly affect the stability of the loss function during training. Learning rates larger than 10^{-6} are found to lead to significant oscillations in the loss functions. This hinders the learning process and leads to higher values of the momentum loss at the end of training. Figure 6.1 shows the effect of the learning rate on the evolution of the momentum loss. It can be seen that the lowest learning rate of 10^{-6} yields the most stable loss function, preventing the high frequency oscillations that occur when the learning rate is set between 10^{-3} and 10^{-5}. It should be noted that these oscillations are likely affected by the choice of batch size. Additionally, regardless of the learning rate, the momentum losses exhibit low frequency, high amplitude oscillations, which are undesirable. They are attributed to the fact that the combination of hyperparameters here is not yet optimal, which triggers some of the failure modes of the algorithm, discussed at the end of this section. The key point to take away here is that high values of the learning rate yield high-frequency oscillations in the loss function. These can be suppressed by setting a lower learning rate.

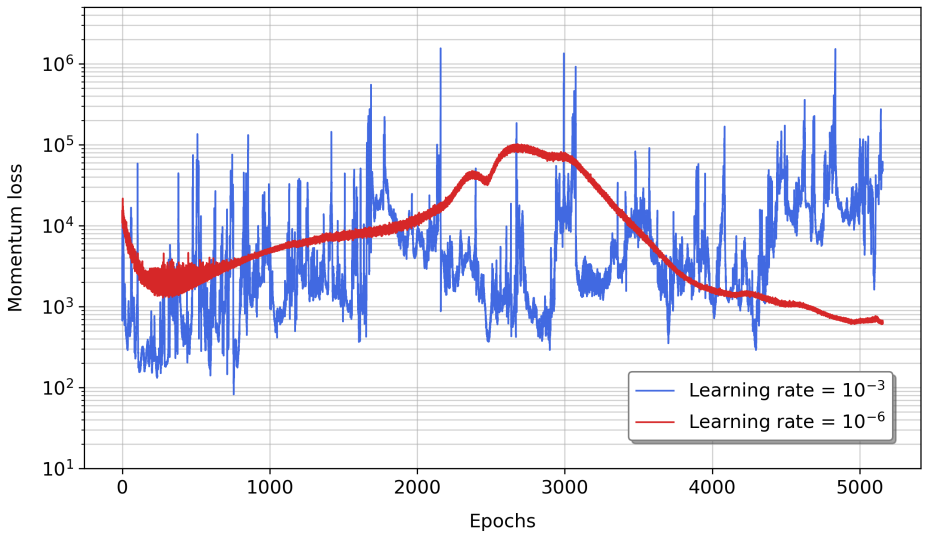
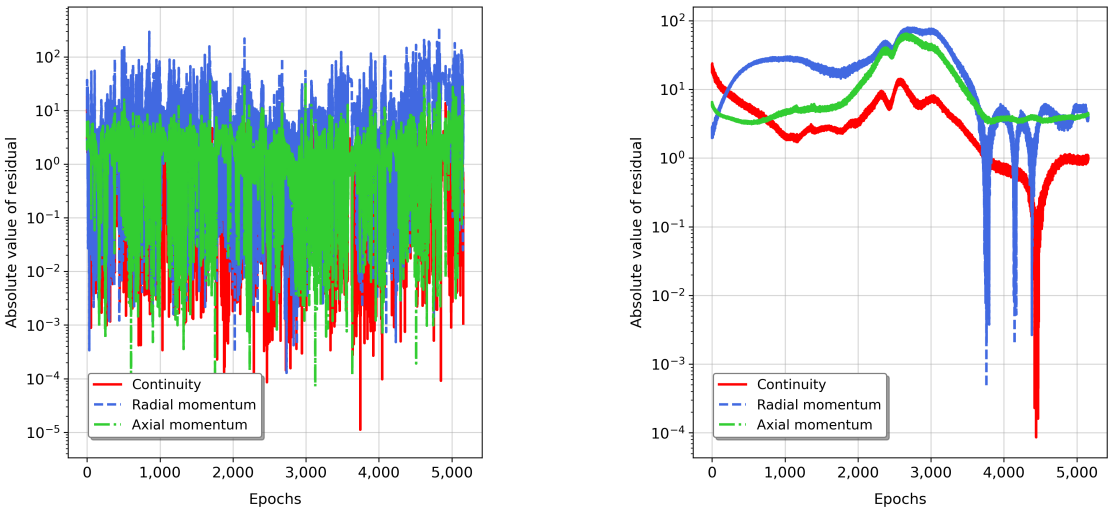


Figure 6.1: Effect of the learning rate on the evolution of the momentum loss.Only the two extreme cases (learning rates of 10^{-3} and 10^{-6}) are shown for better readability. The remaining learning rates (10^{-4} and 10^{-5}) exhibit similar oscillatory behavior as the test with a learning rate of 10^{-3}.

The oscillations in the momentum loss are likely related to the computation of the residuals of the continuity and Navier-Stokes equations. The evolution of these residuals for the tests with a learning rate of 10^{-3} and 10^{-6} are shown in Figure 6.2. The same behavior seen in Figure 6.1 is visible here. Choosing a high value of the learning rate (10^{-3}) yields significant oscillations in the computation of the residuals, whereas the lower learning rate (10^{-6}) suppresses these oscillations. Due to the improved stability in both the residuals and the momentum loss, a learning rate of 10^{-6} is selected to perform the physics-informed reconstructions.



(a) High learning rate (10^{-3})

(b) Low learning rate (10^{-6})

Figure 6.2: Effect of the learning rate on the residuals of the continuity and Navier-Stokes equations.

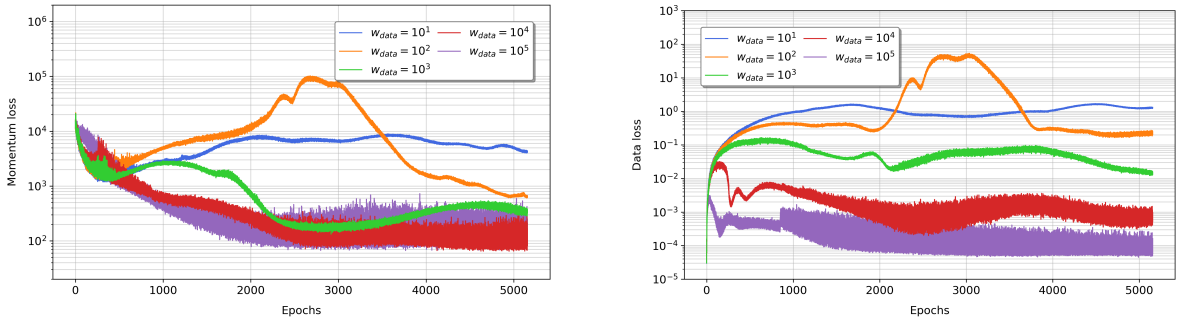
Effect of the data loss weight

The effect of the data loss weight (w_{data}) is analyzed next. The optimal learning rate of 10^{-6} is used while maintaining the batch size at 10^4. The data weight is then sampled from 10^1 to 10^5, taking steps in powers of ten to obtain an evenly sampled logarithmic distribution. The effect of the data weight on the evolution of the loss components, both momentum and data, is shown in Figure 6.3.

Several observations can be made from the evolution of the losses. Low values of the data weight are not able to apply a sufficiently strong constraint on the total loss,. As a consequence the total loss is totally dominated by the momentum loss and practically disregards the data term. As a result, the final value of the momentum loss at the end of training is higher, indicative of a worse reconstruction. Additionally, the data loss is allowed to grow significantly during training due to the low weight. Moreover, using a low value of the data weight appears to lead to losses with less variance, i.e. less large frequency oscillations.

On the other hand, using a higher data weight appears to be beneficial for the minimization of the momentum loss, albeit of the cost of increased variance. Increasing the data weight imposes a stronger constraint on the total loss, forcing the momentum loss to decrease in a way that is consistent with the training data. As a result the final value of the data loss is lower. Additionally, higher data weights lead to lower values of the momentum loss. It should be noted that this improvement only occurs up to a certain point. If the data weight is set too high, the constraint imposed by the data term on the total loss may prevent the momentum loss from decreasing sufficiently.

The effect of the data weight shown in Figure 6.3 should be understood with respect to the difference between the data and momentum losses at the start of training. Figure 6.3 shows that a data weight of 10^5 yields the best performance, leading to the lowest value of both loss components by the end of training, albeit at the cost of increased variance. But this is only meaningful relative to the initial



(a) Momentum loss

(b) Data loss

Figure 6.3: Effect of the data loss weight (w_{data}) on the loss evolution during training

discrepancy between the two loss terms — as a consequence of data training the data loss starts at a value of 10^{-4}, while the momentum loss starts at a value of 10^4. It is this relative difference that is important, not necessarily the absolute value of the data weight. Using a weight of 10^5 in this case brings the effective value of the data loss at the start of training from 10^{-4} to 10^1, which results in a gap of three orders of magnitude between it and the momentum loss. This is the key conclusion drawn from this analysis. It appears that maintaining a difference of three to four orders of magnitude between the data loss and the momentum loss at the start of physics-informed training yields the best performance. Any less, and the data loss becomes irrelevant. Any more, and it may prevent the momentum loss from decreasing sufficiently. This guideline is used for the remainder of this thesis to select the appropriate data weight at the beginning of each reconstruction.

Effect of the batch size

Using the most favorable data weight (10^5) and learning rate (10^{-6}), the effect of the batch size is tested next. The batch size determines the amount of collocation and measurements points sampled at each training iteration to compute the network's loss. A large batch size will generally lead to loss functions with less variance, since more points from the domain are sampled at each iteration, making the loss less likely to suffer severe oscillations in between batches. This comes at the expense of a higher computational cost for a given number of iterations, since a larger amount of data needs to be processed. Additionally, a larger batch size also implies that less iterations are needed to train the network for a given number of epochs. On the other hand, a low batch size is computationally cheaper but may result in losses with high variance, as the network struggles to generalize due to the small sample size at each iteration and the loss is more strongly affected by the variations between samples.

Considering that the total number of points in the domain is in the order of 10^5, a range of batch sizes between 10^3 and 10^4 has been tested. A comparison of the evolution of the momentum loss for the selected batch sizes is shown in Figure 6.4. A fixed number of $100,000$ training iterations is used for these tests. Depending on the batch size, the total number of epochs will vary. For this reason the horizontal axis of Figure 6.4 shows the training iterations and not the epochs to more clearly compare the evolution of the losses of the different runs.

Generally, it can be seen that all batch sizes lead to momentum losses with similar behavior. They all converge to approximately the same value, in the order of 10^2, by the end of training. The smaller batch sizes lead to losses with larger variance, as expected. This is particularly noticeable for the smallest batch size of $1 \cdot 10^3$, whose loss oscillates significantly during training as a consequence of the sampling of the collocation points. Batch sizes of $5 \cdot 10^3$ and $1 \cdot 10^4$ show a similar level of variance, whereas the largest batch size of $5 \cdot 10^4$ shows the least variance of all. Based on the data of Figure 6.4 alone it would appear that choosing the largest batch size is desirable. However, recall that the batch size also affects the computational cost of the training process. This factor should also be taken into account when making a selection. Performing a single iteration with a batch size of 10^4 is more expensive that doing so with a batch size of 10^3. Figure 6.4 therefore paints an incomplete picture of the effect of the batch size.

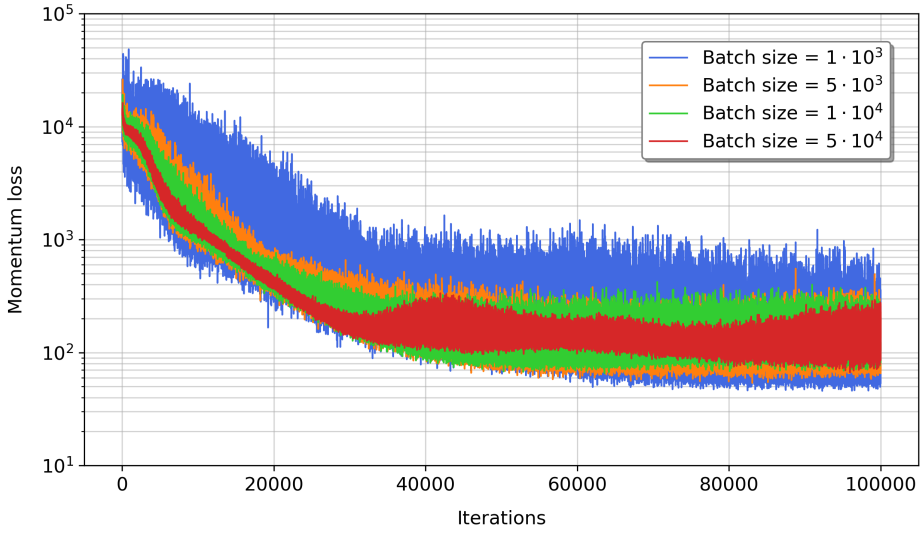


Figure 6.4: Effect of the batch size on the evolution of the momentum loss. Note that the horizontal axis shows iterations and not epochs to more clearly compare the evolution of the losses.

To address this, Table 6.2 compares several metrics of the batch size tests, including the runtime of the simulations. These metrics contain useful information to make a better judgment on the optimal batch size. In order to perform a fair comparison, all tests have been run using the same hardware. The data shows that the batch size does not significantly affect the minimum and final value of the momentum loss, in agreement with Figure 6.4. All four runs show very similar performance for these two metrics, that is, all four runs end up converging to a similar loss value within the 100,000 iterations. However, a significant difference can be observed in the runtime — the three smallest batch sizes yield similar runtimes, while the largest batch size ($5 \cdot 10^4$) is significantly more computationally expensive than the rest. Taking this into consideration, the second largest batch size (10^4) is deemed to be the optimal choice to perform the reconstructions. This batch size is large enough to result in a loss function with low variance without significantly increasing the computational cost. Additionally, it yields the lowest final value of the momentum loss of the four runs. A batch size of 10^4 is therefore selected to perform the reconstructions.

Table 6.2: Effect of batch size on training performance. Best performance highlighted in gray. The largest batch size significantly increases the runtime, shown in red.

Batch size	**Epochs** (after 100,000 iterations)	**Minimum momentum loss**	**Final momentum loss**	**Runtime**$^{(*)}$
$1 \cdot 10^3$	515	$4.58 \cdot 10^1$	$2.52 \cdot 10^2$	2 hrs, 28 min
$5 \cdot 10^3$	2,578	$5.38 \cdot 10^1$	$1.36 \cdot 10^2$	2 hrs, 30 min
$1 \cdot 10^4$	5,156	$6.56 \cdot 10^1$	$1.06 \cdot 10^2$	2 hrs, 32 min
$5 \cdot 10^4$	25,780	$6.67 \cdot 10^1$	$2.01 \cdot 10^2$	3 hrs, 20 min

$^{(*)}$ Using a single NVIDIA Quadro RTX 8000 GPU @ 33 MHz, running CUDA Version 12.1.

Failure modes

Several failure modes of the reconstruction algorithm have been identified while conducting the preliminary tests. These are briefly discussed here. It is important to understand how the PINN fails in order to better understand the behavior of the network during more complex reconstructions, such as those involving the mixture fraction. This will prove to be useful when analyzing results in subsequent sections of this chapter. Understanding the way in which the PINN fails can be helpful to recognize patterns in the results of the reconstruction attempts and the behavior of the network during training. This is very useful to gain a deeper understanding of what is happening to the network

behind the scenes, and can be used to conduct a more critical assessment of the reconstruction results and generalize on the applicability of PINNs as a flow field reconstruction method for other problems.

The first pathology of the reconstruction algorithm is that it appears to be quite sensitive to the choice of hyperparameters. The interplay between the data and momentum losses (and in later tests, the mixture and boundary condition losses) can be complex and result in loss behavior that is difficult to predict. Although the reconstruction algorithm generally manages to successfully reduce the momentum loss while maintaining a sufficiently low value of the data loss, it does not always do so in a steady, smooth manner. In some reconstructions the losses show significant variance as a consequence of the sampling of collocation points, whereas in other runs the losses are well behaved and decrease steadily during training.

Additionally, the momentum loss has been found to stagnate at a value close to 10^1 and be difficult to minimize beyond that point. The PINN exhibits a tendency to find trivial solutions to the reconstruction problem when given too much freedom. It does so by reconstructing the velocity fields as constant near-zero values to quickly minimize the residuals of the momentum loss. This is the reason why a data loss term is needed in the first place. And even in the presence of a data loss the PINN sometimes struggles to balance the two loss terms, leading to trivial solutions regardless. The complexity of the loss landscape makes it such that the PINNs have a tendency to get stuck in basins of local optima (the trivial solutions) that are difficult for the network to escape from.

Because of the tendency of the PINNs to drive the velocity reconstruction towards trivial solutions, the use of velocity boundary conditions is not particularly effective. The only velocity boundary conditions that can be imposed without providing DNS data are those that are known *a priori*. These are the values of axial and radial velocity in the far field, which can be assumed to be approximately zero. The reference axial velocity is exactly zero in the far field, whereas the radial velocity takes on an average value of $-3.7 \cdot 10^{-3}$. Providing zero-velocity boundary conditions in the far field during data training has been found to be detrimental for the PINN. As a consequence of the boundary conditions the PINN estimates the velocity fields to be constant and equal to the values enforced in the far field. This makes it difficult for the PINN to then correct its estimation of the velocities in the flame region during physics-informed training. Providing velocity boundary conditions during data training essentially encourages the PINN to find trivial solutions for the velocity fields. This is undesirable as it makes it harder to fight against the inherent tendency of the PINN to go towards trivial solutions which is already there to begin with.

Providing velocity boundary conditions only during physics training is less detrimental to the PINN than doing so at the data training stage. Nevertheless, this has not been found to yield any meaningful improvements in the reconstruction accuracy nor the behavior of the network during training. It appears that other factors such as the interplay between the data and physics losses have a much larger effect on the network's behavior. For this reason, unless explicitly stated otherwise, the reconstructions shown in this chapter do not use velocity boundary conditions.

Another way in which the PINN fails, which is related to the aforementioned complexity of the loss landscape and the oscillations of the loss function, is through instability of the reconstructed fields. During some reconstruction attempts the oscillations in the loss are so severe that the network adjusts its weights in a way that leads to the introduction of unphysical and erroneous features into the reconstructed flow fields. These are small localized regions of high error that generally enter the domain through its boundaries (as opposed to simply appearing in the middle of the domain).

Depending on the configuration of the model and the state of the reconstructions prior to the appearance of these errors, the network is sometimes robust enough to suppress them, eliminating them completely and returning to its initial state. Sometimes, however, the network cannot suppress the errors. The affected regions then tend to grow and propagate within the domain, completely ruining the quality of the reconstructions. An example of how these erroneous regions propagate into the domain through the boundaries is shown in Figure 6.5.

It is believed that these failure modes are interrelated. The losses oscillate, causing erroneous regions to appear in the domain. The network may get stuck in local optima within its loss landscape, and the PINNs have a tendency to diverge towards trivial solutions to the reconstruction problem. The reasons

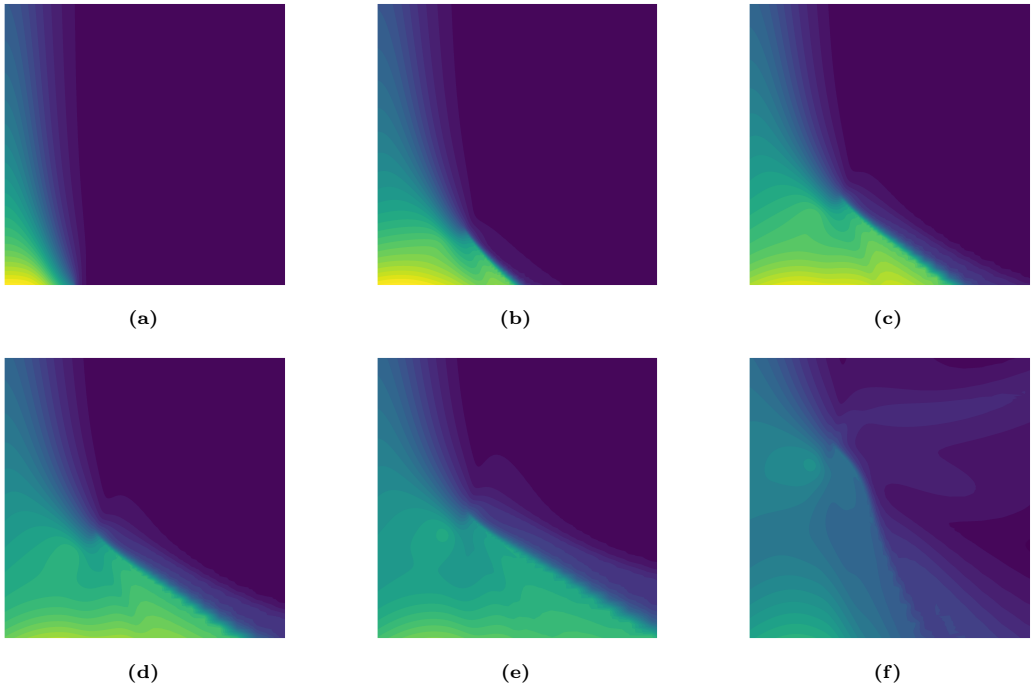


Figure 6.5: Erroneous region propagating into the domain through the bottom boundary. A successful reconstruction of the mixture fraction field (a) is affected by an erroneous oscillation that enters through the bottom of the domain, widening the mixture fraction profile near the base of the flame (b). The oscillation grows (c) and propagates upwards (d, e) until the reconstructed mixture fraction field is completely corrupted (f).

for these failure modes are not fully understood, although the discussion presented here does provide some clues regarding their origin. The are believed to be a result of the inherent training pathologies of PINNs discussed in Chapter 4 (Section 4.4), in addition to being related to the spurious oscillations that appear in the computation of gradients using automatic differentiation, as discussed in Chapter 5.

In any case, despite the fact that the reconstruction algorithm can exhibit a large degree of instability during training, it is also quite robust. The results presented in the following sections demonstrate that the PINN is capable of producing remarkably accurate reconstructions, even in the absence of data for several flow variables (see Section 6.5). Several factors need to come together to obtain successful reconstructions. The configuration of the network and the choice of hyperparameters needs to be carefully selected, and the data training needs to result in a favorable initialization of the velocity fields. And even then, there is some randomness due to the initialization of the weights and the sampling of the collocation and residual points, which can affect the behavior of the network and the quality of the reconstructions.

The purpose of this discussion is to make the reader aware of these failure modes as we present the results of various reconstructions in the following sections. Being mindful of the ways in which the PINN can fail has proven to be absolutely crucial in order to obtain successful reconstructions. It provides clues to identify ways in which a failed reconstruction can be adapted to nudge the PINN in the right direction and transform a failed attempt into a successful reconstruction. A significant amount of oversight is needed to select the right weights to balance the loss terms effectively and to critically evaluate the results of the reconstructions.

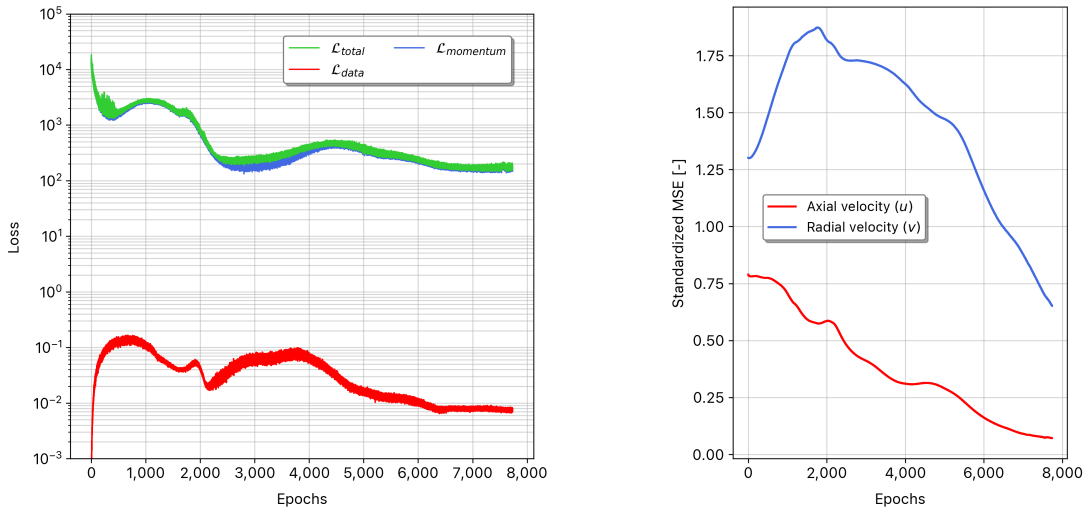
6.2. Baseline momentum reconstruction

A baseline momentum reconstruction has been performed to establish a benchmark that can be used to assess any changes to the model's performance following the addition of the mixture fraction. This baseline reconstruction is purely momentum-based — it uses the standard PINN model without the inclusion of the mixture fraction loss nor any mixture fraction data. The results of Section 6.1 are used to select the setup for this reconstruction. As such, the baseline reconstruction presented here contains the optimal configuration based on the analysis conducted in the previous section. The results of this baseline are the best velocity reconstructions that have been obtained using the standard PINN model. The setup of this baseline reconstruction is summarized in Table 6.3.

Table 6.3: Settings of the baseline momentum reconstruction

Parameter	**Typical value**
Input variables	Spatial coordinates (r, z)
Output variables	u, v, ρ, p, T
Training variables	ρ, p, T
Reconstructed variables	u, v
Hidden layers	15
Neurons per layer	75
Activation function	Sigmoid linear unit (SiLU)
Training scheme	Data training followed by physics training
Data training duration	$6,000$ epochs
Momentum training duration	$7,500$ epochs
Weight of momentum loss	10^0
Weight of data loss	10^5
Batch size	10^4
Optimizer	Adam
Learning rate	10^{-3} for data training, 10^{-6} for physics training
Number of saved model states	100 to 200

The performance of the network during the data training phase is not discussed. It suffices to state that the PINN accurately reconstructs the data fields by the end of the data training phase. The performance of the network during momentum training is shown in Figure 6.6.



(a) Evolution of losses

(b) Evolution of reconstruction errors

Figure 6.6: Training performance of the PINN during the baseline reconstruction. The total loss is dominated by the momentum loss. For this reason the two losses coincide almost exactly during the majority of the training process.

Figure 6.6a shows the evolution of the loss components during training. The data loss starts out at a very low value, in the order of 10^{-5}, as a result of the successful data training phase. The data loss increases rapidly during the early part of training as the network adjusts its weights to accommodate the newly introduced momentum loss. During this initial training phase the momentum loss decreases rapidly by an order of magnitude from 10^4 to 10^3. The two loss components reach an equilibrium within the first 500 epochs. At that point the data loss stabilizes at a value of 10^{-1} — it is not allowed to increase further due to its weighting factor in the network's total loss function. After 3,000 epochs the momentum loss has decreased by another order of magnitude to 10^2, while the data loss remains at 10^{-1}. For the remainder of the training process the momentum loss slowly oscillates and reaches a final value in the order of 10^2. The data loss continues to decreases, reaching a final value in the order of 10^{-2}.

Figure 6.6b shows the evolution of the reconstruction errors for the two velocity components during momentum training. The reconstruction errors of both velocity components decrease significantly. The axial velocity error decreases steadily, while the radial velocity error shows some transient behavior at the early stages of the reconstruction. The difference between the axial and radial velocity errors remains roughly constant, indicating that the reconstruction is equally effective at reducing both errors in absolute terms. The radial velocity error is higher than the axial velocity error throughout the reconstruction, indicating that the PINN finds it particularly difficult to reconstruct the radial velocity. This is a trend that can be observed in most of the reconstruction attempts discussed in the remainder of this thesis. A likely explanation for this can be understood by examining the reconstructed axial and radial velocity fields at the end of training, shown in Figure 6.7 and Figure 6.8, respectively.

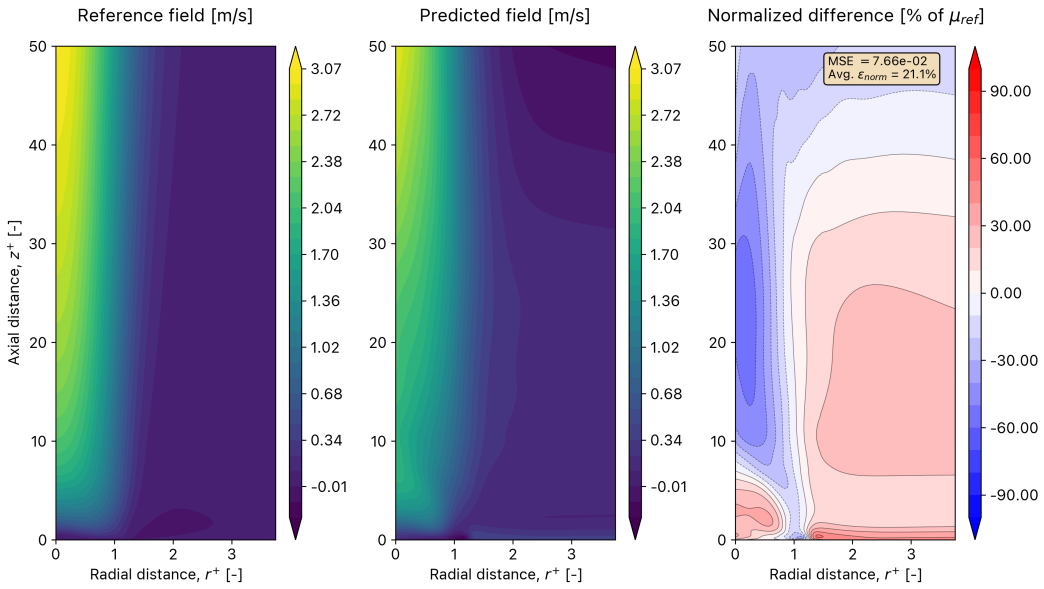


Figure 6.7: Baseline axial velocity reconstruction

It can be seen that the radial velocity field is characterized by a small localized region of negative radial velocity near the base of the flame. This is a result of the air entrainment caused by baroclinic vorticity. As discussed in Chapter 2 the misalignment of the density and pressure gradients leads to the generation of such vorticity, which dominates the vorticity field of the flame. The vortices that form near the base of the flame are responsible for the air entrainment mechanism that gives the radial velocity field its characteristic shape. Beyond that small localized pocket of air entrainment, the radial velocity field is essentially constant. This appears to be problematic for the PINN. As shown in Figure 6.8 the PINN is capable of successfully reconstructing the magnitude of the radial velocity and to capture the existence of a region near the bottom of the domain with negative radial velocity. The PINN even manages to correctly locate the center of the air entrainment region. However, it is not capable of fully resolving this high-gradient region, which results in an underestimation of the absolute magnitude of the air entrainment velocity. The large variations in radial velocity are difficult for the PINN to resolve. This is a recurring issue throughout most of the reconstruction attempts, and in order to evaluate the

quality of the reconstructions significant attention is paid to the ability of the PINN to resolve the air entrainment region.

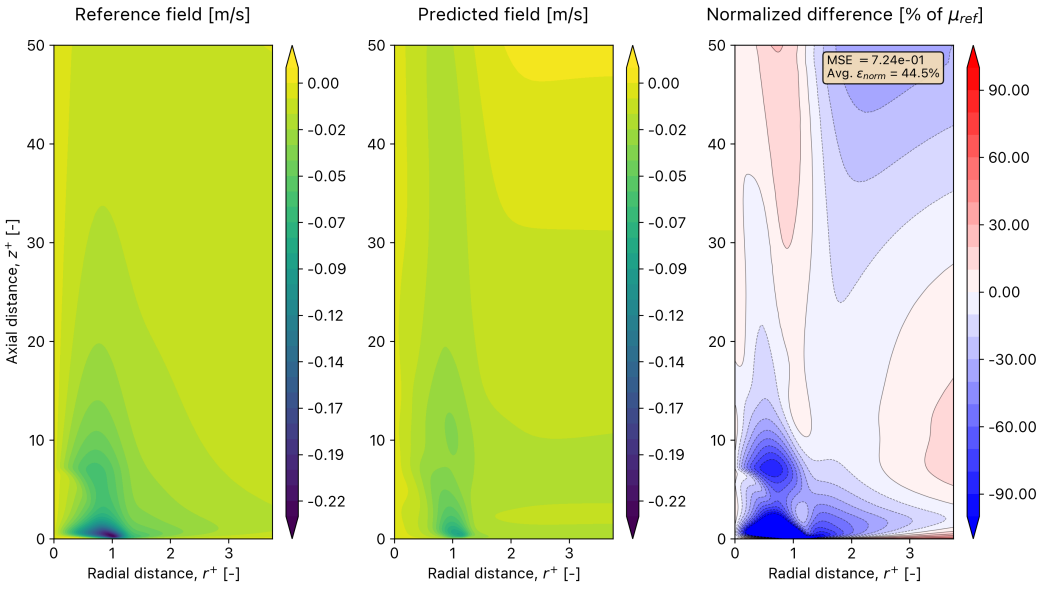


Figure 6.8: Baseline radial velocity reconstruction

On the other hand, the axial velocity field shown in Figure 6.7 shows a smoother transition from the large velocity region in the plume of the flame to the zero velocity region in the far field. The PINN is capable of reconstructing the axial velocity significantly better than the radial velocity. It correctly reconstructs the characteristic jet-like shape of the plume, in addition to the axial velocity magnitude within it. The upper region of the plume is slightly underestimated by the PINN, while the base of the flame is not perfectly resolved and shows some deviation with respect to the DNS data. Nevertheless, the characterization of the axial velocity field is generally better than that of the radial velocity.

Considering the fact that the PINN has never seen any velocity data at any point during the training process, its ability to correctly reconstruct the general features of the axial velocity profile is deemed sufficient. The same is true, albeit to a lesser extent, for the radial velocity. The baseline reconstruction is therefore an adequate benchmark for the mixture fraction tests — matching the baseline reconstruction accuracy is deemed sufficient, although improvements in the air entrainment region are desirable.

Table 6.4 provides a summary of several performance metrics of the baseline reconstruction to give a reference point in the evaluation of the mixture fraction tests. For the sake of completeness, the state of the data fields at the end of the baseline reconstruction are shown in Figure 6.9 (density), Figure 6.10 (pressure), and Figure 6.11 (temperature). It can be seen that the reconstruction process has not corrupted the data fields, which are still correctly estimated by the PINN thanks to the appropriate weighting of the data loss during momentum training.

Table 6.4: Performance metrics of the baseline reconstruction

Error metric	**Unit**	**Value**	
		u	v
Standardized global MSE	–	$7.44 \cdot 10^{-2}$	$6.75 \cdot 10^{-1}$
Standardized flame MSE	–	$5.27 \cdot 10^{-2}$	$1.24 \cdot 10^{0}$
Average absolute global normalized error	% of μ_{ref}	20.9	42.7
Average absolute flame normalized error	% of μ_{ref}	18.0	69.7

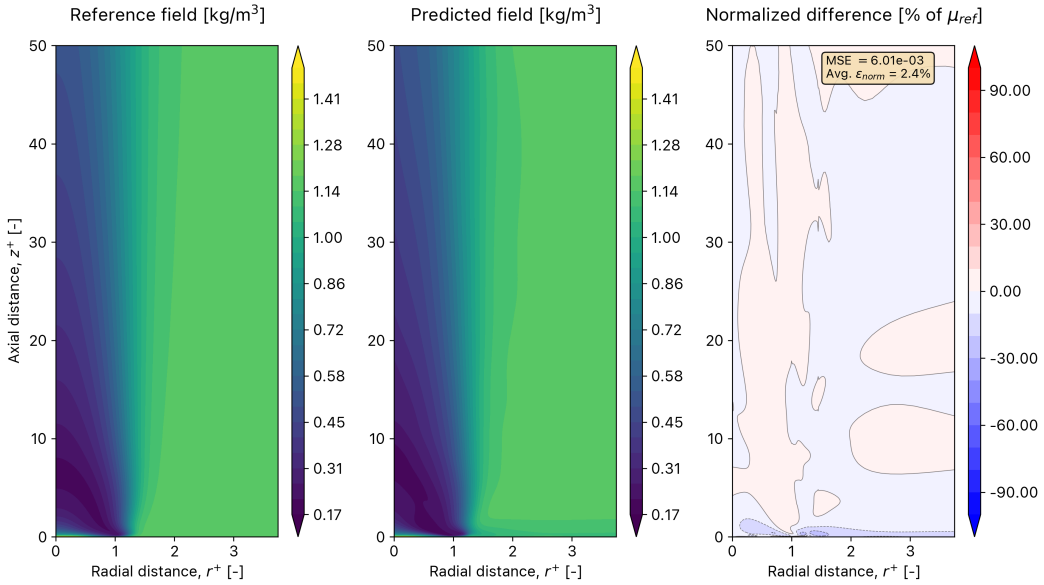


Figure 6.9: Baseline density reconstruction

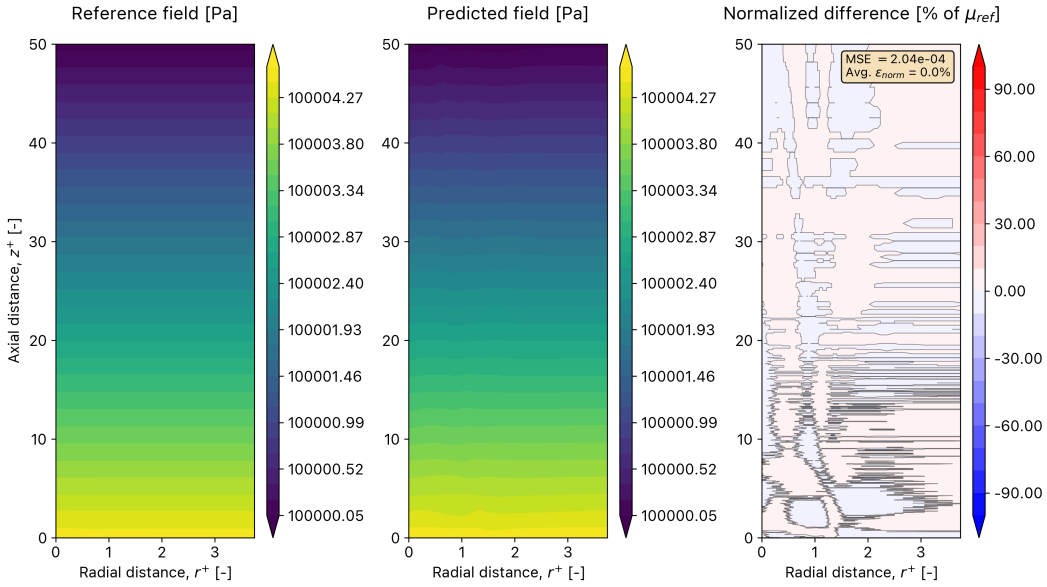


Figure 6.10: Baseline pressure reconstruction

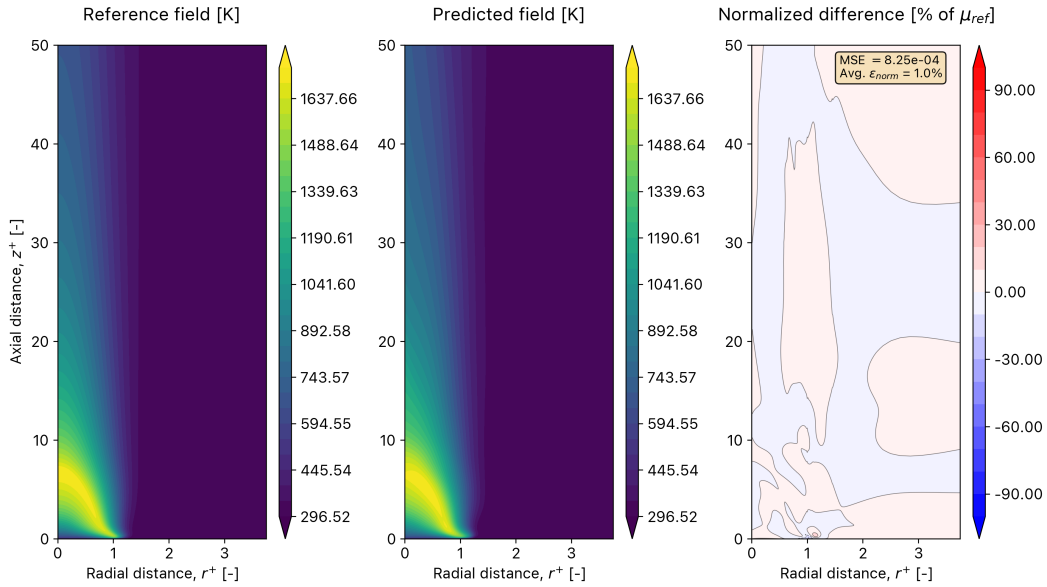


Figure 6.11: Baseline temperature reconstruction

6.3. Mixture fraction as a reconstruction target

Prior to this work, the mixture fraction had not been incorporated into PINN models aimed at reconstructing pool fire flow fields. The integration of the mixture fraction within the PINN reconstruction framework is a novel contribution of this thesis. The first series of tests following the addition of the mixture fraction aims to assess the efficacy of the extended PINN model at reconstructing the mixture fraction field. These tests, discussed in this section, serve a dual purpose. Firstly, they enable an examination of the algorithm's response to the increase in complexity that comes as a consequence of its extension with the mixture fraction. The model is extended to include an additional output variable (the mixture fraction) and an additional term in its loss function (the mixture loss). The tests provide insights into the response of the model to these modifications.

Additionally, the tests provide an opportunity to assess the model's ability to reconstruct the mixture fraction field. This is the first attempt at reconstructing the mixture fraction field of a pool fire using a PINN, and therefore the capabilities of the newly developed reconstruction framework are unknown. In the case that the model can effectively reconstruct the mixture fraction field the accuracy of these reconstructions is assessed. The performance of the network during training may provide valuable insights into the interplay between the mixture loss, the momentum loss, and the data loss. This is useful not only in the context of this thesis, but also in a general sense to better understand the limitations of PINNs when tackling more complex reconstruction problems that involve additional loss terms.

Several tests have been performed to reconstruct the mixture fraction, treating it as an additional reconstruction target, analogous to the velocity components in standard momentum-based training. Initially the extended PINN model undergoes a data training phase, focusing solely on density, pressure, and temperature data. This is followed by a physics-informed training phase, where momentum and mixture losses are incorporated into the network's loss function, alongside the data loss term, to reconstruct the velocity and mixture fraction fields.

It should be noted that the data training phase is different than in previous tests. An additional loss component is included to enforce boundary conditions on the mixture fraction field. The boundary conditions are only applied in regions where the value of the mixture fraction is known *a priori*. This is the case in the far field (right boundary) and in the bottom boundary. The far field is unaffected by the flame. Hence, at the right boundary, the fluid is composed only of air, with no fuel or reaction products. This corresponds to a mixture fraction of zero — a boundary condition is imposed on

the right boundary to force $\xi = 0$. The bottom boundary is made up of the fuel inlet that spans the pool radius ($r \leq a$) and an isothermal wall outside of the pool ($r > a$), where a is the pool radius. At the fuel inlet the mixture fraction is set to its maximum value of $\xi = 0.81$. At the isothermal wall the composition is known to be made up only of oxidizer, and hence the mixture fraction is set to zero.

The standard PINN architecture has been extended to include the mixture fraction as an additional output variable. The width of the network (the amount of neurons per layer) is increased to maintain the ratio of 15 neurons per output variable used in previous work [26, 76, 93]. This results in an increase from 75 neurons per layer in the standard case to 90 neurons per layer in the extended PINN. The depth of the network (the amount of layers) is left unchanged. Typical settings used in the mixture fraction reconstructions are shown in Table 6.5.

Table 6.5: Typical settings of the extended reconstruction algorithm for the mixture fraction reconstructions

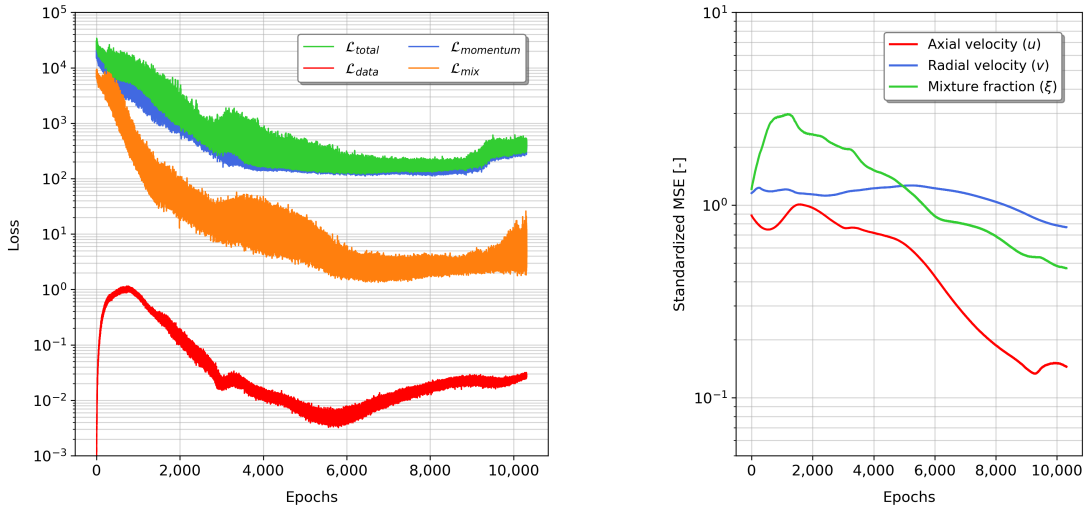
Parameter	**Typical value**
Input variables	Spatial coordinates (r, z)
Output variables	u, v, ρ, p, T, ξ
Training variables	ρ, p, T
Reconstructed variables	u, v, ξ
Hidden layers	15
Neurons per layer	90
Activation function	Sigmoid linear unit (SiLU)
Training scheme	Data training followed by physics training
Training duration	10,000 to 15,000 epochs (total)
Weight of loss components	10^2 to 10^4 (defined w.r.t $\mathcal{L}_{\text{phys}}$)
Batch size	10^4
Optimizer	Adam
Learning rate	10^{-3} for data training, 10^{-6} for physics training
Number of saved model states	100 to 200

The reconstruction attempts are divided into two groups. In the first set of tests, discussed in Section 6.3.1, the momentum and mixture losses are added to the network simultaneously after data training. This type of reconstruction is referred to as a *single-stage* reconstruction, since the physics-informed training is performed in one single stage. The second group of tests, discussed in Section 6.3.2, are *multi-stage* reconstructions, where the momentum and mixture losses are instead added successively one after the other. In this way the physics-informed training phase is split into two separate stages. The terms single-stage and multi-stage training are used for the remainder of this thesis to classify reconstructions based on the way in which the physics-informed losses are incorporated into the network during training.

6.3.1. Single-stage reconstruction

The results of the single-stage reconstruction are discussed hereinafter. The initial data training phase is performed successfully. The performance of the network during the physics-informed training phase is shown in Figure 6.12. Figure 6.12a shows the evolution of the loss components, while Figure 6.12b shows the evolution of the reconstruction errors for the three target variables, namely axial velocity, radial velocity, and mixture fraction.

Figure 6.12a shows that both physics-informed loss components decrease steadily during training, stabilizing after around 6,000 epochs. The data loss is initially very low, in the order of 10^{-6}, thanks to the data training phase. It then increases rapidly as the network adjusts its weights to accommodate for the new loss components, slightly worsening the predictions of the data fields in the process. Due to the weight on the data loss, here set to 10^3, the data loss settles at a value of 10^0 within the first 500 epochs and is not allowed to increase any further. During the rest of training the data loss decreases back down to 10^{-3} before settling at 10^{-2}. Even though this value is significantly larger



(a) Evolution of losses

(b) Evolution of reconstruction errors

Figure 6.12: Training performance of the PINN during the single-stage mixture fraction reconstruction

than the initial value of 10^{-6}, the data fields are still reconstructed exactly by the PINN, as shown by Figure 6.13. This indicates that a data loss of 10^{-2} is sufficiently low to accurately represent the data fields, and any improvements beyond that point do not result in significant changes in the large-scale flow features that are reconstructed by the PINN. The mixture fraction boundary condition loss shows a similar evolution to the data loss, as one would expect. It settles lower than the data loss at around 10^{-3}.

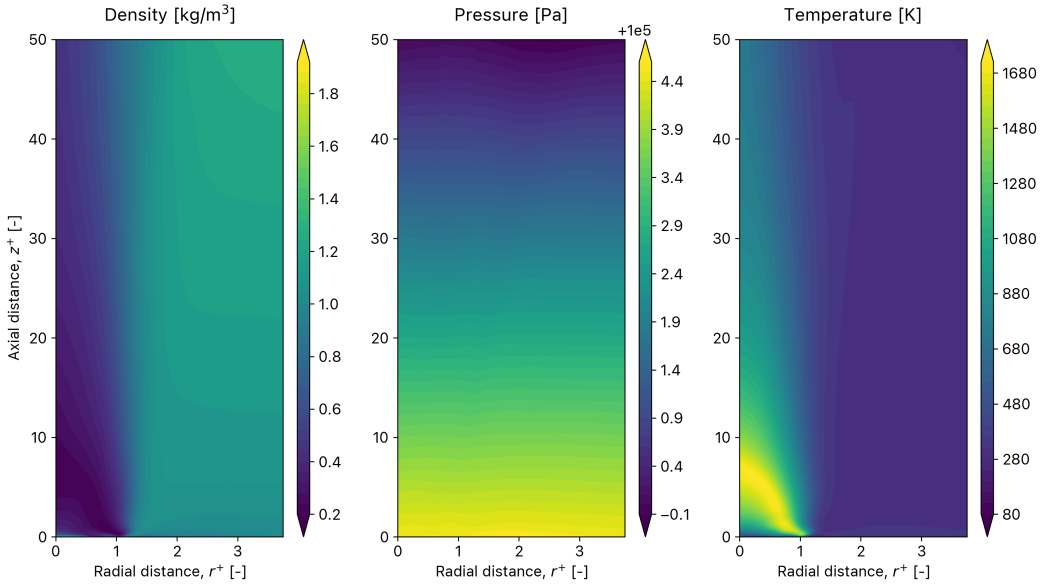


Figure 6.13: State of the data fields at the end of training. Despite some minor oscillations and a substantial increase in the data loss, the PINN is still able to accurately reconstruct all data fields.

Figure 6.12b shows that the reconstruction errors for all three target variables decrease during training. The axial velocity errors shows the largest decrease, followed by the mixture fraction error. The radial velocity error seems to be harder for the PINN to minimize, as seen also in the baseline reconstruction of Section 6.2. Figure 6.14 to 6.16 show a comparison between the reconstructed fields at the end of training and the DNS data for the three target variables.

Figure 6.14 and Figure 6.15 show that, qualitatively, the reconstruction of the velocity fields is comparable to that of the baseline case. The general shape of the axial velocity profile is correctly identified. The PINN correctly identifies the plume region as a positive axial velocity region that stretches along the left boundary of the domain. Nevertheless, the network slightly underestimates the magnitude of the axial velocity at the bottom of the plume region, for $10 < z^+ < 30$. The radial velocity reconstruction is also qualitatively similar to that of the baseline. The PINN struggles to resolve the large gradient region near the base of the flame, as seen in the baseline reconstruction. The magnitude of the radial velocity in that region is underestimated, indicating that the PINN still cannot accurately resolve the air entrainment near the bottom of the flame.

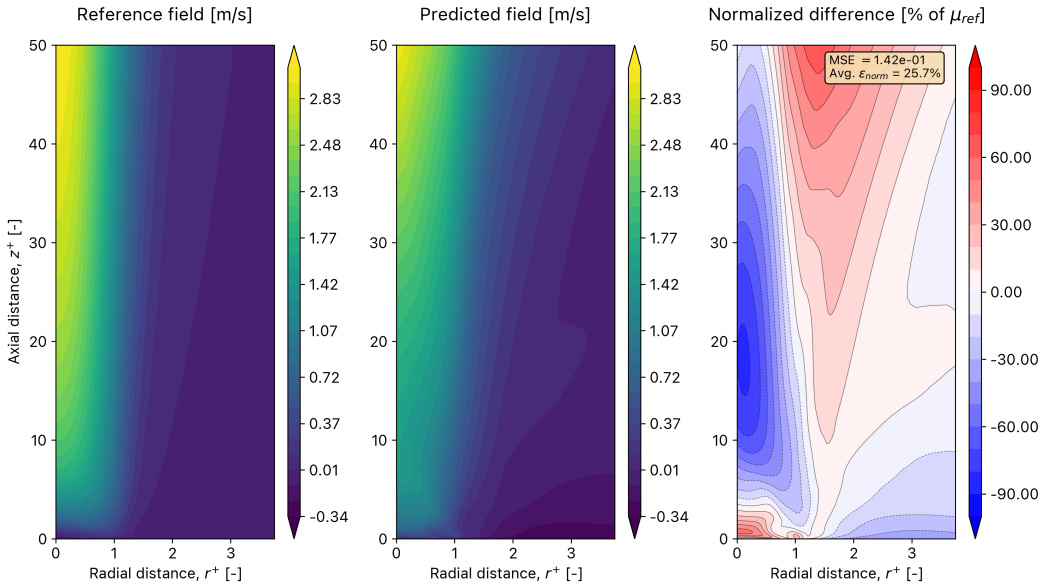


Figure 6.14: Reconstructed axial velocity field after the single-stage reconstruction

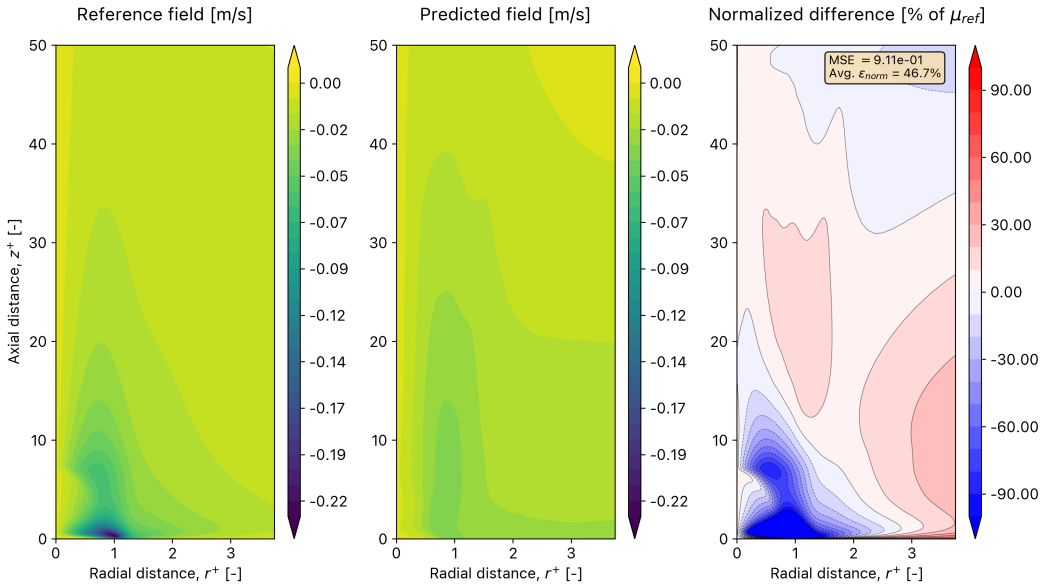


Figure 6.15: Reconstructed radial velocity field after the single-stage reconstruction

An evaluation of the key performance metrics of the reconstruction is shown in Table 6.6. The data is in agreement with the qualitative assessment. Although the metrics have worsened slightly with respect to the baseline they remain in the same order of magnitude, indicating that the two reconstructions are comparable.

Table 6.6: Performance metrics of the single-stage mixture fraction reconstruction

Error metric	**Unit**	**Value**		
		u	v	ξ
Standardized global MSE	–	$1.34 \cdot 10^{-1}$	$8.55 \cdot 10^{-1}$	$5.37 \cdot 10^{-1}$
Standardized flame MSE	–	$1.24 \cdot 10^{-1}$	$1.16 \cdot 10^{0}$	$5.34 \cdot 10^{-1}$
Average absolute global normalized error	% of μ_{ref}	24.8	45.8	115.3
Average absolute flame normalized error	% of μ_{ref}	24.0	57.0	116.4

Figure 6.16 shows the reconstructed mixture fraction field. The PINN is capable of accurately reconstructing the magnitude and shape of the mixture fraction field near the base of the flame. The mixture fraction is also accurately reconstructed in the far field. This can be attributed to the boundary condition loss, which enforces the correct values of the mixture fraction in that regions. The PINN is unable to accurately reconstruct the field in the plume region. Although it correctly depicts the diffusive transport of the mixture fraction upwards with the plume, it overestimates its magnitude in that region. Note that because the average value of the mixture fraction is very low, since it is almost zero everywhere and high only close to the base, the normalized error shown on the right of Figure 6.16 gives an exaggerated depiction of the error in the plume. In any case, it is clear that the mixture fraction reconstruction is not sufficiently accurate in the plume region.

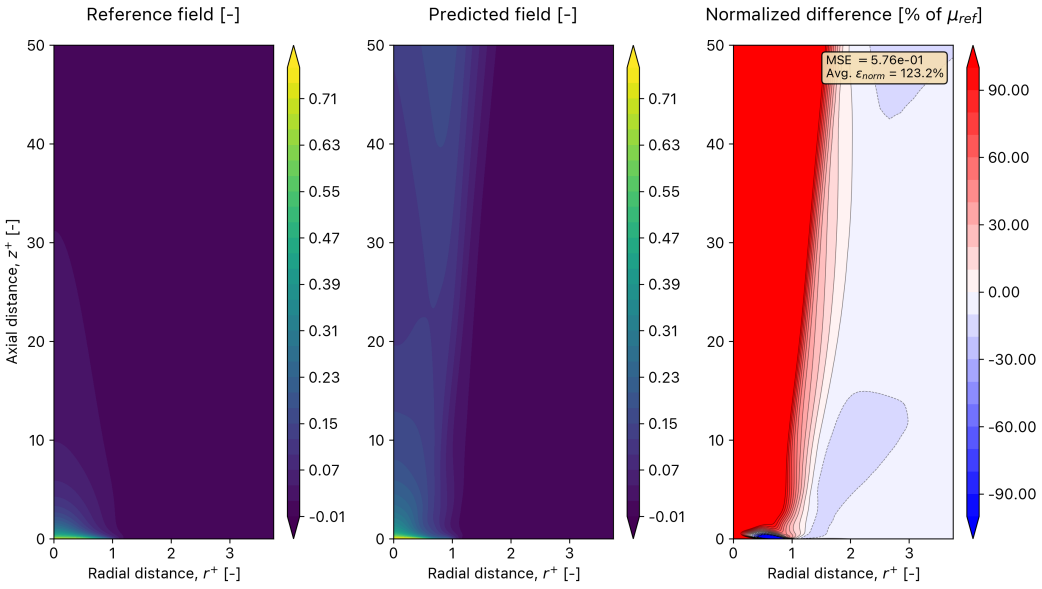


Figure 6.16: Reconstructed mixture fraction field after the single-stage reconstruction

Despite the fact that the accuracy of the mixture fraction reconstruction can be improved, the results shown in this section demonstrate the PINN algorithm's robustness to extensions in its formulation and loss function composition. The extended PINN is capable of reconstructing the velocity fields with similar accuracy to the baseline case, and it can partially reconstruct the mixture fraction field, achieving good reconstruction accuracy in the far field and the bottom boundary, and poor accuracy in the plume region. Additionally, the mixture and momentum losses appear to be compatible with one another, decreasing simultaneously during training.

6.3.2. Multi-stage reconstructions for improved accuracy

A series of multi-stage reconstructions has been conducted with the goal of improving the reconstruction accuracy. We wish to improve the reconstruction of the mixture fraction field in the plume region while maintaining adequate velocity reconstructions.

As discussed above, multi-stage reconstructions are those where the mixture and momentum losses are introduced separately into the training process, instead of simultaneously. Introducing one physics-informed loss component at a time simplifies the loss landscape during training — the motivation for making this modification is that, by simplifying the loss landscape in this way, the network may be able to better optimize each loss component independently, leading to improved reconstruction accuracies by the end of training. Despite the fact that the results of the single-stage reconstruction show that the mixture and momentum losses are compatible, it is hypothesized that introducing them simultaneously may be suboptimal. This has also been discussed in Chapter 4 (Section 4.4), where the use of curriculum learning and warm restarts have been introduced as possible extensions to mitigate the well-known training pathologies of PINNs. Multi-stage reconstructions can be thought of as a way to apply the principles of curriculum learning – the complexity of the learning process is increased progressively over time. In addition, by using multi-stage reconstructions warm restarts are performed in between training phase — the state of the network is maintained and the optimizer is reinitialized while the loss function is updated.

Two types of multi-stage reconstructions have been performed. The first type introduces the mixture loss first, followed by the momentum loss. These are referred to as *mixture-to-momentum* reconstructions. The second group are *momentum-to-mixture* reconstructions, where the order is reversed and the PINN trains first on the momentum loss before introducing the mixture loss.

Mixture-to-momentum training

The mixture-to-momentum multi-stage reconstruction is discussed first. During this run the mixture fraction loss weight is set to $w_{mix} = 1$ since it is expected to dominate the total loss function. The reconstructed mixture fraction field is shown in Figure 6.17. It can be seen that the reconstruction is significantly improved with respect to the first attempt, shown in Figure 6.16. The overestimation of the mixture fraction in the plume region is significantly reduced while maintaining a correct reconstruction near the base and in the right boundary. The velocity reconstructions are not shown since the reconstruction accuracy is very low — this is to be expected due to the absence of a momentum loss term during this training phase.

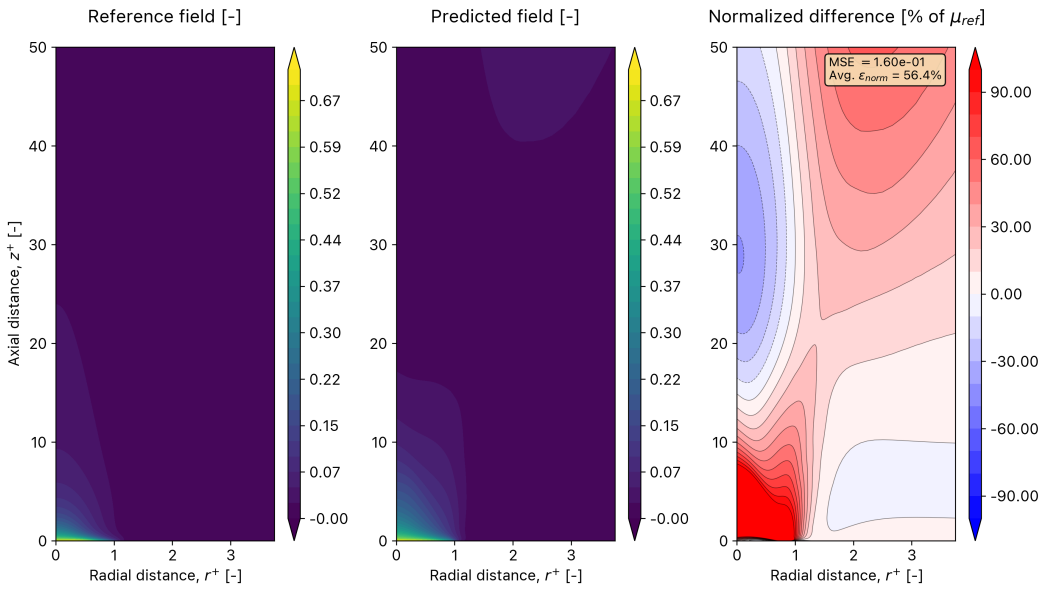


Figure 6.17: Mixture fraction field after mixture training (mixture-to-momentum reconstruction).

The mixture-trained model is saved, and the second training phase is conducted by training this model with a physics-informed loss that now includes the momentum term alongside the mixture term. The resulting reconstruction is shown in Figure 6.18. It can be seen that the mixture fraction reconstruction worsens as a result of momentum training. Nevertheless, it remains better than the one obtained in the single-stage reconstruction of Section 6.3.1. The PINN is capable of accurately reconstructing the general shape of the mixture fraction field. However, it still overestimates the value of the mixture fraction in the plume region above the flame.

To address this limitation an alternative momentum training phase has been conducted. The weight of the data loss is decreased by an order of magnitude to increase the relative weight of the physics-informed terms on the total loss function. By doing so, the goal is to obtain satisfactory velocity reconstructions without sacrificing accuracy in the reconstruction of the mixture fraction field. The resulting mixture fraction reconstruction is shown in Figure 6.19.

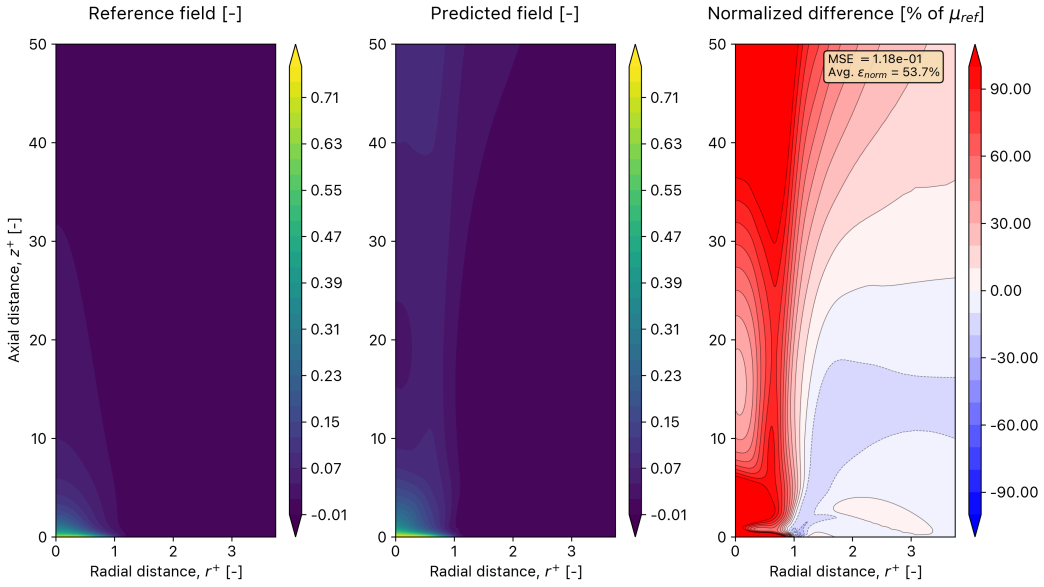


Figure 6.18: Mixture fraction field after momentum training (mixture-to-momentum reconstruction).

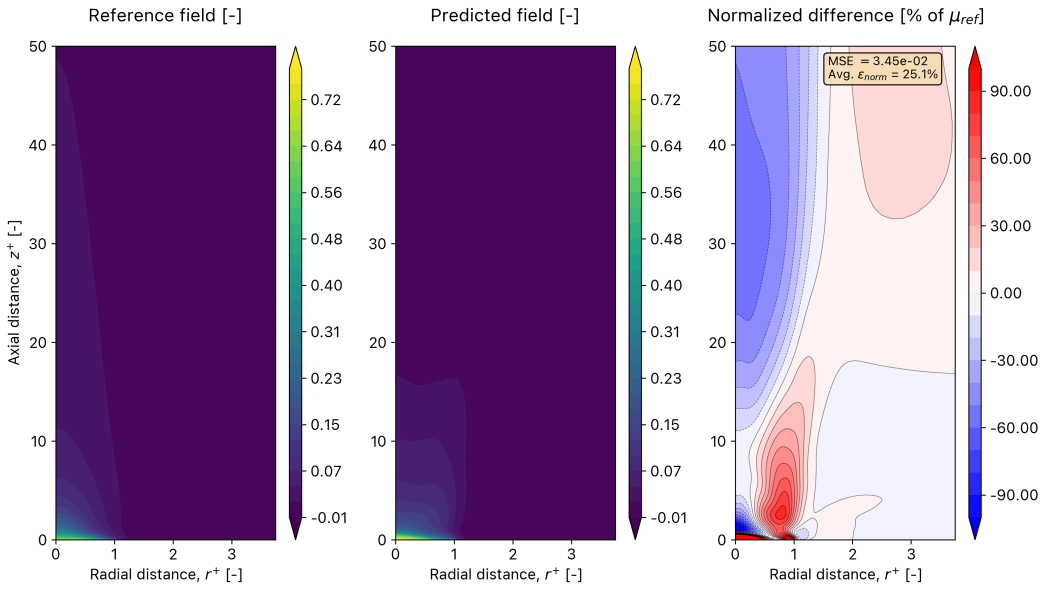


Figure 6.19: Mixture fraction field after improved momentum training (mixture-to-momentum reconstruction).

It can be seen that the modified momentum training phase yields a significant improvement in the mixture fraction reconstruction while maintaining adequate velocity reconstructions. Table 6.7 shows a comparison of the obtained mixture fraction reconstruction accuracy between the multi-stage reconstructions (both with and without the weight adjustment) and the single-stage reconstruction discussed in Section 6.3.1. Table 6.8 shows the same performance metrics for the axial velocity, and Table 6.9 does so for the radial velocity.

Table 6.7: Mixture fraction performance metrics of the mixture-to-momentum multi-stage reconstruction. Comparison against the single-stage reconstruction. The numbers in brackets indicate the order of the training phases. Best performance highlighted in gray.

Error metric (evaluated for ξ)	**Unit**	**Single-stage**	**Multi-stage**		
			Mixture phase (1)	Momentum phase (2A)	Modified momentum (2B)
Standardized global MSE	–	$5.37 \cdot 10^{-1}$	$1.49 \cdot 10^{-1}$	$1.11 \cdot 10^{-1}$	$4.52 \cdot 10^{-2}$
Standardized flame MSE	–	$5.34 \cdot 10^{-1}$	$2.01 \cdot 10^{-1}$	$1.27 \cdot 10^{-1}$	$4.41 \cdot 10^{-2}$
Average absolute global normalized error	% of μ_{ref}	115.3	52.7	50.9	29.6
Average absolute flame normalized error	% of μ_{ref}	116.4	67.2	55.4	28.9

Table 6.8: Axial velocity performance metrics of the mixture-to-momentum multi-stage reconstruction. Comparison against the single-stage reconstruction. The numbers in brackets indicate the order of the training phases. Best performance highlighted in gray.

Error metric (evaluated for u)	**Unit**	**Single-stage**	**Multi-stage**		
			Mixture phase (1)	Momentum phase (2A)	Modified momentum (2B)
Standardized global MSE	–	$1.34 \cdot 10^{-1}$	$6.07 \cdot 10^{-1}$	$8.54 \cdot 10^{-2}$	$1.92 \cdot 10^{-1}$
Standardized flame MSE	–	$1.24 \cdot 10^{-1}$	$5.16 \cdot 10^{-1}$	$7.33 \cdot 10^{-2}$	$1.78 \cdot 10^{-1}$
Average absolute global normalized error	% of μ_{ref}	24.8	58.2	20.3	30.7
Average absolute flame normalized error	% of μ_{ref}	24.0	53.2	18.8	29.5

Table 6.9: Radial velocity performance metrics of the mixture-to-momentum multi-stage reconstruction. Comparison against the single-stage reconstruction. The numbers in brackets indicate the order of the training phases. Best performance highlighted in gray.

Error metric (evaluated for v)	**Unit**	**Single-stage**	**Multi-stage**		
			Mixture phase (1)	Momentum phase (2A)	Modified momentum (2B)
Standardized global MSE	–	$8.55 \cdot 10^{-1}$	$1.24 \cdot 10^{0}$	$1.09 \cdot 10^{-1}$	$8.63 \cdot 10^{-1}$
Standardized flame MSE	–	$1.16 \cdot 10^{0}$	$1.47 \cdot 10^{0}$	$1.13 \cdot 10^{-1}$	$1.13 \cdot 10^{0}$
Average absolute global normalized error	% of μ_{ref}	45.8	71.0	24.5	46.7
Average absolute flame normalized error	% of μ_{ref}	57.0	71.1	24.9	53.9

It can be seen that the multi-stage reconstructions yields superior performance across all four metrics for the three target variables compared to the single-stage reconstruction, reducing the MSE by one order of magnitude both globally and in the flame region. The modified multi-stage reconstruction (2B) yields the best reconstruction accuracy for the mixture fraction field. The best velocity reconstructions for both the axial and radial components are obtained for the standard, unmodified multi-stage reconstruction (2A).

Momentum-to-mixture training

A momentum-to-mixture reconstruction has also been performed to investigate the effect of inverting the order of the introduction of the loss components in a multi-stage reconstruction. The reconstruction performance for the mixture fraction is shown in Table 6.10, where the results are compared against those

of the single-stage reconstruction and the mixture-to-momentum reconstruction. The data shows that the momentum-to-mixture yields superior reconstruction accuracy than the single-stage reconstruction. Nevertheless, the reconstruction accuracy is worse than the one obtained during the mixture-to-momentum reconstruction. The MSE errors are an order of magnitude higher and the average absolute value of the normalized error is increased, both globally and in the flame region. It appears that starting the simulation with the mixture training phase is beneficial for the final reconstruction accuracy of the mixture fraction.

Table 6.10: Mixture fraction performance metrics of the momentum-to-mixture multi-stage reconstruction. Comparison against the single-stage reconstruction. The numbers in brackets indicate the order of the training phases. Best performance highlighted in gray.

Error metric (evaluated for ξ)	**Unit**	**Single-stage**	**Multi-stage**	
			Momentum (1)	Mixture (2)
Standardized global MSE	–	$5.37 \cdot 10^{-1}$	$1.38 \cdot 10^{0}$	$3.60 \cdot 10^{-1}$
Standardized flame MSE	–	$5.34 \cdot 10^{-1}$	$1.25 \cdot 10^{0}$	$2.21 \cdot 10^{-1}$
Average absolute global normalized error	% of μ_{ref}	115.3	190.5	74.4
Average absolute flame normalized error	% of μ_{ref}	116.4	186.3	62.8

The reconstructed mixture fraction field, shown in Figure 6.20, supports this claim. Although the general shape of the field is correctly identified by the PINN, it strongly overestimates the magnitude of the mixture fraction in the plume region. Additionally, the region near the fuel inlet shows erroneous oscillations and a stronger deviation from the reference field than previous runs. Although the reconstruction accuracy shows a significant improvement with respect to the single-stage reconstruction (Figure 6.16), it is inferior to the best mixture-to-momentum reconstruction (Figure 6.19).

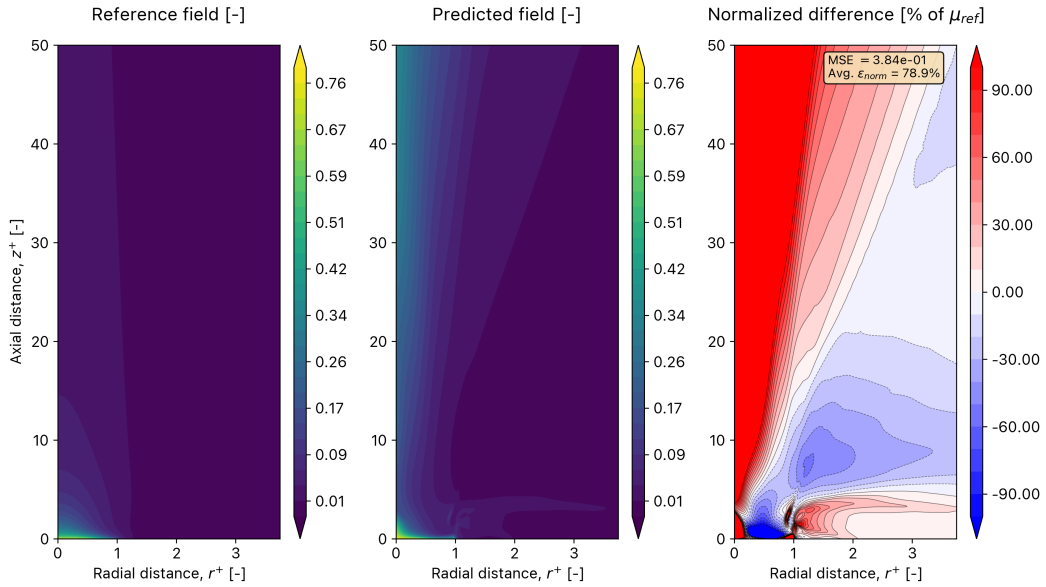


Figure 6.20: Reconstructed mixture fraction field after the momentum-to-mixture multi-stage reconstruction

The reconstruction performance for the axial and radial velocity components is summarized in Table 6.11 and Table 6.12, respectively. The data shows that the momentum-to-mixture reconstruction also yields superior reconstruction accuracy for the two velocity components compared to the single-stage reconstruction. Additionally, the reconstruction accuracy improves during the second phase of the momentum-to-mixture training, when the mixture loss dominates the training process. This indicates that the mixture and momentum losses are compatible — the velocity reconstructions improve when constrained by both the momentum loss and the mixture loss. Compared to the mixture-to-momentum

reconstruction, the axial velocity reconstruction is improved while the radial velocity reconstruction worsens. The axial velocity reconstruction shown in Figure 6.21 is the best reconstruction obtained so far, yielding slight improvements over the baseline results in terms of MSE and average absolute normalized errors, both globally and in the flame region. The reconstructed axial velocity field is shown in Figure 6.21.

Table 6.11: Axial velocity performance metrics of the momentum-to-mixture multi-stage reconstruction. Comparison against the single-stage reconstruction. The numbers in brackets indicate the order of the training phases. Best performance highlighted in gray.

Error metric (evaluated for u)	**Unit**	**Single-stage**	**Multi-stage**	
			Momentum (1)	Mixture (2)
Standardized global MSE	–	$1.34 \cdot 10^{-1}$	$5.71 \cdot 10^{-2}$	$5.88 \cdot 10^{-2}$
Standardized flame MSE	–	$1.24 \cdot 10^{-1}$	$5.78 \cdot 10^{-2}$	$5.09 \cdot 10^{-2}$
Average absolute global normalized error	% of μ_{ref}	24.8	47.8	16.5
Average absolute flame normalized error	% of μ_{ref}	24.0	56.3	15.4

Table 6.12: Axial velocity performance metrics of the momentum-to-mixture multi-stage reconstruction. Comparison against the single-stage reconstruction. The numbers in brackets indicate the order of the training phases. Best performance highlighted in gray.

Error metric (evaluated for v)	**Unit**	**Single-stage**	**Multi-stage**	
			Momentum (1)	Mixture (2)
Standardized global MSE	–	$8.55 \cdot 10^{-1}$	$9.28 \cdot 10^{-1}$	$7.94 \cdot 10^{-1}$
Standardized flame MSE	–	$1.16 \cdot 10^{0}$	$1.22 \cdot 10^{0}$	$1.07 \cdot 10^{0}$
Average absolute global normalized error	% of μ_{ref}	45.8	47.8	40.8
Average absolute flame normalized error	% of μ_{ref}	57.0	56.3	49.8

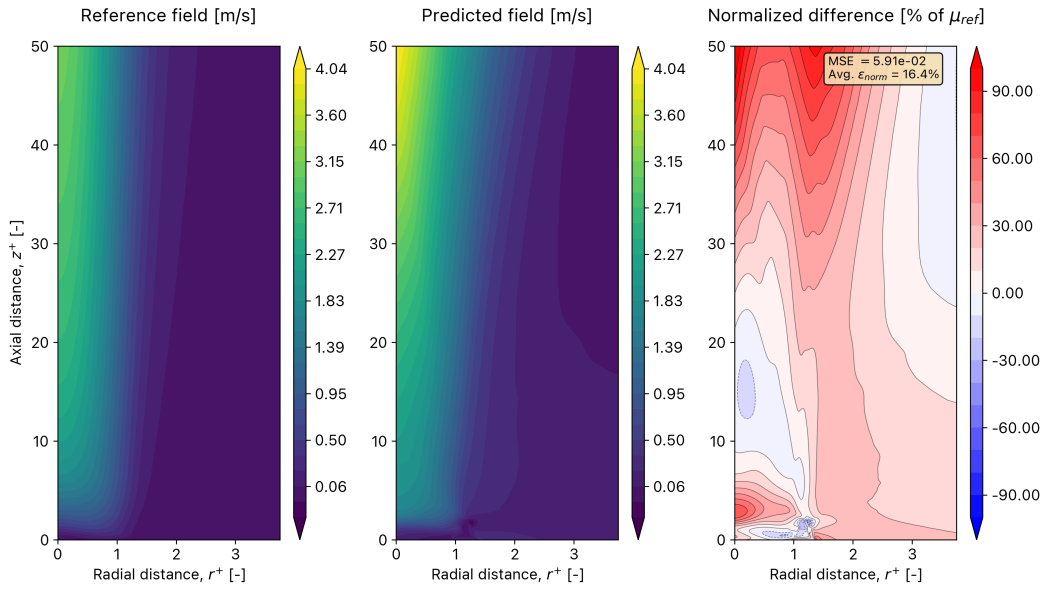


Figure 6.21: Reconstructed axial velocity field after the momentum-to-mixture multi-stage reconstruction

Final remarks

Several conclusions can be drawn from the results presented in this section. Firstly, the results of the reconstructions indicate that the extended PINN model works as expected. The mixture fraction has been successfully added to the PINN algorithm — the reconstruction framework is robust to the addition of new physics-informed loss terms. In addition, the extended PINN model is capable of accurately reconstructing the mixture fraction field. Overall, the reconstruction accuracy for the two velocity components is equivalent to that of the baseline or even slightly better in the case of the axial velocity after momentum-to-mixture training. Additionally, the results indicate that the momentum and mixture losses are compatible and complimentary. The velocity and mixture fraction fields improve simultaneously while their respective losses decrease. This shows compatibility between the formulations of the two physics-informed loss terms.

Moreover, the multi-stage reconstructions yield superior performance over the single-stage reconstructions. The results show that the first term in the multi-stage reconstruction tends to dominate the training process — the momentum-to-mixture run seems to be beneficial for the velocity reconstructions, while the mixture-to-momentum run yields superior reconstruction accuracy for the mixture fraction. This can be understood by recalling that the PINNs have a tendency to get stuck in basins of local minima within their loss landscape due to its complexity. The first stage of a multi-stage reconstruction fixes the direction that the optimizer takes within the loss landscape. Starting a multi-stage reconstruction with mixture training effectively pushes the network towards a state that prioritizes reconstructing the mixture fraction over the velocity. The network then struggles to optimize the velocity reconstructions from that state once the momentum term is added. The opposite is true if the multi-stage reconstructions starts with momentum training; in that case the velocity reconstructions are prioritized instead.

It should be noted that the improved performance of multi-stage reconstructions comes at the cost of added complexity and computational time. Since multiple training stages are performed, the computational time is increased. This is the case because the number of total training iterations is fixed at the start of training, since the behavior of the losses cannot be predicted *a priori*. Nevertheless, most of the successful reconstructions do not need such long training times, and the network's loss reaches its minimum value and converges well before the end of training. In order to properly compare the cost of the single and multi-stage reconstructions the time to convergence of the losses should be considered, instead of the total training time.

The single-stage reconstruction converges after approximately $6,500$ epochs. Let us compare this to the time of convergence of the multi-stage reconstructions. Starting with the mixture-to-momentum reconstruction, the first stage converges after $4,000$ epochs. The second stage does so in $1,250$ epochs for the standard case, and $1,000$ epochs for the run with modified weights. The total time to convergence of the mixture-to-momentum reconstruction is therefore between $5,000$ and $5,250$ epochs, less than that of the single-stage reconstruction. The momentum-to-mixture convergence time is the following. The initial stage converges quickly, in around $2,500$ epochs. The second stage, however, takes $3,500$ epochs to converge. The total convergence time is therefore $6,000$ epochs, the same as the single-stage reconstruction. Hence, we can conclude that although the total training time is significantly longer for the multi-stage reconstructions, they actually yield faster or equivalent convergence times. This indicates that splitting the training into two separate phases and readjusting the weights in between is beneficial for the reconstruction algorithm. This is agreement with the work of Loshchilov and Hutter [102], introduced in Chapter 4 (Section 4.4), which demonstrated that performing warm restarts improves the performance of stochastic gradient descent optimization algorithms when training deep neural networks.

As a final note, it should be stated that beyond longer training times the multi-stage reconstructions also require more user oversight than the single-stage reconstruction. In between the two training stages the results of the model need to be processed, a new model needs to be constructed, and the weights of the model readjusted based on the difference between the data and physics losses. Some of these steps could be automated — the second model could be constructed from the beginning and the weight selection process can be automated. This, however, has not been done in this thesis, but the reconstruction can easily be modified to include these extensions in the future to reduce the amount of user oversight needed to run multi-stage reconstructions.

6.4. Mixture fraction as a data source

In this section the use of the mixture fraction as a data source for the velocity reconstructions is explored. The goal is not to reconstruct the mixture fraction field, but instead to assess whether including it in the data loss of the PINN yields any improvements in the velocity reconstructions. The mixture fraction is treated like the standard training variables of previous reconstructions (density, pressure, and temperature), using it to guide the velocity reconstruction process. The data loss function is adjusted by adding a term to compute the mean squared error between the estimated and reference mixture fraction fields. To ensure a fair comparison with previous reconstructions, a maximum of three flow variables are included in each attempt. Consequently, with the addition of the mixture fraction, one of the other variables must be omitted. Three series of tests have been conducted. In the first, the mixture fraction replaces the temperature in the data loss. In the second, it replaces the density. And in the third, it replaces the pressure.

Mixture fraction as a substitute for temperature

In the first test the mixture fraction replaces the temperature as a data source in the reconstruction. The data loss is therefore made up of the differences between the DNS data and the estimated fields of density, pressure, and mixture fraction. The performance metrics of this reconstruction are summarized and compared to the baseline reconstruction in Table 6.13.

Table 6.13: Summary of reconstruction performance when substituting temperature for the mixture fraction. Comparison against the baseline reconstruction. Best performance highlighted in gray.

Error metric	**Unit**	**Axial velocity** (u)		**Radial velocity** (v)	
		Baseline	$T \longrightarrow \xi$	Baseline	$T \longrightarrow \xi$
Standardized global MSE	–	$7.44 \cdot 10^{-2}$	$7.07 \cdot 10^{-2}$	$6.75 \cdot 10^{-1}$	$9.92 \cdot 10^{-1}$
Standardized flame MSE	–	$5.27 \cdot 10^{-2}$	$3.65 \cdot 10^{-2}$	$1.24 \cdot 10^{0}$	$1.30 \cdot 10^{0}$
Average absolute global normalized error	% of μ_{ref}	20.9	18.2	42.7	50.8
Average absolute flame normalized error	% of μ_{ref}	18.0	14.0	69.7	70.2

It can be seen that substituting the mixture fraction for the temperature results in an improvement in the axial reconstruction accuracy with respect to the baseline. This improvement can be seen both globally and within the flame region. The reconstructed axial velocity field is shown in Figure 6.22. Comparing it with the baseline axial velocity reconstruction shown in Figure 6.7 shows a small but noticeable improvement in the reconstruction quality, specially near the base of the flame. The bottom right boundary of the plume region ($r^+ \approx 1$, $z^+ \approx 10$) is resolved significantly better by this reconstruction. The general shape of the plume is reconstructed more accurately in that region, and the underestimation of the axial velocity magnitude in the plume is reduced.

The reconstructed radial velocity field is shown in Figure 6.23. It can be seen that the reconstruction quality is poor. The inability of the PINN to reconstruct the air entrainment region, present in previous reconstructions, is magnified compared to the baseline. Additionally, the PINN incorrectly estimates the radial velocity to be approximately zero near the bottom boundary to the right of the flame. This is in agreement with the data shown in Table 6.13, which shows a drop in performance across all metrics for the radial velocity reconstruction compared to the baseline.

It can be concluded that that the reconstruction algorithm is affected by the replacement of temperature in the data loss with the mixture fraction. The axial velocity reconstruction appears to benefit from the presence of the mixture fraction data combined with the added constraint provided by the mixture loss. On the other hand, the radial velocity suffers a clear reduction in its reconstruction accuracy compared to the baseline. Just like the momentum loss, the mixture loss seems to prioritize improvements in the axial velocity component over the radial one, and the air entrainment region remains problematic for the PINN to resolve.

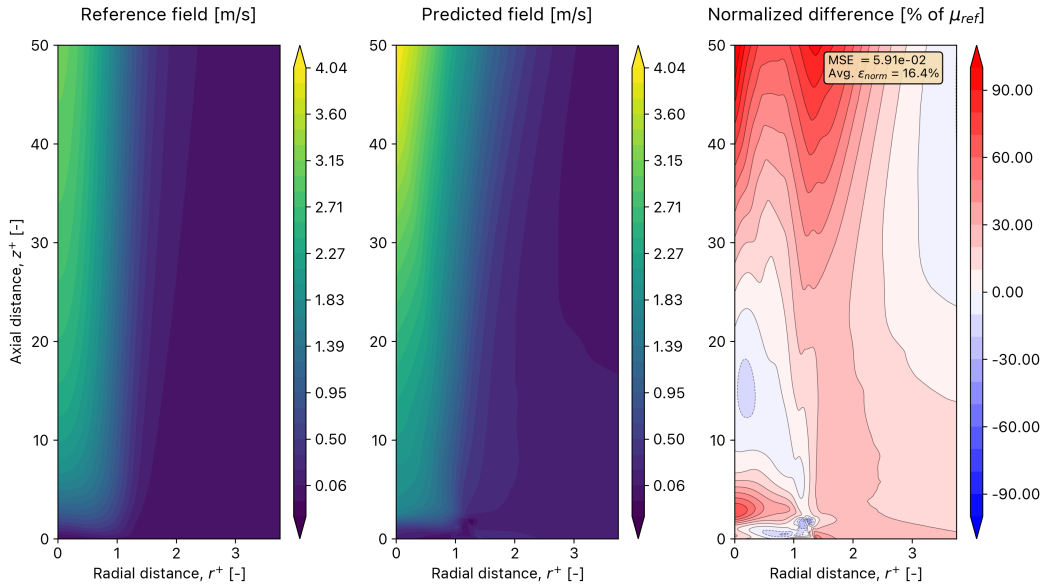


Figure 6.22: Reconstructed axial velocity field when substituting temperature for the mixture fraction

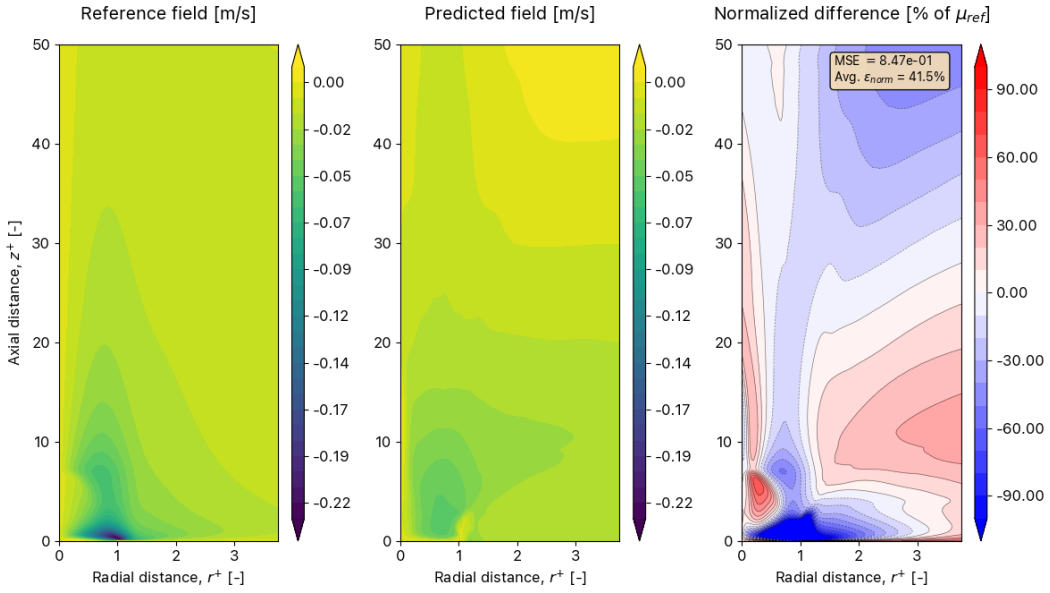
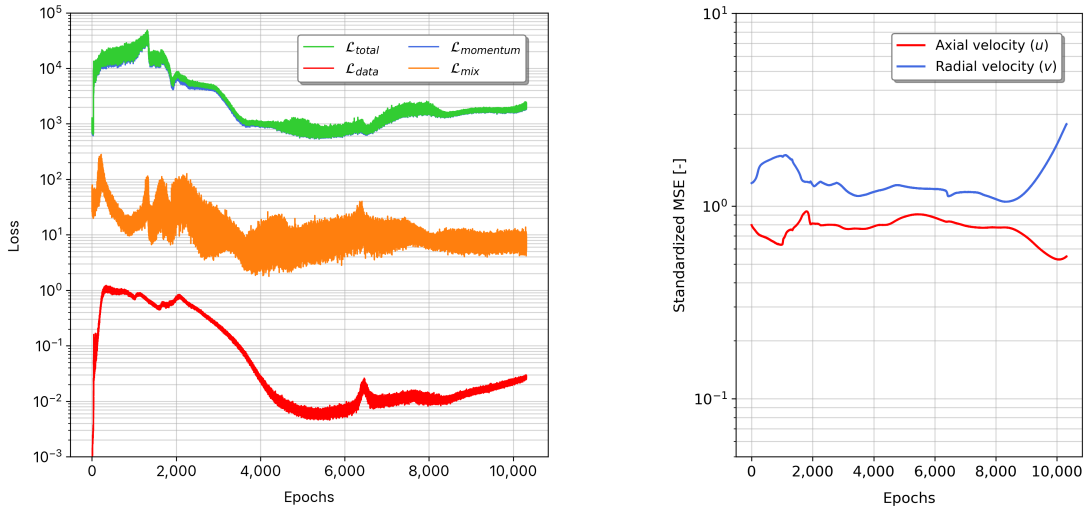


Figure 6.23: Reconstructed radial velocity field when substituting temperature for the mixture fraction

Mixture fraction as a substitute for density

A separate test has been conducted where the mixture fraction is used as a substitute for the density in the data loss. The performance of the network during the physics-informed training phase of this reconstruction attempt is shown in Figure 6.24. The evolution of the losses, shown in Figure 6.24a, is indicative of a suboptimal reconstruction. The mixture loss decreases during training, but only does so by an order of magnitude. The momentum loss increases significantly at the start of training, settling at a value one order of magnitude higher than its initial value. It then decreases back to its initial state and remains there — the PINN is unable to minimize the momentum loss during this reconstruction attempt. The poor quality of this reconstruction can also be anticipated from the evolution of the reconstruction errors, shown in Figure 6.24b. Although the axial velocity error decreases during training, it only does so marginally. The radial velocity error remains approximately constant for most of the training process,

experiencing a sudden increase in the final 2,000 epochs.



(a) Evolution of losses

(b) Evolution of reconstruction errors

Figure 6.24: Training performance of the PINN when substituting density for the mixture fraction

The reconstructed velocity fields are shown in Figure 6.25 and Figure 6.26. It can be seen that, as expected from the training performance, this reconstruction attempt is clearly unsuccessful. Neither the axial nor radial velocity can be accurately represented. Surprisingly, the reconstruction now overestimates the magnitude of the radial velocity in the air entrainment region. Whereas in previous runs the PINN was unable to accurately resolve the air entrainment region and significantly underestimated the magnitude of the radial velocity there, the opposite is true in this case. Nevertheless, the reconstruction for both velocity components is quite poor. The axial velocity is only reconstructed accurately in the bottom region of the plume. The top region of the plume is underestimated, while the far field region is overestimated. The PINN is not able to capture the general shape of the axial velocity field and fails to adequately characterize the structures in the flow field. The performance metrics of this reconstruction are shown in Table 6.14. The data is in agreement with the analysis of the training performance and that of the reconstructed fields. For both the axial and radial velocity the baseline reconstruct provides superior reconstruction accuracy across all metrics.

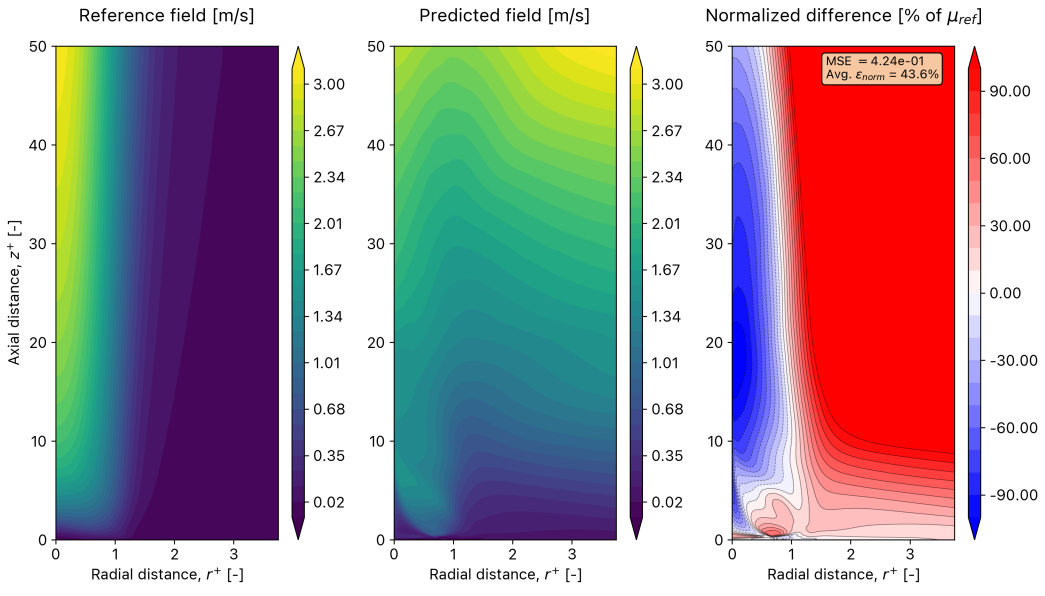


Figure 6.25: Reconstructed axial velocity field when substituting density for the mixture fraction

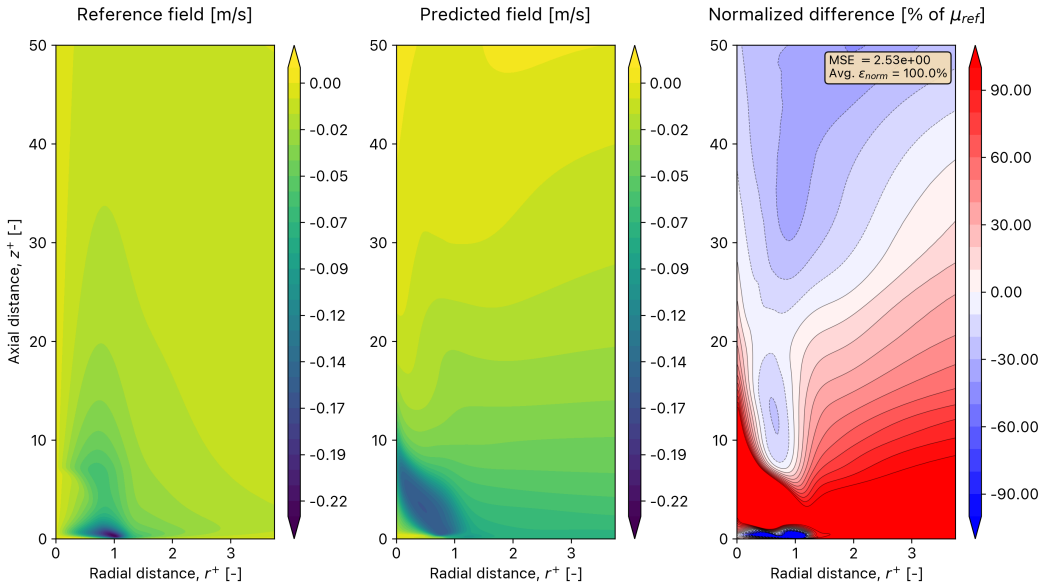


Figure 6.26: Reconstructed radial velocity field when substituting density for the mixture fraction

Table 6.14: Summary of reconstruction performance when substituting density for the mixture fraction. Comparison against the baseline reconstruction. Best performance highlighted in gray.

Error metric	**Unit**	**Axial velocity** (u)		**Radial velocity** (v)	
		Baseline	$\rho \longrightarrow \xi$	Baseline	$\rho \longrightarrow \xi$
Standardized global MSE	–	$7.44 \cdot 10^{-2}$	$5.34 \cdot 10^{-1}$	$6.75 \cdot 10^{-1}$	$2.46 \cdot 10^{0}$
Standardized flame MSE	–	$5.27 \cdot 10^{-2}$	$2.25 \cdot 10^{-1}$	$1.24 \cdot 10^{0}$	$3.17 \cdot 10^{0}$
Average absolute global normalized error	% of μ_{ref}	20.9	47.7	42.7	100.4
Average absolute flame normalized error	% of μ_{ref}	18.0	33.4	69.7	120.6

These results indicate that using the mixture fraction as a substitute for the density is much more problematic than substituting it for the temperature. The PINN is unable to accurately reconstruct the velocity fields when density data is not provided, even in the presence of mixture fraction data and with the use of the mixture loss as an additional constraint in the training process. On the other hand, removing the temperature as a data source seems to have less of an effect on the reconstruction — substituting it for the mixture fraction data yields an improvement in the axial velocity reconstructions compared to the baseline, albeit at the cost of a reduction in the reconstruction accuracy of the radial velocity.

Mixture fraction as a substitute for pressure

Removing the pressure from the data loss is even more problematic than removing the density. The reconstructed velocity fields, shown in Figure 6.27 and Figure 6.28, indicate that the PINN is completely unable to reconstruct the velocity fields. The PINN erroneously predicts both fields to be approximately constant. The axial velocity shows the faint outline of a flow structure that is reminiscent of the flame shape seen in the reference temperature field. The shape and magnitude of the plume, the key flow feature in the axial velocity field, is completely misrepresented by the PINN. The radial velocity is estimated to be constant and equal to zero everywhere in the domain, with the exception of the plume region, where it takes on a small but negative constant value. It is clear that without pressure data the PINN is unable to accurately reconstruct the two velocity fields. Both fields assume a value equal to its true value in the far field, but beyond that, they are unrecognizable from their reference fields.

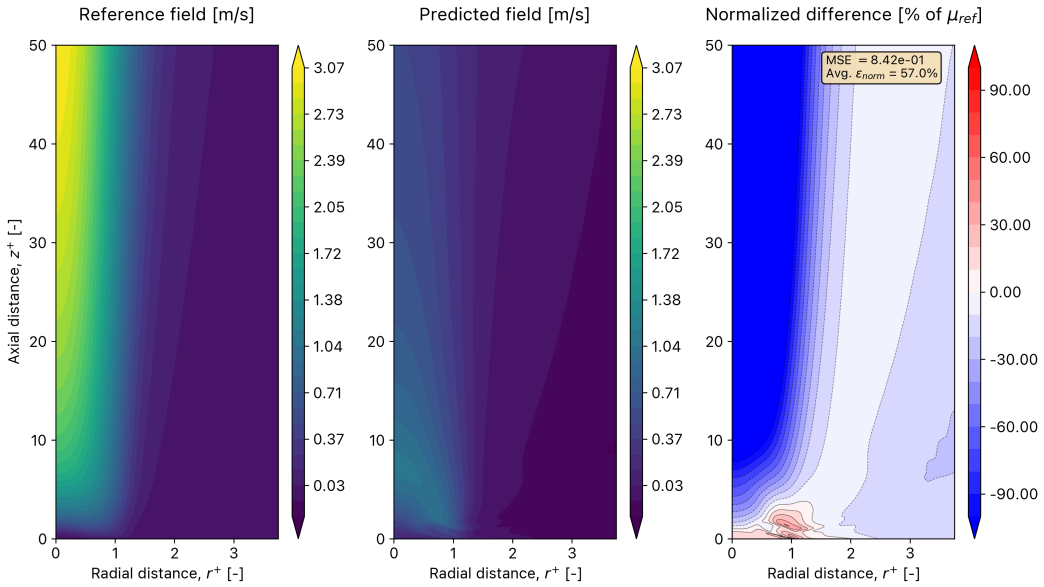


Figure 6.27: Reconstructed axial velocity field when substituting pressure for the mixture fraction

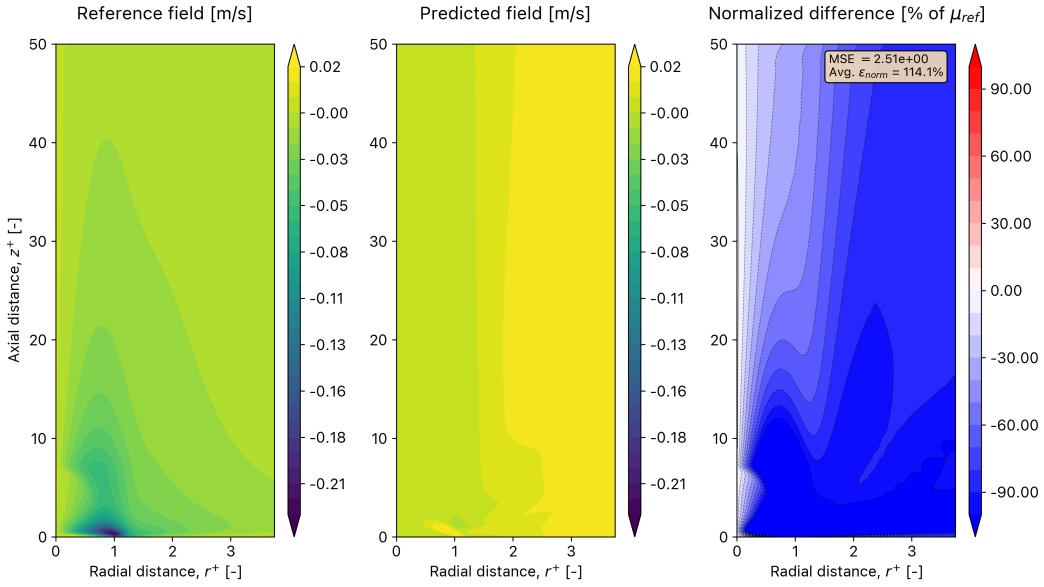


Figure 6.28: Reconstructed radial velocity field when substituting pressure for the mixture fraction

The results of this section show that pressure and density data are significantly more important than the temperature data in order to obtain successful reconstructions. This is to be expected considering the feedback that exists between these quantities and the velocity components in the momentum loss. The density appears explicitly in the continuity and Navier-Stokes equations, as does the pressure gradient. The direct feedback between these quantities and the velocity fields makes it difficult for the PINN to estimate the latter in the absence of the former. The temperature, on the other hand, only appears in the computation of the viscosity, which is used to evaluate the components of the stress tensor. This weaker feedback to the velocity fields makes the network more resilient to the removal of temperature data. In that case, the addition of the mixture fraction improves the axial velocity reconstruction, indicating that the additional constraint through the mixture loss is sufficient to offset the error in the computation of the stress tensor due to a lack of temperature data. Even then, despite the fact that the

axial reconstruction improves with respect to the baseline, the radial velocity experiences a significant decrease in reconstruction accuracy, specially within the air entrainment region.

Overall, using the mixture fraction as a data source is not sufficient to offset the absence of pressure and density data, and even removing the temperature data has a clear impact on the quality of the velocity reconstructions. It appears that in order to find minimal data solutions to the reconstruction problem that do not rely on density and/or pressure data, the lack of information must be compensated by another mechanism to allow the PINN to accurately reconstruct the velocity fields. Previous work has identified the advantages of initializing the missing fields using reasonable guesses for the performance of the network instead of removing them altogether [26, 93] . In the following section the mixture fraction is used to construct improved initializations to overcome the limitations presented here, exploring new minimal data solutions that exploit the inclusion of the mixture fraction in the reconstruction algorithm.

6.5. Minimal data solutions using the mixture fraction

Based on the difficulties of the PINN to reconstruct the velocity fields in the absence of pressure, density, and temperature data, we now turn our attention to the use of the mixture fraction to construct improved initializations for the missing quantities. We wish to determine if the mixture fraction can be a valuable tool in the search for new minimal data solutions, which, based on the results of the previous section, are difficult to perform if the lack of data is not compensated through another mechanism to guide the PINN during the reconstruction process.

Minimal data solutions are defined as successful reconstructions that rely on the fewest possible number of training variables to guide the PINN . Such solutions are highly desirable from a practical standpoint — the less information we need to give the PINN algorithm to obtain successful reconstructions, the better. This relaxes the constraints on the amount and type of data that needs to be collected to perform the reconstructions, paving the way for the application of this technique to more challenging cases with practical relevance for combustion researchers. Minimal data solutions are particularly interesting in the present study due to the fact that some of the training variables used until now (namely pressure, and to a lesser extent, density) are difficult to measure experimentally.

Additionally, we have seen in Section 6.4 that removing these quantities is too big of a challenge for the PINNs to overcome. If the PINN is only provided data on a few flow states, for example density and temperature, while leaving the pressure field unconstrained, it is likely that the PINN will minimize its momentum loss by finding trivial solutions where the unconstrained flow fields are driven towards constant (or even zero) values. This is a well known training pathology of PINNs, as discussed in Chapter 4. We would like to find solutions where we can remove some of these flow variables and still obtain adequate velocity reconstructions. Is there a way to use the mixture fraction to improve our chances of finding these minimal data solutions?

The effect of the mixture fraction in this process is twofold. On one hand, the mixture fraction is used to construct new and improved initializations for the missing flow fields. This alone significantly affects the quality and type of possible minimal data solutions that can be obtained. On the other hand, the presence of the mixture loss acts as an additional constraint to guide the velocity reconstruction. The mixture loss is particularly powerful in this case, since the mixture fraction is included as part of the data loss in these reconstructions and therefore the network's estimate of the mixture fraction field is very accurate. If the different loss components are weighted adequately, following the guidelines derived in Section 6.1, the PINN is forced to respect the mixture fraction field due to the presence of the data loss. Consequently, reductions in the mixture loss can only be obtained through improvements in the velocity fields.

Several minimal data reconstruction attempts have been performed. First, only the density is removed, constructing an improved guess using the mixture fraction to help offset the reduction in training data (Section 6.5.1) . Then, both density and temperature are removed, guessing both of them through the mixture fraction (Section 6.5.2). The task of removing the pressure field is tackled afterwards. As discussed previously, reconstructions without the pressure field are challenging for the PINN due to the crucial role of the pressure gradient in the momentum loss formulation. Pressure and

density are removed first (Section 6.5.3). Finally, all three fields (pressure, density, and temperature) are removed, providing initial guesses for these fields with the help of the mixture fraction. This is the most challenging reconstruction attempt, where a PINN is tasked with reconstructing the velocity fields of the flame using only the data on the mixture fraction field (Section 6.5.5).

6.5.1. Removing density

The challenge of removing the density field is tackled first. An improved initialization of the density field is constructed using the ideal gas law with a variable specific gas constant (R), as opposed to assuming a constant value. Assuming a constant value for the specific gas constant of $R = R_{\text{air}}$ is only valid in regions of the domain where the mixture fraction is low, indicating a high oxidizer mass fraction. Within the flame, where the mixture fraction is high, the composition of the fluid is made up of mostly fuel and reaction products. Using a constant value of R_{air} in this region yields an incorrect guess for the density field.

To mitigate this, the reconstruction algorithm uses the mixture fraction field to compute a guess for the local value of R everywhere in the domain by evaluating the composition of the gas at each point. The procedure to obtain this estimation is discussed in Section 5.5. The result is a piecewise linear function of the form $R = f(\xi)$. The resulting specific gas constant field using the Burke-Schumann approximation for the composition is shown in Figure 6.29 (middle left). It can be seen that the use of the mixture fraction significantly improves the estimate for the specific gas constant within the flame compared to the assumption of constant R, as expected. This definition of $R(\xi)$ is used to construct a mixture fraction enhanced guess for the density field based on the initial estimate of the mixture fraction field. This guess is based on the ideal gas law. Based on the network's estimates for pressure, temperature, and mixture fraction, the initialization for the density field is calculated as:

$$\rho_{\text{guess}} = \frac{p_{\text{est}}}{R_{\text{est}} \cdot T_{\text{est}}}, \quad \text{with} \quad R_{\text{est}} = f\left(\xi_{\text{est}}\right) \tag{6.1}$$

where p_{est}, T_{est}, and ξ_{est} are the PINN estimates for the pressure, temperature, and mixture fraction, respectively. A comparison between the enhanced density guess and the reference data is shown on the right side of Figure 6.29. It can be seen that the guess resulting from this enhanced mixture fraction method is very close to the reference data.

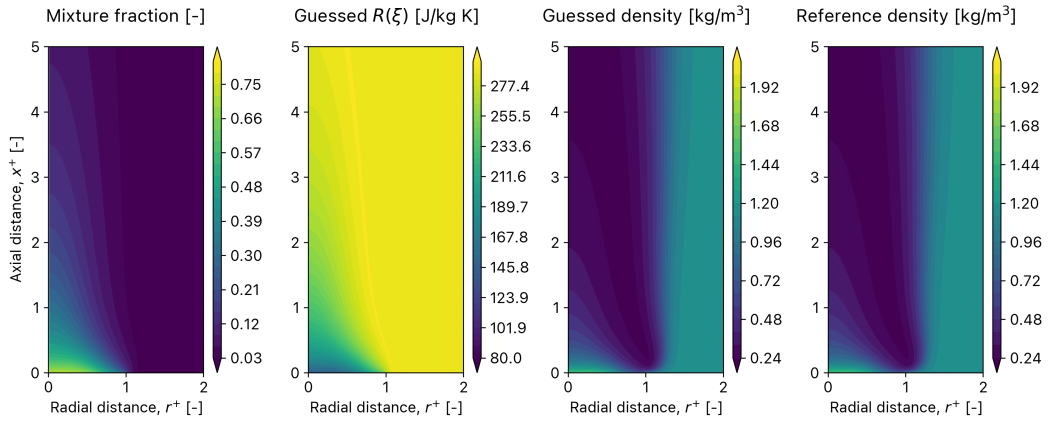


Figure 6.29: Estimating the local specific gas constant from the mixture fraction. The mixture data (left) is used to estimate the local specific gas constant (middle left), which is in turn used to guess the density field (middle right). The guessed density field matches the reference data (right) accurately.

The mixture fraction enhanced density guess is used in the reconstruction process as follows. First, as usual, the PINN is trained on the data loss, which in this case only contains information about pressure, temperature, and mixture fraction. After the data training is complete an additional loss term that quantifies the difference between the estimated density field and the guessed density field is added to the

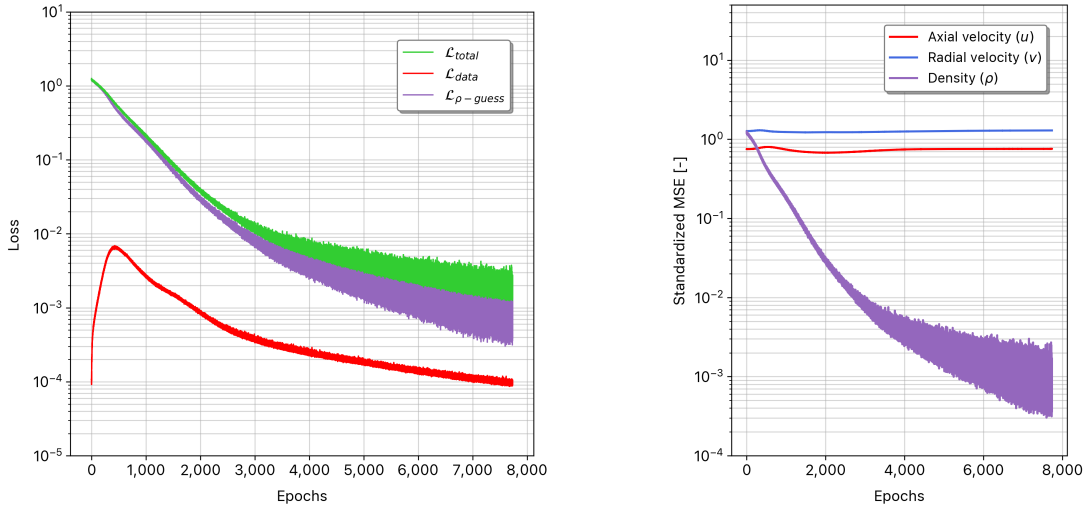
network's loss function. This density loss term is computed as follows:

$$\begin{aligned} \Delta\rho_{\text{guess}} &= \rho_{\text{est}} - \rho_{\text{guess}} = \rho_{\text{est}} - \frac{p_{\text{est}}}{R\left(\xi_{\text{est}}\right) \cdot T_{\text{est}}} \\ \mathcal{L}_{\rho\text{-guess}} &= \text{MSE}\left(\Delta\rho\right) \end{aligned} \tag{6.2}$$

The guess for the density field (ρ_{guess}) is computed once, following the data training phase and prior to physics informed training. Since the pressure, temperature, and mixture fraction data are included in the data loss of the network, their estimated values will be very close to the reference data, yielding an accurate initial guess for the density field. The guess is computed once and remains unchanged during the training process. The reason for performing this computation once instead of at every training iteration is to prevent oscillations in the data fields (which are likely to happen during the early stages of physics-informed training) from corrupting the density guess, introducing errors within it and complicating the training process. The density guess is instead computed once at the beginning of physics-informed training, when the data loss is minimal. It then serves as a constant reference point to provide the network with an idea of what a sensible solution for the density field looks like.

After data training the PINN is trained on the density guess loss while maintaining the data loss on pressure and temperature. The performance of the model during this training phase is shown in Figure 6.30. It can be seen that the model successfully minimizes the density loss while also maintaining a low value of the data loss. The data loss initially increases as the network adjusts to the presence of the density loss term, but then decreases back to its initial value. If the network had computed the density guess at every iteration, this initial increase in the data loss would have likely destabilized the training process. As the data fields worsen during the first 1,000 epochs the quality of the density guess would have worsened too, creating a positive feedback loop where the density guess loss would hinder the training process by validating the network's incorrect guesses on the data fields.

The reconstruction error for density (Figure 6.30b) quickly drops to 10^{-3} as the PINN minimizes the density guess loss, verifying that the mixture enhanced guess yields a good estimate of the reference data. The velocity reconstruction errors remain constant. This is expected, since the network has no physics-informed mechanism to improve the velocity fields during this training phase. By the end of this training phase the model correctly estimates the pressure, temperature, and mixture fraction through the data loss and has an appropriate initial guess for the density field thanks to the minimized density guess loss.

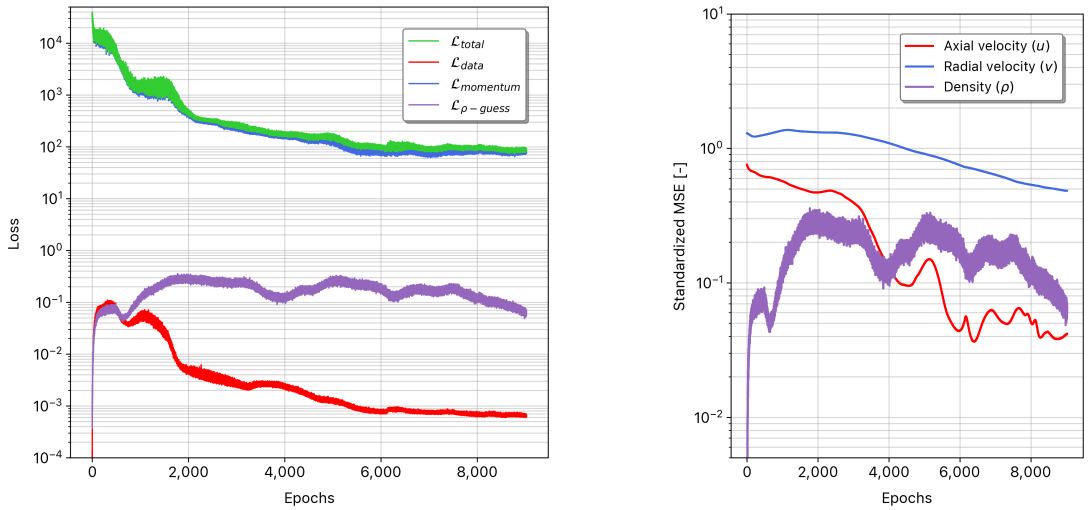


(a) Evolution of losses

(b) Evolution of reconstruction errors

Figure 6.30: Training performance of the PINN during density training

The model is then extended to incorporate the physics-informed loss terms. Based on the results of Section 6.3 a multi-stage mixture-to-momentum reconstruction is selected to reconstruct the velocity fields. The performance of the network during the first stage of physics-informed training, where only the momentum loss is incorporated into the network, is shown in Figure 6.31.



(a) Evolution of losses

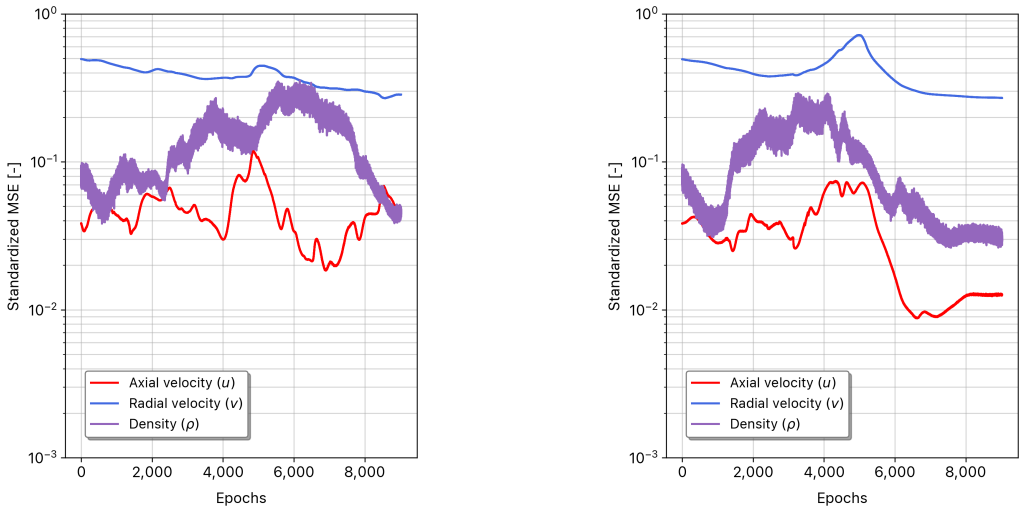
(b) Evolution of reconstruction errors

Figure 6.31: Training performance of the PINN during momentum training in the absence of density data

It can be seen that the performance of the network is indicative of a successful reconstruction. The momentum loss decreases by two orders of magnitude while maintaining a sufficiently low value of the data loss. The density loss increases at the beginning of training before settling at a value of 10^{-1}. This fluctuations can also be observed in the evolution of the density reconstruction error in Figure 6.31b. The density error oscillates slightly during training but remains sufficiently low to ensure accurate density reconstructions. The velocity errors clearly decrease during training as well, indicating that the PINN is improving its estimation of the velocity components as a consequence of the physics-informed training. As usual, the radial velocity error is higher than the axial error and is more difficult for the PINN to minimize.

After performing the momentum training, the PINN undergoes the second leg of the momentum-to-mixture multi-stage reconstruction, where the mixture loss is added alongside the momentum loss. The evolution of the reconstruction errors during this training phase is shown in Figure 6.32a. The density error experiences some oscillations during this training phase. By the end of training, however, the error has decreased by an additional half an order of magnitude. The axial velocity error, which also starts at a low value, also experiences oscillations but remains low enough to indicate that a successful axial velocity reconstruction has been obtained. The radial velocity error decreases to a value of $1 \cdot 10^{-1}$, one order of magnitude lower than its value at the beginning of the multi-stage reconstruction.

An alternative second leg of training is performed where the PINN is trained only on the momentum loss for a second time — the resulting multi-stage reconstruction is hence composed of two consecutive phases of momentum training, stopped halfway to reinitialize the optimizer and readjust the loss weights. The evolution errors for this run are shown in Figure 6.32b. In general the errors behave in a similar manner, with the exception of the axial velocity error, which decreases even further in this second case reaching a value of approximately 10^{-2} by the end of training.



(a) Momentum-to-mixture training

(b) Double momentum training

Figure 6.32: Evolution of the reconstruction errors during the final phases of two multi-stage reconstructions in the absence of density data.

To conclude this analysis, a comparison of the reconstruction accuracies of the two velocity components using these two different types of multi-stage reconstructions is presented in Table 6.15. It can be seen that the PINN is not only capable of matching the baseline performance in the absence of density data, but actually outperforms the baseline reconstructions for both the axial and the radial velocity components across all metrics. Both multi-stage reconstruction yield superior performance than the baseline. This indicates that the mixture enhanced guess combined with a multi-stage reconstruction is a valuable tool to perform minimal data solutions.

Table 6.15: Summary of reconstruction performance in the absence of density data. Comparison against the baseline reconstruction. Best performance highlighted in gray.

Error metric	**Unit**	**Axial velocity** (u)			**Radial velocity** (v)		
		Baseline	Momentum to mixture	Double momentum	Baseline	Momentum to mixture	Double momentum
Standardized global MSE	–	$7.44 \cdot 10^{-2}$	$4.76 \cdot 10^{-2}$	$1.12 \cdot 10^{-2}$	$6.75 \cdot 10^{-1}$	$2.85 \cdot 10^{-1}$	$2.79 \cdot 10^{-1}$
Standardized flame MSE	–	$5.27 \cdot 10^{-2}$	$3.25 \cdot 10^{-2}$	$9.02 \cdot 10^{-3}$	$1.24 \cdot 10^{0}$	$3.38 \cdot 10^{-1}$	$3.48 \cdot 10^{-1}$
Average absolute global normalized error	% of μ_{ref}	20.9	16.3	7.5	42.7	28.6	24.5
Average absolute flame normalized error	% of μ_{ref}	18.0	13.9	7.0	69.7	30.8	27.2

Comparing the two multi-stage reconstructions it can be seen that the double momentum approach is superior to the momentum-to-mixture approach. For the axial velocity component the double momentum approach yields a significant improvement with respect to the baseline. The MSEs are reduced significantly, both globally and within the flame region. This conclusion is confirmed by visually inspecting the resulting axial velocity field at the end of the double momentum reconstruction, shown in Figure 6.33. The PINN is now capable of reconstructing the lower plume region with increased accuracy, removing the oscillations present in that region in the baseline case. Although the magnitude of the axial velocity in the upper plume region remains slightly overestimated by the PINN compared to the reference data, this overestimation is reduced by the double momentum reconstruction.

The double momentum approach also yields significant improvements with respect to the baseline in the radial velocity reconstruction, as does the momentum-to-mixture approach. The best performing radial velocity reconstruction obtained with these tests is shown in Figure 6.34. Visual inspection of the reconstructed radial velocity fields is in agreement with the data of Table 6.15 and the decrease in the radial reconstruction error shown in Figure 6.32b. The quality of the radial velocity reconstruction is significantly superior to the baseline.

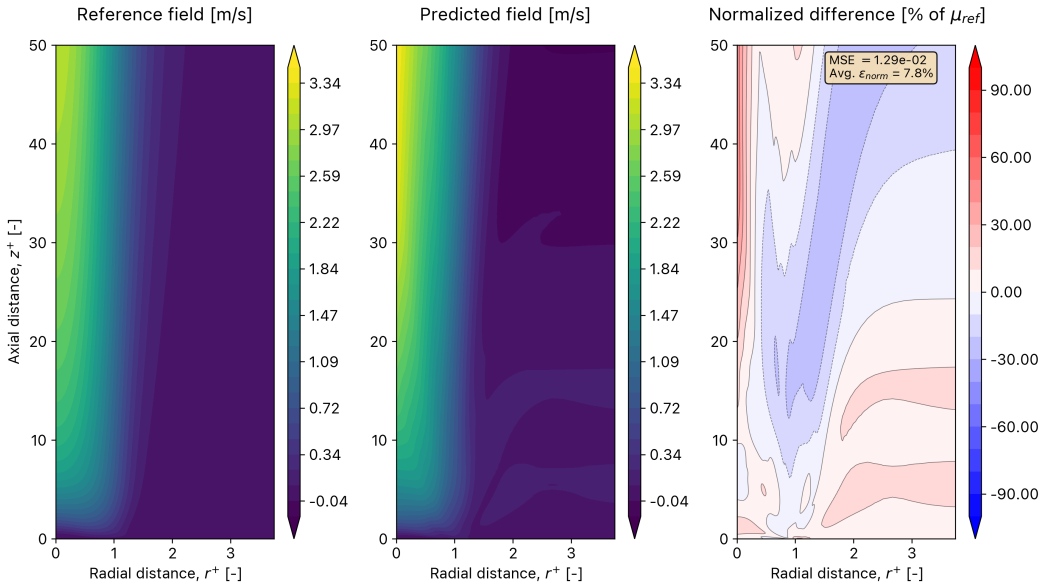


Figure 6.33: Reconstructed axial velocity in the absence of density data using double-momentum training

Particularly surprising is the ability of the PINN to resolve the air entrainment region. This is significant due to the fact that, up until now, the reconstruction of this region (and of the radial velocity as a whole) had proven to be a difficult task for the PINN. The improvement in the air entrainment region is also visible, albeit to a lesser extent, in the momentum-to-mixture reconstruction (not pictured for the sake of brevity). It appears that the substitution of the density data by a guess based loss is beneficial, yielding improved radial velocity reconstructions across all metrics for both the double momentum and the momentum-to-mixture reconstructions.

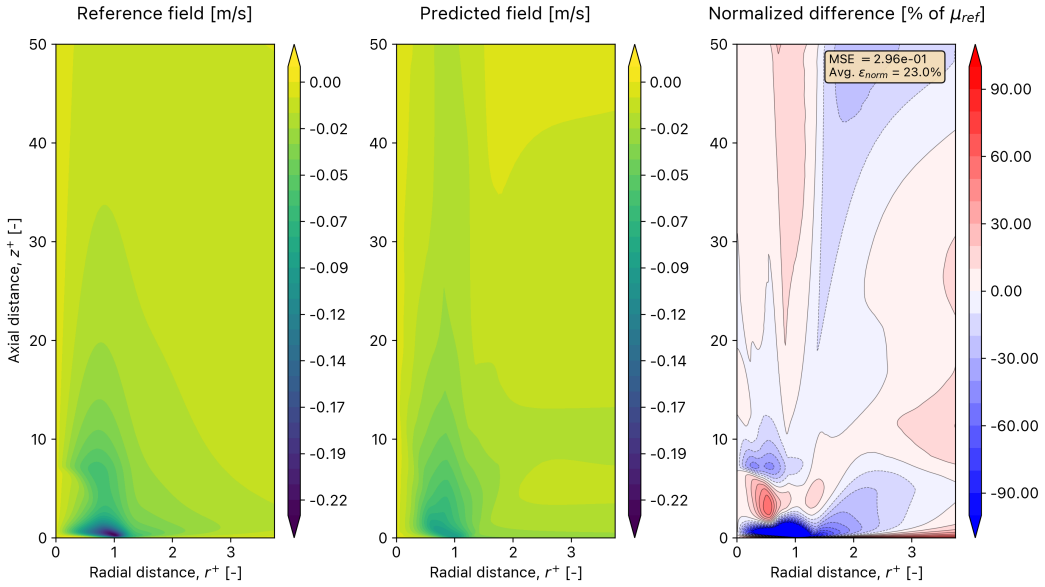


Figure 6.34: Reconstructed radial velocity in the absence of density data using double-momentum training

It is hypothesized that the reason for this improvement in performance is related to a relaxation of the constraint imposed on the density gradients due to the absence of density data in the loss function. It appears that removing the density data and substituting it with a reasonable initial guess in combination with a guess-based loss may help alleviate the occurrence of spurious oscillations on the density gradient

which in turn improves the velocity reconstructions. In a way, the substitution of data by mixture fraction enhanced guesses appears to give the PINN more freedom in its optimization of the momentum loss. The results shown in this section hint towards the fact that providing an adequate initialization in conjunction with an appropriate guess-based loss may be more optimal, in a global sense over the entire training period, than providing data on a flow quantity itself.

An argument can be made that a similar relaxation could be obtained by simply reducing the weight of the data loss during training. Although this is in principle true, practically speaking it introduces a new set of problems that prevent the PINN from obtaining adequate reconstructions. As discussed previously, PINNs have a tendency to drive the reconstructed quantities towards near zero values in an attempt to quickly minimize the physics-informed losses, resulting in trivial solutions. The networks get stuck in local minima within their loss landscapes that they cannot escape from. This tendency to arrive at trivial solutions when given too much flexibility prohibits the application of significant reductions to the data loss term. In this context, it appears that mixture fraction enhanced guess-based loss terms may provide a solution to circumvent this well-known training pathology, giving the network sufficient freedom to improve the computation of their gradients while preventing them from diverging to trivial solutions. The key difference between the two approaches is that, while the standard momentum loss may be incentivized to find zero-velocity solutions that quickly minimize the residuals of the governing equations, this is not possible using the guess-based losses, since they are measuring the absolute difference between a field and its target state directly.

Overall, the results presented so far show that the extended PINN model is capable of reconstructing the velocity fields in the absence of density data, even surpassing the baseline performance, as shown in Table 6.15. These results provide motivation to construct additional mixture fraction enhanced guesses in an attempt to reconstruct the velocity fields in the absence of more data fields beyond density. This possibility is explored in the following sections.

6.5.2. Removing density and temperature

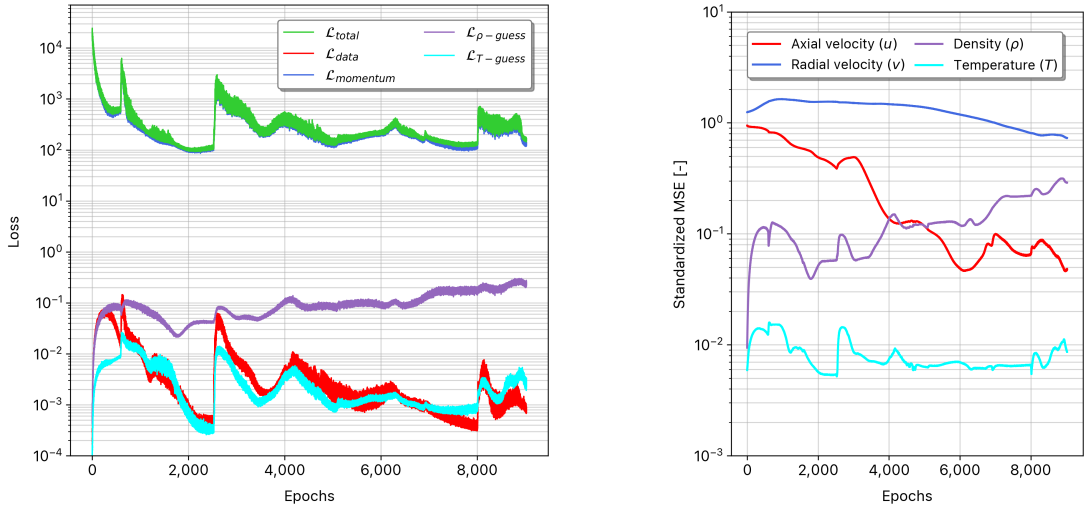
In the next series of tests both density and temperature are removed as data sources for the reconstruction process. The PINN therefore relies only on pressure and mixture fraction data to guide the velocity reconstruction. The density guess loss remains unchanged, using the formulation of Equation 6.2. The Burke-Schumann approximation is used to estimate the temperature field from the mixture fraction, as explained in Section 5.5. The result is a relation of the form $T = f(\xi)$. This relation is used to guess the temperature field at any given instant using the PINN's current estimate of the mixture fraction field. As done for the density, a temperature guess loss is formulated to measure and minimize the difference between the current temperature estimate and the calculated guess:

$$\begin{aligned} \Delta T_{\text{guess}} &= T_{\text{est}} - T_{\text{guess}} = T_{\text{est}} - T(\xi_{\text{est}}) \\ \mathcal{L}_{T\text{-guess}} &= \text{MSE}\left(\Delta T\right) \end{aligned} \tag{6.3}$$

The temperature guess is calculated once, prior to the start of physics-informed training. At that stage the PINN's estimate of the mixture fraction is very close to the reference data. The temperature guess loss serves as a way to punish the network when it drives away from a sensible solution for the temperature field, including trivial solutions. Additionally, and most importantly, it gives a good initialization for the temperature field to kick-start the physics-informed training phase.

It should be noted that since the density guess relies on the estimated temperature field, it should only be computed once the network is capable of accurately estimating the temperature. Before introducing the physics losses the PINN is trained only on the data loss, which in this case solely relies on pressure and mixture fraction data. The temperature guess, which relies on the mixture fraction field only, is introduced afterwards. Finally, only after the PINN can accurately estimate the temperature field, the density guess loss is introduced. Alternatively, all guess losses could be introduced at once, but this sequential introduction is a simple way to remove possible complications during training and ensure that all guess losses are minimized correctly before introducing the physics-informed terms.

As done in Section 6.5.1 the network is put through multiple stages of physics-informed training to investigate if it is possible to obtain adequate velocity reconstructions in the absence of both density and temperature data. The first stage is a standard momentum training phase. The performance of the network during this phase is shown in Figure 6.35.



(a) Evolution of losses

(b) Evolution of reconstruction errors

Figure 6.35: Training performance of the PINN during momentum training in the absence of density and temperature data

It can be seen that the momentum loss decreases by two orders of magnitude, despite experiencing some oscillations in the process. The density guess loss evolves in a similar way as previously shown in Section 6.5.1, increasing at the beginning of training alongside the data loss before settling at approximately 10^{-1} and diverging from the data loss. The temperature guess, on the other hand, follows the evolution of the data loss very closely. The same behavior can be observed in the evolution of the reconstruction errors, Figure 6.35b. While the density error increases during training, that of temperature remains low throughout.

The reason for this discrepancy is likely related to the role of density in the momentum loss — the density field appears explicitly in the formulation of the momentum and continuity equations, and therefore the field's evolution is directly related to the evolution of the momentum term. On the other hand, the feedback between the temperature field and the momentum loss is much weaker — temperature only appears in the momentum equations indirectly through the computation of the viscosity. The temperature field is therefore not strongly affected by the momentum loss. The temperature guess loss measures the discrepancy between the estimated temperature field and the temperature guess, which is a function of the mixture fraction. Since the mixture fraction is part of the data loss, it is reasonable to expect the temperature guess loss to be directly related to the evolution of the data loss, as seen in Figure 6.35. In any case, both the temperature and density guess losses remain low enough to ensure accurate reconstructions of both fields by the end of training.

The reconstruction error of the axial velocity is reduced during training, while that of the radial velocity remains constant. The reconstructed axial velocity field after momentum-training is satisfactory. The radial velocity, shown in Figure 6.36, is suboptimal, particularly in the air entrainment region, which remains problematic. The PINN is put through an additional physics-informed training phase in an attempt to the improve the radial velocity reconstruction. As done previously, two different types of multi-stage reconstructions are considered, namely a double momentum reconstruction and a momentum-to-mixture reconstruction.

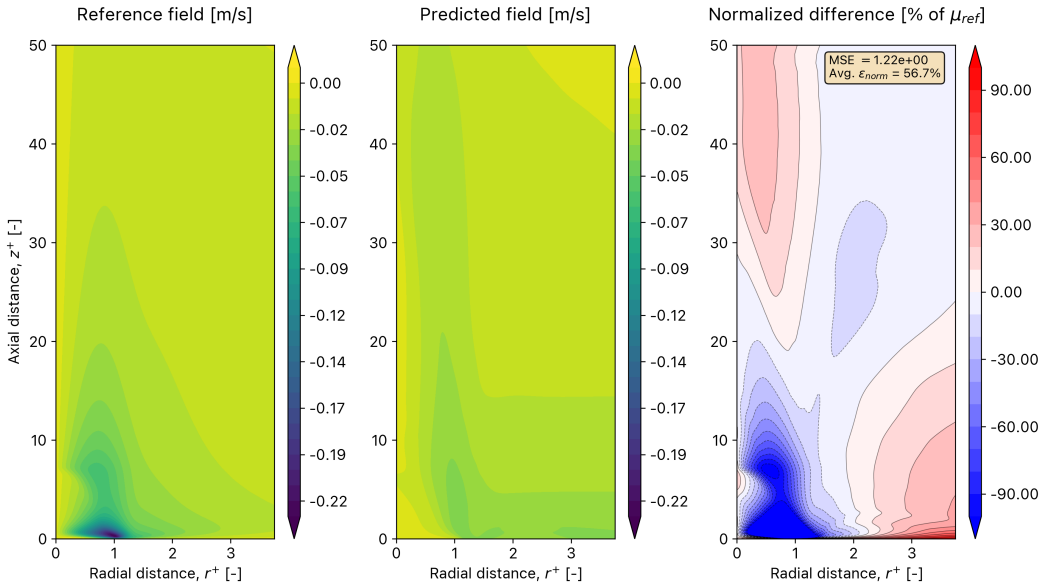


Figure 6.36: Reconstructed radial velocity field in the absence of density and temperature data using a single-stage reconstruction

A comparison of the final reconstruction accuracies for the two velocity components at the end of each of these runs is shown in Table 6.16. In both cases the results of the multi-stage reconstructions are compared against the baseline and against the results obtained from the first momentum-training phase, which is effectively a single-stage reconstruction. Several important observations can be made from this data.

For the axial velocity, it can be seen that the single-stage reconstruction yields the best performance, although it is only marginally better than the baseline. The data shows a small improvement across all reconstruction metrics with respect to the baseline. This improvement in axial reconstruction is attributed to the ability of the PINN to resolve the lower plume region, removing some of the oscillations that were present in the baseline case. Interestingly, the multi-stage runs worsen the quality of the axial reconstruction, resulting in underperformance with respect to the baseline. This behavior is attributed to the fact that the results after the single-stage reconstruction are already quite close to the optimal solution that can be reconstructed with this setup. As a consequence, by adding additional physics-informed phases and forcing the network to continue training and adapting its parameters, the estimated axial velocity fields are modified in a way that reduces their reconstruction accuracy. This indicates that the PINN can be overtrained, resulting in a decrease in the reconstruction accuracy. This is a consequence of the way in which the training is performed — the termination condition is fixed to kick in only when the network reaches a maximum number of iterations, or if the loss diverges. An additional termination constraint could be implemented to ensure that, if the total loss does not reach a new minimum with a fixed number of iterations or epochs, the termination is concluded. This would prevent the PINN from moving away from an optimal solution by terminating simulations that are not improving its loss function.

The radial velocity reconstruction does see an improvement as a result of multi-stage training. The double momentum run yields the best performance, improving the reconstruction quality over the baseline and over the single-stage reconstruction, with the momentum-to-mixture reconstruction in close second. Both multi-stage reconstructions offer superior performance than the baseline reconstruction across all four performance metrics. In this case, due to the fact that the radial velocity was not fully reconstructed by the single-stage reconstructions, improvements in the reconstruction accuracy can be obtained through additional physics-informed training.

Table 6.16: Summary of reconstruction performance in the absence of density and temperature data. Comparison of single and multi-stage reconstructions against the baseline. Best performance highlighted in gray.

Error metric	**Unit**	**Axial velocity** (u)			
		Baseline	Single-stage	Double momentum (multi-stage)	Momentum-to-mixture (multi-stage)
Standardized global MSE	–	$7.44 \cdot 10^{-2}$	$4.73 \cdot 10^{-2}$	$2.88 \cdot 10^{-1}$	$1.67 \cdot 10^{-1}$
Standardized flame MSE	–	$5.27 \cdot 10^{-2}$	$5.07 \cdot 10^{-2}$	$1.78 \cdot 10^{-1}$	$1.92 \cdot 10^{-1}$
Average absolute global normalized error	% of μ_{ref}	20.9	15.4	36.5	29.8
Average absolute flame normalized error	% of μ_{ref}	18.0	16.5	30.4	32.4
Error metric	**Unit**	**Radial velocity** (v)			
		Baseline	Single-stage	Double momentum (multi-stage)	Momentum-to-mixture (multi-stage)
Standardized global MSE	–	$6.75 \cdot 10^{-1}$	$1.15 \cdot 10^{0}$	$5.52 \cdot 10^{-1}$	$6.34 \cdot 10^{-1}$
Standardized flame MSE	–	$1.24 \cdot 10^{0}$	$1.55 \cdot 10^{0}$	$7.23 \cdot 10^{-1}$	$8.08 \cdot 10^{-1}$
Average absolute global normalized error	% of μ_{ref}	42.7	55.2	33.3	36.9
Average absolute flame normalized error	% of μ_{ref}	69.7	69.3	37.5	40.5

To highlight the differential effect of the multi-stage reconstructions on the two velocity components, Figure 6.37 and Figure 6.38 show the estimated fields at the end of the double momentum reconstruction for the axial and radial velocity, respectively. It can be seen that the radial velocity reconstruction has significantly improved, particularly within the air entrainment region, while the axial velocity has significantly worsened.

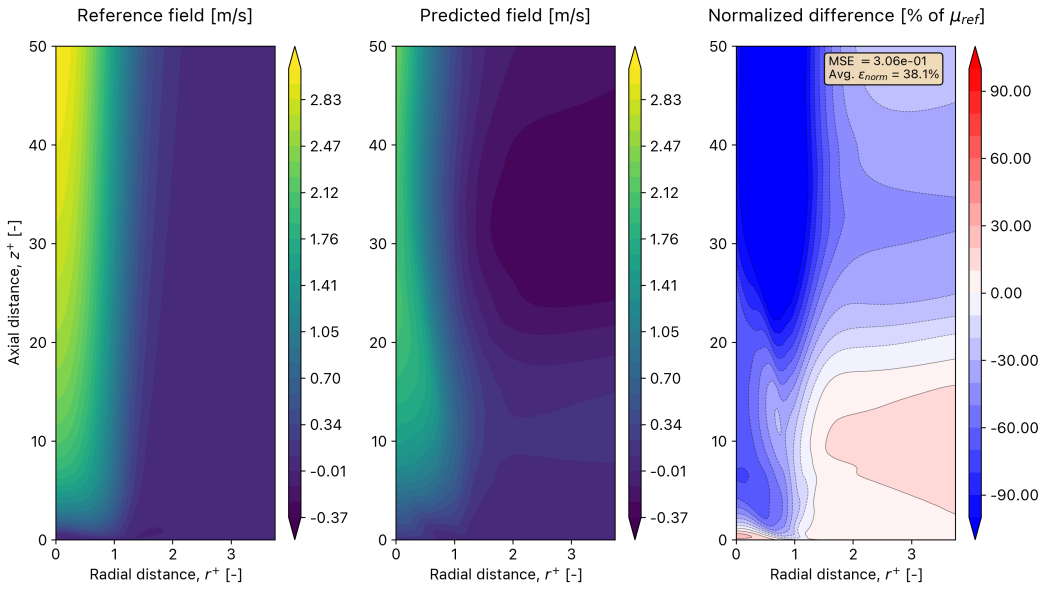


Figure 6.37: Reconstructed axial velocity field in the absence of density and temperature data using a multi-stage double momentum reconstruction

This differential treatment of the two velocity components is problematic. But there are some alternatives to address this issue. Firstly, one could opt to use single-stage reconstructions and accept that suboptimal reconstruction accuracies may be obtained for some flow quantities at the expense of improved stability. Alternatively, future work could explore avenues to improve or revisit the formulation of the momentum loss to improve the stability of the reconstruction, preventing the PINN from moving away from (locally) optimal solutions. This however may be difficult, for at least two reasons. Firstly, the instabilities that are observed are likely related to the oscillations in the computation of gradients via automatic differentiation that were observed and discussed in Chapter 5. It remains to be seen whether the occurrence of these spurious oscillations can be mitigated by reformulating the momentum loss. And secondly, a second factor that is believed to contribute to the instability of the PINN reconstruction is the complexity of the loss landscape itself, specially in cases where there are many loss terms in the

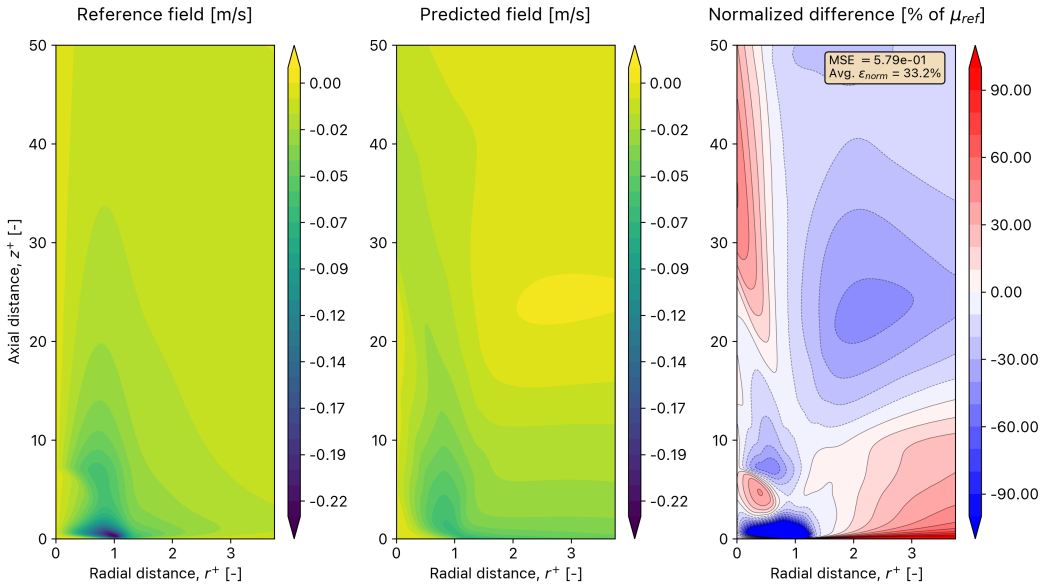


Figure 6.38: Reconstructed radial velocity field in the absence of density data using a single-stage double momentum reconstruction

composition of the network's total loss function.

In any case, the main conclusion that is drawn from the results presented in this section is that the mixture enhanced temperature guess yields a good initialization for the training process and results in adequate velocity reconstructions in the absence of both temperature and density data. Even disregarding the use of multi-stage reconstructions, the results from the single-stage reconstruction provide a good general characterization of the velocity fields. While it is true that the air entrainment region at the base of the flame is not well resolved by the single-stage reconstruction, the results are deemed to be satisfactory considering the amount of data provided to the PINN during this run. In the absence of density and temperature, and using only pressure and mixture fraction data, the PINN is capable of correctly reconstructing the characteristic flow features in both the axial and radial velocity fields using the mixture fraction enhanced guesses. Still, the reliance of these reconstructions on pressure data remains a point of concern from a practical standpoint. The focus of the next sections is an attempt to perform successful velocity reconstructions in the absence of pressure data.

6.5.3. Removing pressure and density

We now take on the challenge of removing the pressure data from the reconstructions. The ultimate goal is to remove all three data variables (pressure, density, and temperature) and rely solely on the mixture fraction. That possibility is explored in Section 6.5.5. Here, the temperature data is maintained alongside the mixture fraction. The goal is to see if adequate reconstructions can be obtained in the absence of pressure and density, and to test the response of the algorithm to the removal of the former.

As done previously for the density and for the temperature, a guess for the pressure field is used to initialize the training process. In this case the guess does not rely on the mixture fraction data, since an appropriate guess for the pressure field can be constructed without it. The pressure field is assumed to be a linear function of the streamwise coordinate z. Noting that the pressure at the top boundary of the domain is the ambient pressure (p_0), the pressure within the domain is assumed to increase linearly with decreasing z according to the hydrostatic equation. This results in the following initial guess for the pressure field:

$$p_{\text{guess}} = p_0 - \rho_{air} \cdot g_z \cdot (z_{max} - z) \tag{6.4}$$

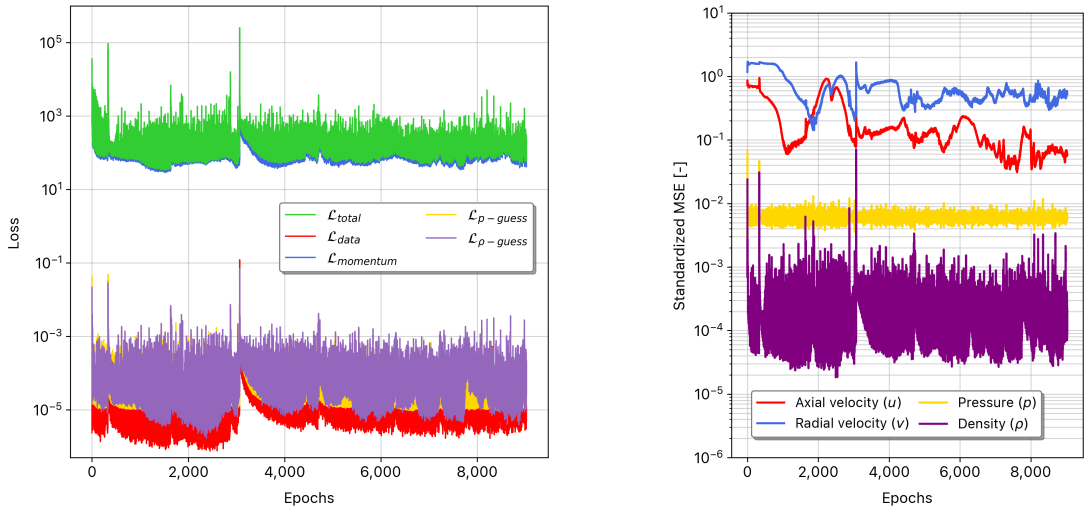
where ρ_{air} is assumed constant and equal to the density of air at standard conditions, $g_z = -9.81$

m/s^2 is the gravitational acceleration in the axial direction, and z_{max} is the axial coordinate of the top boundary. As usual, a guess-based loss is defined by calculating the difference with respect to this guess and monitoring the resulting mean squared error:

$$\begin{aligned} \Delta p_{\text{guess}} &= p_{\text{est}} - p_{\text{guess}} \\ \mathcal{L}_{p\text{-guess}} &= \text{MSE}\left(\Delta p\right) \end{aligned} \tag{6.5}$$

As done in Section 6.5.2, the pressure and density losses are introduced sequentially. Since the density guess depends on the pressure estimation of the network, the pressure loss is introduced first. Only after the pressure is correctly estimated by the PINN the density guess is evaluated, using the formulation of Equation 6.2, and the corresponding density loss is added to the network.

Following the findings of Section 6.3 a series of multi-stage reconstructions are performed. The performance of the network during this first stage of training is shown in Figure 6.39.



(a) Evolution of losses

(b) Evolution of reconstruction errors

Figure 6.39: Training performance of the PINN during momentum training in the absence of pressure and density data

Several observations can be drawn from the results shown in Figure 6.39. The momentum loss decreases significantly at the beginning of training, reducing by two orders of magnitude within the first 500 epochs. It then remains relatively stable at a value of approximately 10^2. This behavior is different from the one shown in Section 6.5.1 and Section 6.5.2, where the momentum loss showed a much more gradual decrease during physics-informed training. It appears that the removal of pressure from the data loss and its substitution by the corresponding guess loss reduces the time of convergence of the momentum loss. The data losses also show differences with respect to previous guess-based reconstructions. In this case, the density loss follows the evolution of the data loss very close, as does the pressure loss. All three of these losses remain sufficiently low during training to indicate that correct estimations of the pressure, density, and mixture fraction fields have been maintained throughout.

The difference in the behavior of the momentum loss with respect to previous runs may be indicative of a significant difference in the training of the network, and therefore on the accuracy of the reconstructions. In order to investigate this the estimated velocity fields after the single-stage momentum phase are shown in Figure 6.40 and Figure 6.41.

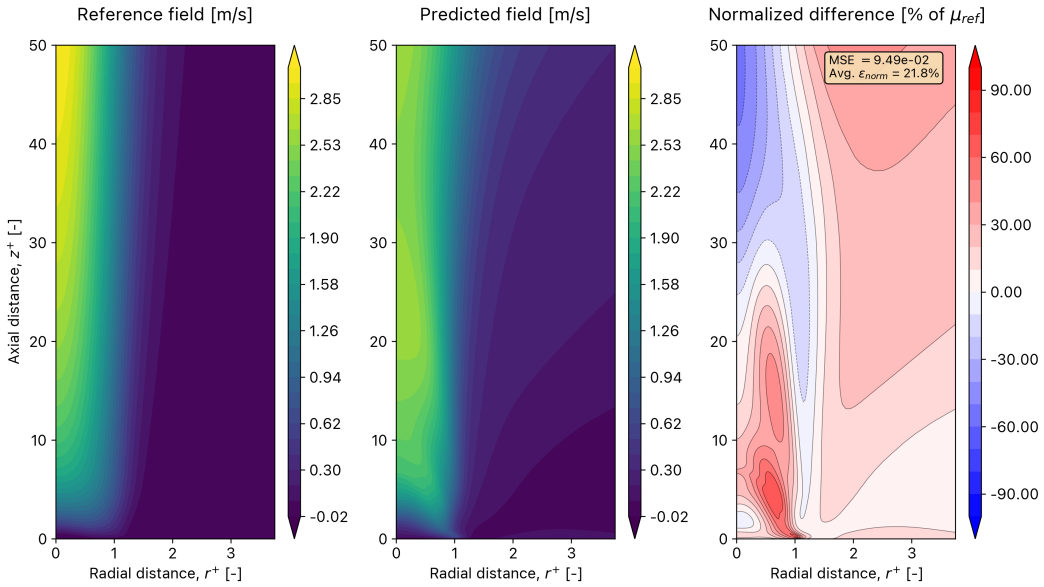


Figure 6.40: Reconstructed axial velocity field in the absence of pressure and density data using a single-stage momentum reconstruction

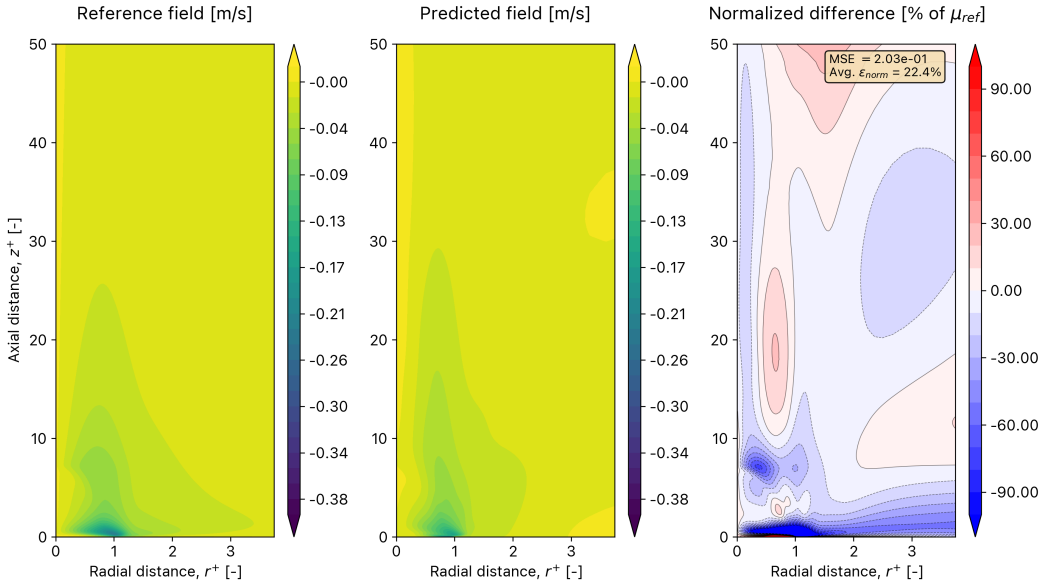


Figure 6.41: Reconstructed radial velocity field in the absence of pressure and density data using a single-stage momentum reconstruction

It can be seen that the accuracy of the radial velocity reconstruction is superior to that of previous runs, specially in the air entrainment region that, until now, had proven to be problematic. At the same time, the axial velocity reconstruction is qualitatively worse than those of previous single-stage momentum reconstruction attempts. The reconstruction of the plume has worsened — the network underestimates the magnitude of the axial velocity in the upper plume region and overestimates it in the lower region, stretching the shape of the plume near the base. Nevertheless, the axial reconstruction is generally adequate and still provides a good characterization of the axial velocity flow field.

It appears that the removal of pressure from the data loss and its substitution by a guess-based loss term has decreased the difference between the reconstruction accuracy of the two velocity components. Whereas previously the axial velocity was generally reconstructed significantly better than the radial

velocity, this gap has now reduced. Part of the reason for this decreased difference is the deterioration of the axial velocity reconstruction. But most importantly, the quality of the radial velocity reconstruction in the air entrainment region has significantly improved. The reconstruction accuracies of this run are shown in Table 6.17 — they are compared against the baseline reconstruction and against a single-stage momentum reconstruction where only the density is removed.

Table 6.17: Reconstruction performance of single-stage momentum reconstructions in the absence of data. Comparison of baseline against removal of density and removal of density and pressure. Best performance highlighted in gray.

Error metric	**Unit**	**Axial velocity** (u)			**Radial velocity** (v)		
		Baseline	Removing ρ	Removing ρ and p	Baseline	Removing ρ	Removing ρ and p
Standardized global MSE	–	$7.44 \cdot 10^{-2}$	$4.98 \cdot 10^{-2}$	$8.90 \cdot 10^{-2}$	$6.75 \cdot 10^{-1}$	$7.19 \cdot 10^{-1}$	$2.63 \cdot 10^{-1}$
Standardized flame MSE	–	$5.27 \cdot 10^{-2}$	$6.11 \cdot 10^{-2}$	$1.06 \cdot 10^{-1}$	$1.24 \cdot 10^{0}$	$9.59 \cdot 10^{-1}$	$2.55 \cdot 10^{-1}$
Average absolute global normalized error	% of μ_{ref}	20.9	17.2	20.7	42.7	45.1	24.3
Average absolute flame normalized error	% of μ_{ref}	18.0	19.8	23.1	69.7	55.2	26.2

The performance data of Table 6.17 agrees with the qualitative assessment of the reconstructed flow fields. Removing the pressure data worsens the reconstruction accuracy of the axial velocity compared to the baseline and the removal of density alone. The baseline reconstruction yields the best performance within the flame region, while removing only the density yields slight improvements over the baseline in the global performance metrics. Nevertheless, the differences between these two runs are not very significant. It is clear, however, that removing the pressure data is detrimental to the axial velocity reconstruction. The radial velocity reconstruction, on the other hand, clearly benefits from substituting the pressure data by a guess-based loss. It yields a clear improvement both in terms of the MSE and the average absolute normalized error, both globally and in the flame region. Additionally, it can be seen that the reconstruction errors of the radial velocity are now much closer to those of the axial component.

To complete the discussion on the effect of the removal of the pressure data, a series of multi-stage reconstructions are conducted. The goal is to investigate if the improved radial velocity reconstruction can be maintained while improving the axial reconstruction. As done in Section 6.5.1 and Section 6.5.2, two different multi-stage reconstructions are conducted. One is a double momentum reconstruction, and the other a momentum-to-mixture reconstruction. The performance of both of these runs is compared to the single-stage reconstruction inTable 6.18.

Table 6.18: Reconstruction performance of multi-stage reconstructions in the absence of pressure and density data. Comparison against the single-stage reconstruction under the same conditions. Best performance highlighted in gray.

Error metric	**Unit**	**Axial velocity** (u)			**Radial velocity** (v)		
		Single-stage	Double momentum	Momentum to mixture	Single-stage	Double momentum	Momentum to mixture
Standardized global MSE	–	$8.90 \cdot 10^{-2}$	$5.72 \cdot 10^{-1}$	$1.24 \cdot 10^{1}$	$2.63 \cdot 10^{-1}$	$2.18 \cdot 10^{0}$	$3.56 \cdot 10^{0}$
Standardized flame MSE	–	$1.06 \cdot 10^{-1}$	$4.44 \cdot 10^{-1}$	$1.18 \cdot 10^{1}$	$2.55 \cdot 10^{-1}$	$2.80 \cdot 10^{0}$	$4.36 \cdot 10^{0}$
Average absolute global normalized error	% of μ_{ref}	20.7	53.9	300.4	24.3	107.5	138.5
Average absolute flame normalized error	% of μ_{ref}	23.1	47.5	298.9	26.2	125.0	155.7

The multi-stage reconstructions not only fail to provide further improvements on the reconstruction accuracies, they simply fail to adequately reconstruct the flow fields altogether. A reduction in the reconstruction accuracy due to overtraining has already been observed in the reconstructions of Section 6.5.2. This shows that the reconstruction model suffers from instability – if a PINN model that is capable of accurately reconstructing the flow fields is forced to continue training without a termination criterion that allows it to stop the reconstructions if they continue to worsen, it tends to completely ruin the reconstructed fields. As the network is forced to continue adjusting its weights during training, errors creep into the reconstructed fields. These generally manifest themselves as erroneous flow features that pop in and out of the domain, as discussed in Section 6.1. The PINN is sometimes capable of suppressing the oscillations and returning back to its original state. However, in some instances the oscillations spiral out of control — the PINN cannot suppress them and ends up moving away from its initially adequate reconstruction, completely ruining the reconstructed fields in the process. An example of what these erroneous fields look like is shown in Figure 6.42 and Figure 6.43. These results highlight the complexity of the loss landscape that the network is trying to navigate, and show the

importance of defining appropriate termination conditions that allow the network to terminate diverging reconstructions before they fail catastrophically.

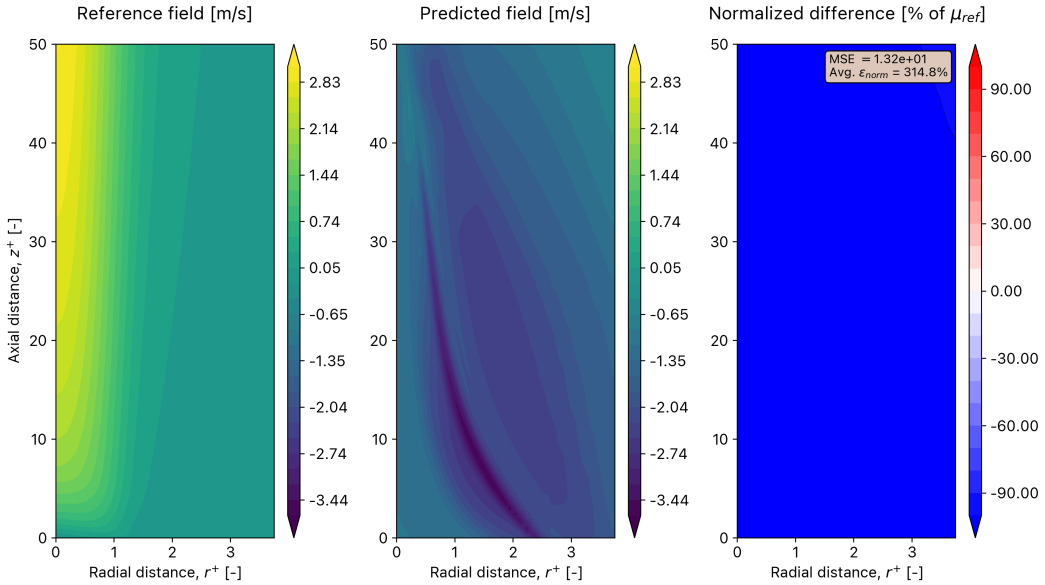


Figure 6.42: Failed axial velocity reconstruction. momentum-to-mixture multi-stage reconstruction in the absence of pressure and density data.

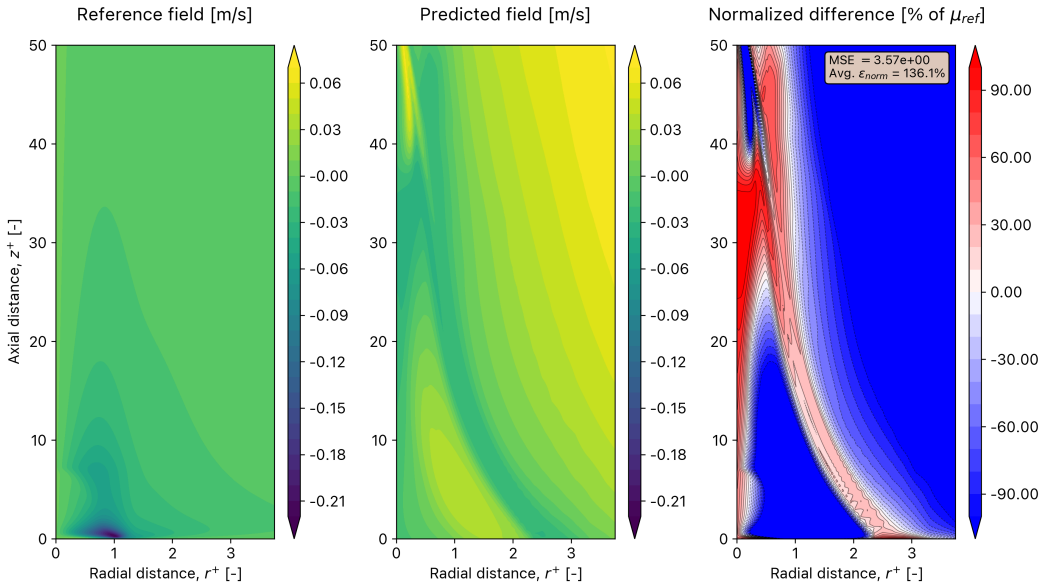


Figure 6.43: Failed radial velocity reconstruction. momentum-to-mixture multi-stage reconstruction in the absence of pressure and density data.

Overall, several conclusions can be drawn from the results presented in this section. Firstly, the extended PINN model is capable of successfully reconstructing the two velocity components in the absence of density and pressure data. The single-stage momentum reconstruction offers superior performance to both types of multi-stage training, which yield completely erroneous velocity fields due to overtraining. The single-stage reconstruction yields a significant improvement in the radial velocity reconstruction compared to the baseline, managing to resolve the air entrainment region much better than any previous reconstruction attempt. Nevertheless, this comes at the cost of a slight decrease in the axial velocity reconstruction. This trade-off is deemed to be favourable, since the axial velocity reconstruction remains

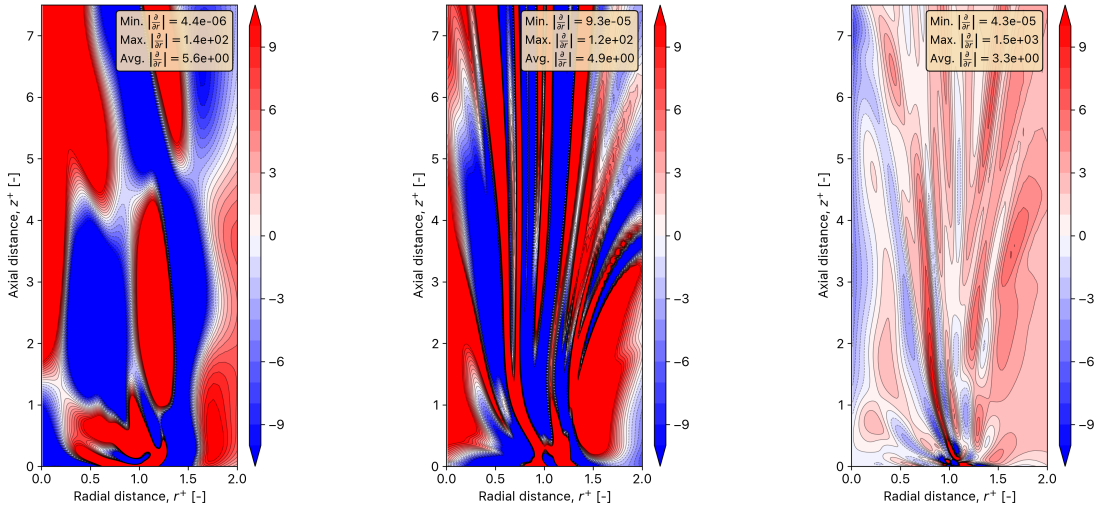
adequate while the radial velocity reconstruction improves significantly.

6.5.4. Suppressing spurious oscillations

In the previous section, we have seen how the reconstruction of the air entrainment region that characterizes the radial velocity field is significantly improved in the absence of pressure data. The radial velocity reconstruction clearly benefits from substituting the pressure data by a guess-based loss. It leads to a clear improvement both in terms of the MSE and the average absolute normalized error, both globally and in the flame region. This begs the question — why does removing the pressure data from the data loss, and substituting it by a guess-based loss, increase the radial reconstruction accuracy and yield such a significant improvement on the reconstruction of the air entrainment region?

To answer this question, consider again the physical mechanism that is at the root of the air entrainment process. As discussed in Chapter 2 air is entrained through the base of the flame as a consequence of baroclinicity. The relative misalignment of the density gradient (which acts primarily in the radial direction) and the pressure gradient (which acts along the axial direction) leads to the generation of baroclinic vorticity. The baroclinic contribution to the vorticity is known to be the dominant vorticity generation mechanism in laminar diffusion flames [117]. It is therefore sensible to assume that the ability of the network to resolve the air entrainment region may be related to the computation of the pressure gradients, which are at the root of the generation of baroclinic vorticity and directly affect the momentum loss as they appear explicitly in the Navier-Stokes equations.

Additionally, recall that the use of automatic differentiation in the computation of gradients has been shown to cause spurious oscillations that yield high residuals and create a bottleneck in the minimization of the momentum loss (see Section 5.4). Is it possible that the use of a pressure guess loss is partially alleviating this phenomenon from affecting the velocity reconstructions? To answer this question, Figure 6.44 shows a comparison of the radial pressure gradient evaluated at the end of training for the baseline case (Section 6.2), the case where only density is removed (Section 6.5.1), and the case discussed in this section, where both density and pressure are removed from the data loss.



(a) Baseline **(b)** Only density removed **(c)** Pressure and density removed

Figure 6.44: Radial pressure gradients after several single-stage momentum reconstructions. Comparison between baseline, removal of density, and removal of density and pressure. The removal of pressure reduces spurious oscillations. The text boxes on the top right show the maximum, minimum, and average of the absolute value of the fields. These statistics are approximately equal across all three reconstructions, indicating that the magnitude of the oscillation remains unaffected, and that it is the size of the affected regions that is reduced.

It appears that, indeed, removing the pressure from the data loss and substituting it with the pressure guess loss significantly reduces the spurious oscillations in the radial pressure gradient. Note that the

maximum, minimum, and average of the absolute value of the pressure gradients are very similar in all three cases (see the text boxes on the top right of the plots in Figure 6.44). This indicates that the magnitude of the oscillations remains constant, but the area that is affected by such oscillations is greatly reduced by the removal of the pressure data. This reduction in the spurious oscillations is believed to be the reason why removing the pressure yields a clear improvement in the radial reconstruction accuracy, specially within the air entrainment region.

Taking this analysis one step further, one may realize that the pressure guess loss is, in a way, another type of data loss. The only difference between the two is that the data loss measures the difference between the estimated pressure and the reference pressure data from the DNS simulations, whereas the guess loss measures the difference with respect to the guessed data field. In both cases the loss is measuring the difference between the estimated pressure and a target pressure field, which is constant — recall that the guessed pressure field is only a function of the axial coordinate z and is therefore independent of any estimations made by the PINN.

Considering that the DNS data satisfies the continuity and Navier-Stokes equations with a sufficient level of accuracy, one may wonder what exactly is the differentiating factor between the two loss terms that causes the reduction in spurious oscillations in the radial pressure gradient shown in Figure 6.44. It is believed that the reason for this reduction lies in the fact that the guess loss term *isolates* the reconstruction of the pressure field, whereas the data loss tries to minimize the combined error of multiple training variables at once. By isolating the pressure field into its own independent loss term, the network is given more flexibility to adapt its parameters in a way that reduces the spurious oscillations in the pressure gradients, which in turn allows the network to obtain improved radial velocity reconstructions, albeit at the cost of a small decrease in the axial reconstruction accuracy. This is also in agreement with the results of Section 6.5.1 and Section 6.5.2, where improvements in the reconstruction of the air entrainment region where also observed, albeit to a lesser extent than those presented here.

The results shown in this section indicate that the isolation of the pressure field into a guess-based loss can yield a significant improvement in the reconstruction of the air entrainment, which up until now had proven to be particularly problematic for the PINN. In the next section the focus is placed on performing the most ambitious reconstruction so far, where all three training variables from the baseline case (density, pressure, and temperature) are removed, leaving the mixture fraction as the only data source in the reconstruction.

6.5.5. Removing pressure, density, and temperature

The final step in the search for new minimal data solutions with the mixture fraction is discussed in this section. All three training variables from the baseline case are removed from the data loss in this reconstruction attempt, leaving the mixture fraction as the only data source to guide the reconstruction process. Such a reconstruction has never been attempted.

As done in previous sections, the three flow variables that are removed (density, pressure, and temperature) are substituted by guess based losses. The density guess uses the formulation introduced in Section 6.5.1 (Equation 6.2). The temperature guess uses the formulation introduced in Section 6.5.2 (Equation 6.3). And the pressure guess uses the formulation introduced in Section 6.5.3 (Equation 6.5). The guesses are introduced into the PINN sequentially, after performing a data training stage where the data loss relies solely on the mixture fraction. Recall that the density guessed depends on both pressure, temperature and mixture fraction, while the temperature guess depends only on the mixture fraction, and the pressure guess is independent of all other fields. Because of this, the pressure guess loss is introduced first. After minimizing this loss, the temperature guess loss is calculated and minimized. Finally, only after the network can accurately reproduce the temperature and pressure fields, the density guess is evaluated and minimized.

To ensure consistency with the results of Section 6.5.1, Section 6.5.2 and Section 6.5.3 the physics-informed training consists of a single-stage momentum reconstruction followed by two different multi-stage reconstructions (a double momentum reconstruction and a momentum-to-mixture reconstruction). The

axial velocity reconstruction after the initial momentum training phase is successful, in line with the baseline reconstruction, and it is therefore omitted. The radial velocity reconstruction is shown in Figure 6.45. The PINN can reconstruct the radial velocity field relatively well, correctly identifying the size and location of the air entrainment region near the base of the flame. However, the PINN incorrectly predicts the presence of a small pocket of positive radial velocity with the air entrainment region ($0 < r^+ < 1$, $z^+ \approx 5$). Because of the high value of radial velocity in this region, combined with an overestimation of the far field radial velocity, the reconstruction error remains relatively large. Still, it is clear that the PINN estimation is not completely inaccurate, and at the very least it can correctly identify the location of the air entrainment region and the general features of the radial velocity field.

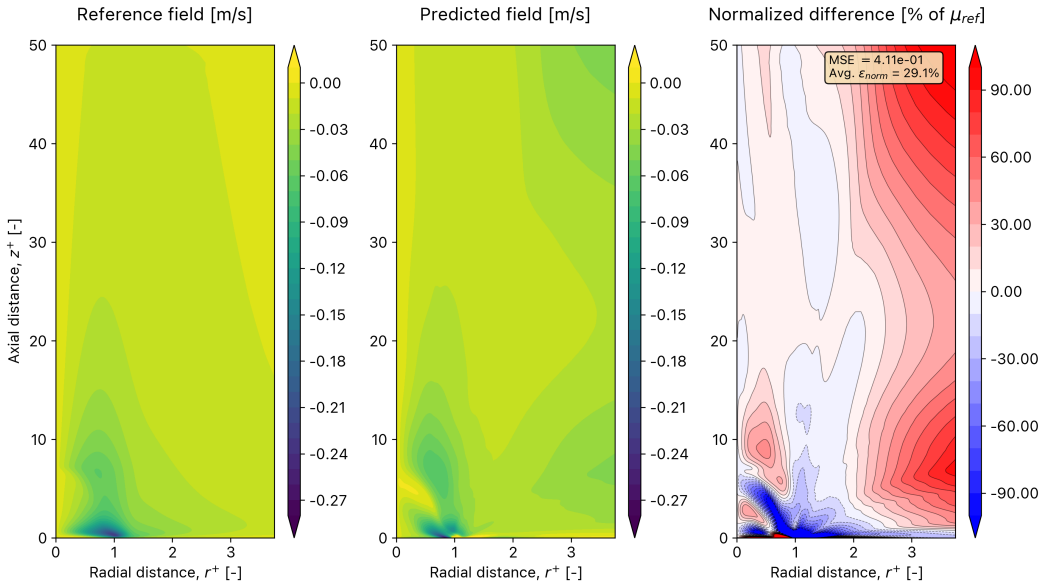


Figure 6.45: Reconstructed radial velocity field in the absence of pressure, density and temperature data using a single-stage momentum reconstruction

To get a better understanding of the performance of this reconstruction attempt, Table 6.19 compares the performance metrics against those of the baseline case and the single-stage reconstruction of Section 6.5.3. The data is in agreement with the visual inspection of the flow fields. The axial velocity metrics are quite similar between the three runs, indicating that the PINN is still capable of reconstructing the axial velocity field as well as in the baseline case even in the absence of pressure, density, and temperature data. The radial velocity metrics show that substituting the pressure, density, and temperature data by guess based losses yields an improvement over the baseline case across all metrics. Nevertheless, the case discussed in Section 6.5.3 still yields the best performance.

Table 6.19: Reconstruction performance of single-stage momentum reconstructions in the absence of pressure data. Comparison of baseline against removal of density and pressure and removal of density, pressure, and temperature. Best performance highlighted in gray.

Error metric	**Unit**	**Axial velocity** (u)			**Radial velocity** (v)		
		Baseline	Removing ρ and p	Removing ρ, p, and T	Baseline	Removing ρ and p	Removing ρ, p, and T
Standardized global MSE	–	$7.44 \cdot 10^{-2}$	$8.90 \cdot 10^{-2}$	$7.03 \cdot 10^{-2}$	$6.75 \cdot 10^{-1}$	$2.63 \cdot 10^{-1}$	$6.13 \cdot 10^{-1}$
Standardized flame MSE	–	$5.27 \cdot 10^{-2}$	$1.06 \cdot 10^{-1}$	$6.91 \cdot 10^{-2}$	$1.24 \cdot 10^{0}$	$2.55 \cdot 10^{-1}$	$5.13 \cdot 10^{-1}$
Average absolute global normalized error	% of μ_{ref}	20.9	20.7	18.9	42.7	24.3	36.2
Average absolute flame normalized error	% of μ_{ref}	18.0	23.1	18.7	69.7	26.2	34.7

The performance of the multi-stage reconstructions is summarized in Table 6.20, where it is compared against that of the single-stage reconstruction. The multi-stage reconstructions yield superior performance for the axial velocity. The momentum-to-mixture reconstruction has the best overall performance metrics, but the differences with respect to the double momentum reconstruction are minimal. Both multi-stage reconstructions yield an average absolute flame normalized error of around 10% — this is a significant

improvement with respect to the single-stage reconstruction and an even bigger improvement with respect to the baseline, indicative of very accurate reconstructions. For the radial velocity the multi-stage reconstructions also outperform the single-stage reconstruction, albeit by a smaller relative margin.

Table 6.20: Reconstruction performance of multi-stage reconstructions in the absence of pressure and density data. Comparison against the single-stage reconstruction under the same conditions. Best performance highlighted in gray.

Error metric	**Unit**	**Axial velocity** (u)			**Radial velocity** (v)		
		Single-stage	Double momentum	Momentum to mixture	Single-stage	Double momentum	Momentum to mixture
Standardized global MSE	–	$7.03 \cdot 10^{-2}$	$3.04 \cdot 10^{-2}$	$2.60 \cdot 10^{-2}$	$6.13 \cdot 10^{-1}$	$3.79 \cdot 10^{-1}$	$3.80 \cdot 10^{-1}$
Standardized flame MSE	–	$6.91 \cdot 10^{-2}$	$2.88 \cdot 10^{-2}$	$2.68 \cdot 10^{-2}$	$5.13 \cdot 10^{-1}$	$3.20 \cdot 10^{-1}$	$2.45 \cdot 10^{-1}$
Average absolute global normalized error	% of μ_{ref}	18.9	12.5	11.7	36.2	29.6	35.0
Average absolute flame normalized error	% of μ_{ref}	18.7	12.1	12.0	34.7	28.0	30.8

The best axial velocity reconstruction in the absence of pressure, density, and temperature data is shown in Figure 6.46. It can be seen that the reconstruction quality is very high. The overestimation of the plume region near the centerline of the flame has been suppressed, and the reconstruction of the lower plume region near the base has significantly improved compared to the baseline. Overall, considering the limited data given to the PINN, this reconstruction is deemed to be of very high quality. Although there is a small underestimation of the axial velocity within the upper part of the plume, it is clear that the PINN can provide a very accurate characterization of the flow field.

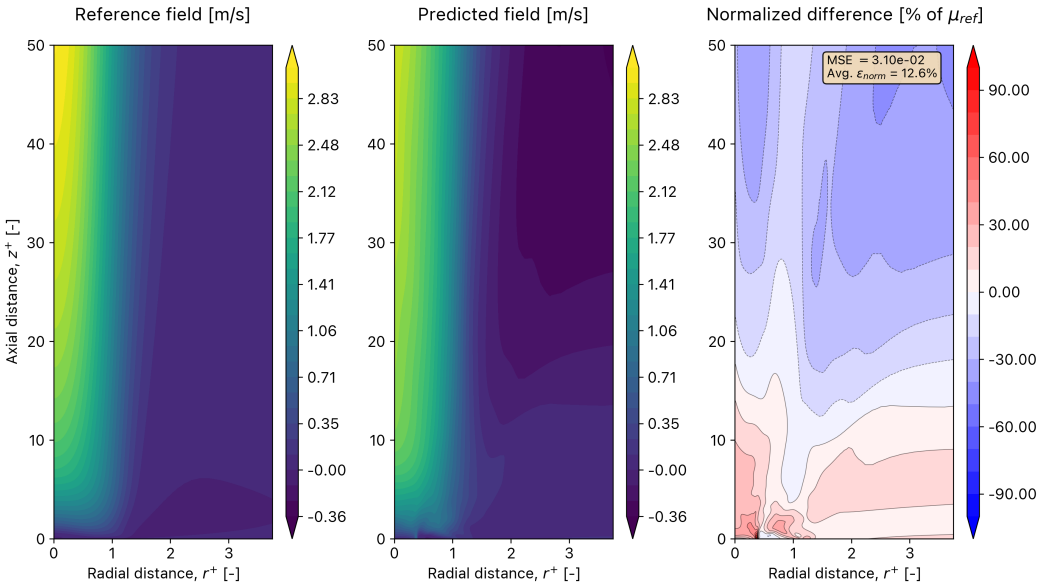


Figure 6.46: Best axial velocity reconstruction in the absence of pressure, density and temperature data

The best radial velocity reconstruction is shown in Figure 6.47. It can be seen that the multi-stage reconstruction is capable of improving the radial velocity reconstruction compared to the the single-step run. Although the network still incorrectly predicts the presence of a small region of positive axial velocity near the base of the flame, the size of this erroneous region has been greatly reduced. The air entrainment region is now resolved well, despite being stretched near the bottom boundary outside of the flame and despite an overestimation of the far field radial velocity.

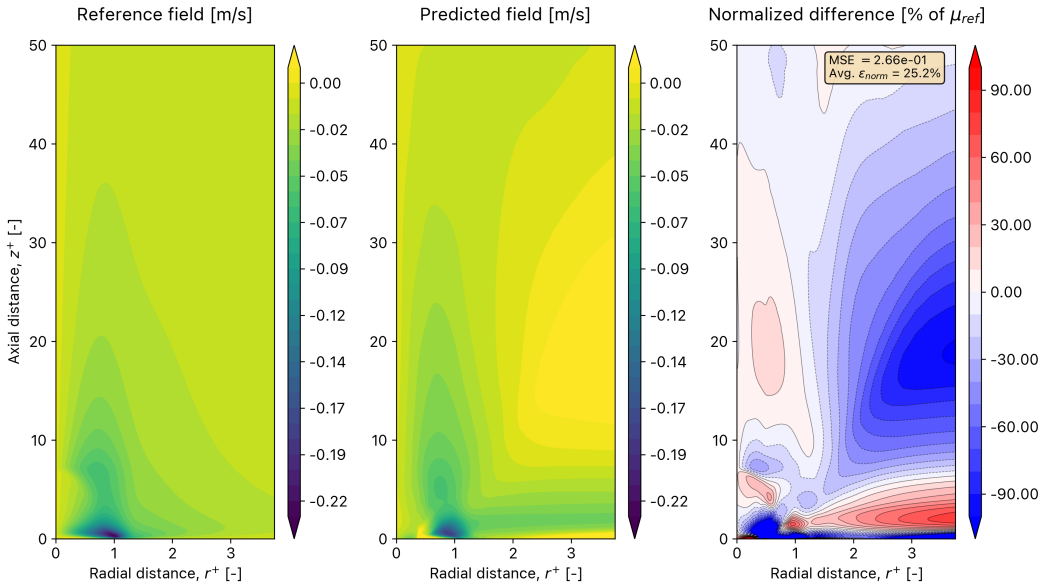


Figure 6.47: Best radial velocity reconstruction in the absence of pressure, density and temperature data

The results presented in this section are promising. Even in the absence of pressure, density, and temperature data, and relying only on the mixture fraction, the network can accurately characterize both velocity fields. Improvements with respect to the baseline are obtained for both the axial and radial velocity components. The improvements on the latter are particularly significant since the PINN had struggled to obtain accurate reconstructions of the air entrainment region in previous attempts. The findings presented here are in accordance with those of Section 6.5.3, demonstrating that a significant improvement in the reconstruction of the air entrainment region is obtained when the constraint on the pressure data is isolated by a guess based loss instead of being applied within a data loss that measures the combined error in multiple fields at once.

Overall, it can be concluded that the mixture fraction is an effective tool in the search for new minimal data solutions, going as far as reconstructing the velocity fields relying on the mixture fraction alone. However, it should also be noted that the quality of the obtained reconstructions is directly related to the fact that the mixture enhanced guesses are very good approximations of the reference data to begin with. In any case, the results provide good indication that, given a reasonable model to generate appropriate guesses for the flow fields, the PINN can robustly reconstruct the velocity fields in the absence of data. This procedure may be extended to other problems where different models can be used to define the guessed flow fields. Future work could also explore the possibility of applying these same guesses to more complex flames, but removing the guess based terms during physics informed training to give the network additional freedom to modify the fields. This may introduce a new set of problems (e.g. how does the removal of the guess terms affect the stability of the reconstruction process?), but it is an interesting avenue for future work to explore nonetheless.

Beyond demonstrating the potential of the mixture fraction as a way to perform minimal data velocity reconstructions in pool fire flames, the results also show that the isolation of the pressure field through a guess-based loss can significantly reduce the spurious oscillations that occur in the computation of its gradients using automatic differentiation. This result is a valuable contribution in and of itself — the reduction in spurious oscillations partially alleviates the instability of the reconstruction method, improving the velocity reconstructions globally and significantly improving the reconstruction accuracy in the air entrainment region, which had previously proven to be problematic for the PINN.

6.6. Reconstructing the HRR

As a final step in the evaluation of the reconstruction capabilities of the extended PINN algorithm, its ability to reconstruct the Heat Release Rate (HRR) of the flame is evaluated in this section. As previously discussed in Chapter 2 and Chapter 5, the HRR is a parameter of interest in the study of reacting flows due to its role as a proxy for the destructive power of a flame. From a practical standpoint it is desirable to obtain reasonable estimates for the HRR from the reconstruction algorithm.

The present approach for reconstructing the HRR relies on the prediction of the mixture fraction field and the diffusion flame structure to compute the enthalpy of the flame. For this reason, the HRR can only be estimated for reconstructions that include the mixture fraction, either as a data source or as a reconstruction target. Additionally, most of the attempts that fail to accurately reconstruct the principal features of the flow are omitted, since evidently they will not result in adequate HRR reconstructions. The reconstructed HRR values are presented in Table 6.21.

Table 6.21: Estimation of the HRR from reconstructed quantities.

Test family	**Section**	**Training variables**	**Training type***	**Reconstructed HRR** [W]	**Error**† [%]
Reference (DNS)	*Section 5.2*	–	–	102.7	–
Reconstructing ξ	Section 6.3	ρ, p, T	Mi-Mo	95.3	-7.2
Reconstructing ξ	Section 6.3	ρ, p, T	Mo-Mi	52.7	-48.7
Minimal solutions	Section 6.5.1	p, T, ξ	S	79.5	-22.6
Minimal solutions	Section 6.5.1	p, T, ξ	Mo-Mi	121.4	18.2
Minimal solutions	Section 6.5.1	p, T, ξ	DM	86.0	-16.2
Minimal solutions	Section 6.5.2	p, ξ	S	86.7	-15.6
Minimal solutions	Section 6.5.2	p, ξ	Mo-Mi	94.9	-7.6
Minimal solutions	Section 6.5.2	p, ξ	DM	112.4	9.4
Minimal solutions	Section 6.5.3	T, ξ	S	98.6	-4.0
Minimal solutions	Section 6.5.3	T, ξ	Mo-Mi	803.0	681.9
Minimal solutions	Section 6.5.3	T, ξ	DM	34.1	-66.8
Minimal solutions	Section 6.5.5	ξ	S	85.7	-16.6
Minimal solutions	Section 6.5.5	ξ	Mo-Mi	82.9	-19.3
Minimal solutions	Section 6.5.5	ξ	DM	66.3	-35.5

* Single-stage (S), momentum-to-mixture (Mo-Mi), mixture-to-momentum (Mi-Mo), double momentum (DM).
† Color coded as follows: error magnitude <10% in green, between 10% and 25% in yellow, and > 25% in red.

Several important findings can be derived from the results shown in Table 6.21. The HRR estimations show good agreement with the analysis of the reconstruction accuracy discussed in previous sections. Consider the tests of Section 6.3. There, we found that mixture-to-momentum training was beneficial to reconstruct the mixture fraction field, while the momentum-to-mixture reconstruction yielded better velocity reconstruction. This can also be observed in Table 6.21: due to the crucial role of the mixture fraction in the computation of the HRR, the mixture-to-momentum reconstruction yields the best HRR estimation from that family of tests, with an absolute error of $\sim 7\%$. Similarly, in Section 6.5.1 we found that the two multi-stage reconstructions resulted in superior reconstruction accuracy than the single-stage reconstruction, and that double momentum training in particular yielded the best results. The results of Section 6.6 are in agreement with this analysis, showing that the multi-stage reconstructions yield lower errors in the HRR estimation, with the double momentum run resulting in the lowest error magnitude. This same trend can also be observed in the results from Section 6.5.2, where once again the multi-stage reconstructions provided superior reconstruction accuracy and here they are found to yield the best

HRR estimations, with absolute errors below 10%. In Section 6.5.3 and Section 6.5.4 we discussed how substituting pressure and density data by mixture fraction enhanced guesses mitigated the spurious oscillations in the radial pressure gradient, leading to a significant improvement in the reconstruction of the air entrainment region. In this case the multi-stage reconstructions failed to reconstruct the velocity fields due to overtraining, and the single stage reconstruction was found to yield the best performance. The data of Table 6.21 supports this analysis — the single-stage reconstruction corresponding to Section 6.5.3 yields an excellent HRR estimation, while the two multi-stage reconstructions completely fail to reconstruct the HRR. Lastly, in Section 6.5.5 we discussed how the PINN is capable of obtaining adequate velocity reconstructions even in the absence of pressure, density, and temperature data thanks to the addition of the mixture fraction. Once again, the HRR estimations are consistent with this analysis. The removal of an additional variable shows a decrease in the HRR reconstruction accuracy with respect to the single-stage reconstruction of Section 6.5.3, but the estimations remain adequate. Overall, the results indicate a strong correlation between the quality of the velocity reconstructions and that of the HRR.

Moreover, the present estimation method can predict the order of magnitude of the HRR quite well. The estimated values are close to the that of the reference data, indicating that the approach is promising and may be used to get a first order estimate of the HRR of the flame. Particularly interesting is the ability of the PINN to obtain adequate HRR estimates in the absence of pressure and density data. The results corresponding to Section 6.5.3 show that the HRR of the flame can be predicted with an absolute error of 4% relying only on temperature and mixture fraction data. This is interesting from a practical standpoint: while pressure and density are hard to obtain experimentally, these results highlight that it may be possible to obtain reliable first order estimates of the HRR relying only on temperature and mixture fraction data. And even in the absence of temperature, relying only on mixture fraction data, the PINN algorithm can reconstruct the HRR with an absolute error of $\sim 15\%$.

Notwithstanding these findings, it is clear that the method is not capable of obtaining very accurate HRR estimations (e.g. error $<1\%$). Additionally, the results show a large degree of variance. This is believed to stem from the interplay between the method used to calculate the HRR, which involves assessing flow quantities at the boundaries of the domain, and the inherent failure modes of PINNs. These failure modes, identified in Section 6.1 and discussed throughout this chapter, lead to appearance of spurious, nonphysical oscillations throughout the domain, particularly near the boundaries. The oscillations introduce errors in the calculation of the enthalpy flux through the boundaries of the domain, leading to erroneous HRR estimates. The oscillations can appear even in successful reconstructions, where the characteristic features of the velocity fields are resolved correctly. This limits the ability of the algorithm to obtain precise HRR reconstructions and leads to a large degree of variance in the estimated HRR values shown in Table 6.21. Because of the presence of these oscillations, the placement and size of the control volume boundaries affects the HRR estimations. To ensure consistency, all HRR estimations shown in Table 6.21 have been performed using a control volume with boundaries set at $r^+ = 5a$ and $z^+ = 20a$. This placement is large enough to completely encompass the flame while staying far from the boundaries of the domain. Nevertheless, the distance to the domain boundaries is not sufficient to mitigate this issue completely, and the effect of spurious oscillations can be observed in the results.

Nevertheless, the PINN algorithm demonstrates potential to accurately estimate the order of magnitude of the HRR in reconstructions that show minimal spurious oscillations, obtaining absolute errors as low as 4%. In instances free from severe oscillations, the correlation between accurate velocity field reconstructions and HRR estimates is clear, suggesting that improvements in handling boundary conditions could enhance the robustness of HRR predictions. However, the oscillations that result from the failure modes of the PINN algorithm affect the quality of the HRR reconstructions. This underscores the need for further refinement of the PINN approach, particularly in managing the appearance of spurious oscillations, to develop a more robust method for predicting the HRR in the reconstruction of reacting flows.

Part III

Closure

7 Conclusions

Physics-informed neural networks (PINNs) have recently emerged as a promising tool to perform flow field reconstruction thanks to their ability to enforce compliance with governing physical laws while seamlessly incorporating multi-fidelity data. This thesis has explored the use of PINNs to reconstruct the flow fields in a pool fire flame, a canonical flow configuration in the study of non-premixed combustion. Reacting flows like pool fires stand in need of flow field reconstruction methods due to the inherent difficulties in obtaining adequate experimental characterizations of such flows as a consequence of the harsh environments that they create. Additionally, they pose an interesting challenge to test the applicability of PINNs to buoyancy-driven multi-physics problems that do not satisfy the Boussinesq approximation.

This thesis has attempted to extend the capabilities of PINNs as a flow field reconstruction method in pool fire flames through the addition of a new physical prior that embeds information about the flame structure. This has been done using the mixture fraction, a scalar quantity that plays a crucial role in the study of diffusion flames by quantifying the local state of mixing within them. In the limit of sufficiently fast chemistry, the introduction of the mixture fraction acts as a universal coordinate transformation that reduces the computation of thermochemical properties into a one-dimensional problem. The addition of the mixture fraction has been used to improve the accuracy of the velocity reconstructions, to find new solutions to the reconstruction problem that rely on minimal data, and to estimate the Heat Release Rate (HRR) of the pool fire, a parameter of interest that quantifies the destructive power of a flame. Several research questions have been formulated in Chapter 1 to tackle the research objective of this thesis. The answers to these questions are summarized in the following paragraphs.

Extending PINN-based reconstruction algorithms with the mixture fraction

A novel PINN-based reconstruction framework has been developed using the `Pytorch` library, which extends the capabilities of previous modeling work carried out with `Tensorflow` [26, 93]. The novel framework is flexible, modular, and robust, allowing users to easily create new PINN-based models, extending them with additional loss terms, and seamlessly applying them to new reconstruction problems. This new framework is a significant step with respect to previously existing reconstruction pipelines and may be used in future work to tackle more complex reconstructions in reacting flows.

The PINN-based framework has been applied to reconstruct the flow field in a two dimensional, steady, laminar, axisymmetric pool fire flame. The mixture fraction has been successfully incorporated into the reconstruction algorithm by implementing a new physics-informed loss term to enforce compliance with the mixture fraction transport equation. A suitable formulation for this governing equation has been derived. The derived formulation has been verified to be compatible with and complimentary to the standard momentum loss that enforces compliance with the continuity and Navier-Stokes equations.

The sensitivity and robustness of the extended PINN algorithm has been tested. The algorithm is found to be generally robust, although it exhibits some critical failure modes that expose its sensitivity. In particular, the PINN shows a tendency to converge towards trivial solutions that yield constant near-zero velocity fields in an attempt to quickly minimize its loss function. Due to the complexity of the loss landscape the network tends to get stuck in locally optimal solutions, complicating the training

process. The PINN also exhibits instability during training — perturbations to the reconstructed fields caused by strong oscillations in the network's loss function can propagate into the domain, generally through its boundaries. In some cases, the network is unable to suppress the perturbations, leading to erroneous reconstructions. These failure modes are believed to be related to or a consequence of the spurious oscillations that appear in the calculation of gradients using automatic differentiation.

Despite these failure modes, the new PINN algorithm has been shown to be robust and capable of obtaining remarkable velocity reconstructions considering the limited data that it is exposed to. The PINN provides adequate characterizations of the features in the velocity fields, specially for the axial velocity. The air entrainment region in the radial velocity profile, caused by baroclinic vorticity, has proven to be difficult for the PINN to handle — the network is often unable to resolve the high velocity gradients in that region and struggles to provide sufficiently accurate reconstructions.

On the effect of the mixture fraction in the reconstruction process

The effect of the mixture fraction on the reconstruction process has been evaluated. The mixture fraction has been used both as an additional reconstruction target (alongside the velocity components) and as a data source to guide the reconstruction process (alongside pressure, temperature, and density data). When using the mixture fraction as a data source, the extended PINN algorithm is capable of matching the baseline performance, but no meaningful improvements in the reconstruction of the velocity fields has been obtained.

Using the mixture fraction as an additional reconstruction target has led to some promising and interesting results. The addition of the mixture fraction-based loss term not only results in adequate reconstructions of the mixture fraction field, but it also serves as an additional constraint to improve the reconstruction of the velocity fields. Multi-stage reconstructions, where the momentum and mixture fraction loss terms are introduced sequentially, have been found to yield superior performance to single-stage reconstructions, where both terms are introduced simultaneously. The former lead to lower minimum and final values of the physics-informed losses, which in turn leads to significant improvements in the velocity reconstructions. This improvement is particularly significant in the air entrainment region of the radial velocity field, which had previously proven to be problematic for the PINN.

In addition, the reinitialization of the optimizer and the readjustment of the loss weights in between training phases has also proven to yield lower convergence times, albeit at the cost of increased complexity and additional user oversight during training. This is in agreement with the work of Loshchilov and Hutter [102], showing that the PINNs benefit from performing warm restarts during training. The order with which the physics-informed losses are introduced during multi-stage training has been found to play an important role in the reconstructions. The term that is introduced first fixes the local optimum that the network converges to. Starting with the momentum loss prioritizes the reconstruction of the velocity fields, while the reconstruction of the mixture fraction benefits from starting the training process with the mixture loss.

On the search for new minimal data solutions

From a practical standpoint, it is desirable to find solutions to the reconstruction problem that rely on a minimal amount of data sources. The addition of the mixture fraction has proven to be very useful in this regard. The extended PINN algorithm is capable of accurately reconstructing both the radial and axial velocity fields in the absence of pressure, density, and temperature data, relying on the mixture fraction as the only data source. Mixture fraction enhanced guesses are used to initialize the missing flow fields. These guesses make use of the Burke-Schumann approximation, based on the assumption of infinitely fast irreversible chemistry, to estimate thermochemical properties from a pre-computed flame structure in mixture fraction space.

Beyond demonstrating the potential of the mixture fraction as a way to perform minimal data velocity reconstructions, the results also show that the isolation of the pressure field through a guess based loss can significantly reduce the spurious oscillations that occur in the computation of its radial gradients using automatic differentiation. This result is a valuable contribution in and of itself — the reduction in spurious oscillations partially alleviates the instability of the reconstruction method, improving the

velocity reconstructions globally and significantly improving the reconstruction accuracy in the air entrainment region.

Reconstructing the HRR

The PINN algorithm has been successfully extended to estimate the flame's HRR from reconstructed flow fields by performing a macroscopic energy balance over a control volume enclosing the flame. This method has been verified by comparison with the DNS data. The method yields generally adequate estimations of the HRR rate. The estimated values are close to the that of the reference data, indicating that the approach is promising and may be used to get a first order estimate of the HRR of the flame. The method predicts the order of magnitude of the HRR quite well, with errors as low as 4% in the best reconstructions. Particularly interesting is the ability of the PINN to obtain adequate HRR estimates in the absence of pressure and density data. This is valuable from a practical standpoint: while pressure and density are hard to obtain experimentally, the results highlight that it is possible to obtain adequate first order estimates of the HRR relying only on temperature and mixture fraction data. In addition, the HRR estimations show a strong correlation with the accuracy of the velocity reconstructions, and they highlight that both physics-informed loss terms (mixture fraction and momentum) are needed to obtain an adequate estimation of the HRR.

Despite these promising results, the ability of the PINN to obtain low error rates in the HRR estimation (i.e., below 5%) remains limited. This is attributed to the presence of spurious oscillations near the far field boundaries as a result of the inherent failure modes of the PINN model. The calculation of HRR is inherently sensitive to these oscillations. As a result, erroneous HRR estimations can be obtained even in cases where the principal features of the velocity fields are accurately reconstructed by the PINN. This underscores the need for further refinement of the PINN model, particularly in managing the appearance of spurious oscillations and in mitigating its inherent failure modes, in order to develop a more robust method for accurately predicting the HRR of pool fire flames.

Interpretation

Overall, the results of this thesis demonstrate the potential of PINNs as a flow field reconstruction method in pool fire flames. Additionally, they highlight the value of the mixture fraction as an extension to the reconstruction algorithm. The addition of the mixture fraction either reduces the amount of flow data that is required to obtain successful velocity reconstruction, leads to higher reconstruction accuracies, alleviates some of the network's well-known failure modes, or a combination of the above. Furthermore, it allows for adequate first order estimations of the HRR of the flame even in the absence of measurement data.

New minimal data solutions that rely solely on the mixture fraction data have been found, using the Burke-Schumann approximation to construct appropriate initializations for the missing fields. The success of these minimal data reconstructions is, to a large extent, a direct consequence of the applicability of the Burke-Schumann approximation to the problem under consideration. It remains to be seen whether applying this same approach to a different problem would yield satisfactory results. In problems where the use of the Burke-Schumann approximation is not fully warranted, the mixture enhanced guesses can still be used as an initialization of the fields. Removing the guess-based losses from the training process would give the PINNs the freedom to adapt the initial guesses as needed, which could still lead to successful reconstructions. For radically different problems where the Burke-Schumann approximation is not applicable at all, the assumptions used to model the flame structure would have to be revisited altogether. In that case, a different model could be used to initialize the missing fields. The model choice would depend on the specific problem under consideration.

Much work remains to be done for PINNs to become a robust method that can be seamlessly applied to reconstruction problems with practical relevance for reaction flows (see Chapter 8). Nevertheless, the results of this thesis indicate that it is possible to merge data-driven approaches, machine learning techniques, and our knowledge of the physical laws that govern the dynamics of fluids in a unified reconstruction framework to improve our ability to model complex flow phenomena.

8

Recommendations

Despite the promising results presented in this thesis, many questions are yet to be answered to develop the PINN-based reconstruction algorithm into a fully functional flow field reconstruction method with practical applications in the study of reacting flows like pool fires. This chapter briefly discusses a set of research directions that could be explored in future work to continue extending the capabilities of PINNs in this context.

From a flow physics standpoint, several interesting research directions are possible. First and foremost, the extended PINN algorithm could be applied to other pool fire configurations. The first step would be to apply it to pool fires of different sizes, eventually exploring the effect of the mixture fraction in the modeling of the unsteady puffing behavior investigated by Sitte and Doan [26]. In the future, the method should also be extended to deal with a three dimensional domain or a turbulent flow configuration. Evidently, this will significantly increase the complexity of the problem, and will require some consideration to decide how to reconstruct turbulent features within the PINN.

Second, the use of this method on real experimental data is also an interesting direction for future research. It would be interesting to assess whether experimental data on a passive scalar can be used as a proxy for the mixture fraction. This information could be used to replicate the minimal data reconstructions discussed in Section 6.5. Moreover, it would also be interesting to test the current reconstruction framework in a problem where the use of the Burke-Schumann approximation is not so well justified, by for example considering a diffusion flame with a lower Damköhler number, or simply considering a totally different reacting flow configuration altogether.

Third, in light of the failure modes of the PINN algorithm, it may be wise to consider ways to improve the numerical stability of the method. Future research should prioritize strategies aimed at enhancing the stability of the reconstructions, mitigating the failure modes of PINNs, and potentially reducing the tendency of the networks to diverge towards trivial solutions. The results of the present work indicate that an interesting first step in this direction could be a more detailed investigation regarding the spurious oscillations that have been observed in the computation of gradients using automatic differentiation and are believed to play a pivotal role in the instability of the reconstructions. Gaining a better understanding of the conditions under which these oscillations occur would be a significant step to obtain a more robust algorithm that is more readily applicable to reconstruction problems of practical relevance. Additionally, this analysis may shed light into the interrelation between the different failure modes that have been characterized in the present work, and could identify a different failure mode as the root cause of the spurious oscillations.

Fourth, improving the ability of the algorithm to estimate the HRR from reconstructed quantities could also be an interesting avenue for future work. It may be beneficial to consider alternative methods to compute the HRR field throughout the entire domain, instead of relying solely on information at the boundaries of a control volume. This approach could offer a more comprehensive representation of the global performance of the algorithm and increase its robustness. However, focusing solely on adjusting the HRR calculation might be akin to treating the symptoms rather than addressing the underlying

issue: the aforementioned instability in the reconstructions and the prevalence of spurious oscillations. Removing or significantly mitigating these failure modes would inherently improve the robustness of the HRR estimations.

Fifth, future work should also explore improvements from the neural network side. The reconstruction problem could be tackled using multiple PINNs instead of one, using, for example, one network for each reconstructed variable. Dynamic loss balancing techniques could also be considered, by for example standardizing the loss components prior to the application of their weights. By doing so, the difference between the data and physics-informed terms could be fixed throughout the entire duration of training, as opposed to only at the start of it. This modification can be easily implemented thanks to the flexibility of the new reconstruction framework, and it may result in improved training behavior.

Lastly, it may be necessary to revisit the network type and architecture, or to use novel variants of PINNs (such as competitive PINNs, for example) that offer advantages over the standard PINN architecture used in this thesis. The multilayer perceptron configuration of the standard PINN may become limited as the complexity of the problem increases, especially when making the jump to a turbulent and/or three dimensional problem. A specific example could be the use of a long short-term memory network (LSTM), or another type of recurrent neural network (RNN), to capture the temporal dependency of the puffing behavior that is characteristic of unsteady pool fires. The field of machine learning is growing at an incredibly fast pace, and improving the design of the network could open the door to physics-informed flow field reconstructions in more complex diffusion flames.

Reference list

[1] G. E. Karniadakis et al. "Physics-informed machine learning". In: *Nature Reviews Physics* 3.6 (June 2021). Number: 6 Publisher: Nature Publishing Group, pp. 422–440. DOI: 10.1038/s42254-021-00314-5.

[2] M. Mendez et al. *Data-Driven Fluid Mechanics: Combining First Principles and Machine Learning*. ISBN: 9781108842143. Cambridge University Press, 2023.

[3] S. Cai et al. "Physics-informed neural networks (PINNs) for fluid mechanics: a review". In: *Acta Mechanica Sinica* 37.12 (Dec. 1, 2021), pp. 1727–1738. DOI: 10.1007/s10409-021-01148-1.

[4] A. Lavin et al. *Simulation Intelligence: Towards a New Generation of Scientific Methods*. Nov. 27, 2022. DOI: 10.48550/arXiv.2112.03235.

[5] S. Cuomo et al. "Scientific Machine Learning Through Physics–Informed Neural Networks: Where we are and What's Next". In: *Journal of Scientific Computing* 92.3 (July 26, 2022), p. 88. DOI: 10.1007/s10915-022-01939-z.

[6] J. L. Callaham, K. Maeda, and S. L. Brunton. "Robust flow reconstruction from limited measurements via sparse representation". In: *Physical Review Fluids* 4.10 (Oct. 30, 2019). Publisher: American Physical Society, p. 103907. DOI: 10.1103/PhysRevFluids.4.103907.

[7] P. Joulain. "The behavior of pool fires: State of the art and new insights". In: *Symposium (International) on Combustion* 27.2 (Jan. 1, 1998), pp. 2691–2706. DOI: 10.1016/S0082-0784(98)80125-2.

[8] T. Poinsot and D. Veynante. *Theoretical and Numerical Combustion*. Third Edition. ISBN: 978-2-7466-3990-4. Bordeaux, France: AFNIL, 2012.

[9] G. Cox. "Fire research in the 21st century". In: *Fire Safety Journal* 32.3 (Apr. 1, 1999), pp. 203–219. DOI: 10.1016/S0379-7112(98)00014-9.

[10] N. N. Brushlinskiy and S. V. Sokolov. "How much is the fire "cost" in the modern world?" In: *Pozharovzryvobezopasnost/Fire and Explosion Safety* 29.1 (Mar. 13, 2020), pp. 79–88. DOI: 10.18322/PVB.2020.29.01.79-88.

[11] Z. Zhuang et al. *Total Cost of Fire in the United States*. Buffalo, NY, USA: National Fire Protection Association (NFPA), Oct. 2017.

[12] M. McNamee et al. "IAFSS agenda 2030 for a fire safe world". In: *Fire Safety Journal* 110 (Dec. 1, 2019), p. 102889. DOI: 10.1016/j.firesaf.2019.102889.

[13] *A WHO plan for burn prevention and care*. 2008. URL: https://www.who.int/publications/i/item/9789241596299.

[14] *World Population Prospects, The 2017 Revision*. ESA/P/WP/248. New York: United Nations, Department of Economic and Social Affairs, 2017.

[15] M. W. Jones et al. *Climate change increases the risk of wildfires*. 2020. URL: https://sciencebrief.org/briefs/wildfires.

[16] H. G. Smith et al. "Wildfire effects on water quality in forest catchments: A review with implications for water supply". In: *Journal of Hydrology* 396.1 (Jan. 5, 2011), pp. 170–192. DOI: 10.1016/j.jhydrol.2010.10.043.

[17] *Climate change: Worries over CO2 emissions from intensifying wildfires*. Nov. 15, 2018. URL: https://www.bbc.com/news/science-environment-46212844 (visited on 07/04/2023).

[18] S. R. Tieszen. "On the Fluid Mechanics of Fires". In: *Annual Review of Fluid Mechanics* 33.1 (2001), pp. 67–92. DOI: 10.1146/annurev.fluid.33.1.67.

[19] M. A. Finney et al. "Role of buoyant flame dynamics in wildfire spread". In: *Proceedings of the National Academy of Sciences* 112.32 (2015). Publisher: National Acad Sciences, pp. 9833–9838.

[20] M. Sibulkin and A. G. Hansen. "Experimental Study of Flame Spreading over a Horizontal Fuel Surface". In: *Combustion Science and Technology* 10.1 (Jan. 1, 1975). Publisher: Taylor & Francis, pp. 85–92. DOI: 10.1080/00102207508946660.

[21] C. Shi et al. "Metro train carriage combustion behaviors – Full-scale experiment study". In: *Tunnelling and Underground Space Technology* 104 (Oct. 1, 2020), p. 103544. DOI: 10.1016/j.tust.2020.103544.

[22] A. Lönnermark et al. *Full-scale fire tests with a commuter train in a tunnel*. SP Report : 2012:05. SP Technical Research Institute of Sweden, Jan. 1, 2012. DOI: 10.13140/RG.2.2.14571.95527.

[23] R. N. Newman. "The ignition and burning behaviour of sodium metal in air". In: *Progress in Nuclear Energy* 12.2 (Jan. 1, 1983), pp. 119–147. DOI: 10.1016/0149-1970(83)90020-3.

[24] M. R. Manco et al. "Evaluation of localized pool fire models to predict the thermal field in offshore topside structures". In: *Journal of the Brazilian Society of Mechanical Sciences and Engineering* 42.12 (Nov. 4, 2020), p. 613. DOI: 10.1007/s40430-020-02694-8.

[25] A. Jenft et al. "Experimental and numerical study of pool fire suppression using water mist". In: *Fire Safety Journal* 67 (July 1, 2014), pp. 1–12. DOI: 10.1016/j.firesaf.2014.05.003.

[26] M. P. Sitte and N. A. K. Doan. "Velocity reconstruction in puffing pool fires with physics-informed neural networks". In: *Physics of Fluids* 34.8 (Aug. 22, 2022), p. 087124. DOI: 10.1063/5.0097496.

[27] Y. Guo et al. "Pool fire burning characteristics and risks under wind-free conditions: State-of-the-art". In: *Fire Safety Journal* 136 (Apr. 1, 2023), p. 103755. DOI: 10.1016/j.firesaf.2023.103755.

[28] D. Moreno-Boza et al. "On the critical conditions for pool-fire puffing". In: *Combustion and Flame* 192 (June 1, 2018), pp. 426–438. DOI: 10.1016/j.combustflame.2018.02.011.

[29] N. Peters. *Combustion Theory*. CEFRC Summer School. June 2010. URL: https://cefrc.princeton.edu/combustion-summer-school/archived-programs/2010-summer-school/lecture-notes.

[30] J. D. Anderson. *Fundamentals of aerodynamics*. Sixth edition. McGraw-Hill Series in Aeronautical and Aerospace Engineering. New York, NY: McGraw Hill Education, 2017. 1130 pp.

[31] X. Xia and P. Zhang. "A vortex-dynamical scaling theory for flickering buoyant diffusion flames". In: *Journal of Fluid Mechanics* 855 (Nov. 2018). Publisher: Cambridge University Press, pp. 1156–1169. DOI: 10.1017/jfm.2018.707.

[32] Y. Chen et al. "Pool fire dynamics: Principles, models and recent advances". In: *Progress in Energy and Combustion Science* 95 (Mar. 1, 2023), p. 101070. DOI: 10.1016/j.pecs.2022.101070.

[33] M. L. Janssens. "Fundamental measurement techniques (Chapter 2)". In: *Flammability Testing of Materials Used in Construction, Transport and Mining (Second Edition)*. Ed. by V. Apte. Woodhead Publishing Series in Civil and Structural Engineering. Woodhead Publishing, Jan. 1, 2022, pp. 23–61. DOI: 10.1016/B978-0-08-102801-8.00092-2.

[34] A. Schönbucher et al. "Static and dynamic radiance structures in pool fires". In: *Symposium (International) on Combustion*. Twenty-First Symposuim (International on Combustion) 21.1 (Jan. 1, 1988), pp. 93–100. DOI: 10.1016/S0082-0784(88)80235-2.

[35] H. Hayasaka. "Unsteady Burning Rates Of Small Pool Fires". In: *Fire Safety Science* 5 (1997), pp. 499–510.

[36] H. Hayasaka. "A Study On Pool Flame Structure Using Thermography". In: *Fire Safety Science* 4 (1994), pp. 125–136.

[37] L. Audouin et al. "Average centreline temperatures of a buoyant pool fire obtained by image processing of video recordings". In: *Fire Safety Journal* 24.2 (Jan. 1, 1995), pp. 167–187. DOI: 10.1016/0379-7112(95)00021-K.

[38] C. Q. G. Tashtoush et al. "Structure of large scale pool fires". In: International Conference on Fire Research and Engineering. 1995.

[39] F. Michaux, P. Mattern, and S. Kallweit. "RoboPIV: how robotics enable PIV on a large industrial scale". In: *Measurement Science and Technology* 29.7 (May 2018). Publisher: IOP Publishing, p. 074009. DOI: 10.1088/1361-6501/aab5c1.

[40] J. F. G. Schneiders et al. "Large-scale volumetric pressure from tomographic PTV with HFSB tracers". In: *Experiments in Fluids* 57.11 (Oct. 13, 2016), p. 164. DOI: 10.1007/s00348-016-2258-x.

[41] D. Chamberlin and A. Rose. "The flicker of luminous flames". In: *Industrial & Engineering Chemistry* 20.10 (1928). Publisher: ACS Publications, pp. 1013–1016.

[42] H. Pitsch. *Laminar Diffusion Flames*. CEFRC Summer School. 2014. URL: https://cefrc.princeton.edu/combustion-summer-school/archived-programs/2014-session/lecture-notes.

[43] Y. G. Guezennec. "Stochastic estimation of coherent structures in turbulent boundary layers". In: *Physics of Fluids A: Fluid Dynamics* 1.6 (June 1, 1989), pp. 1054–1060. DOI: 10.1063/1.857396.

[44] A. M. Naguib, C. E. Wark, and O. Juckenhöfel. "Stochastic estimation and flow sources associated with surface pressure events in a turbulent boundary layer". In: *Physics of Fluids* 13.9 (Sept. 1, 2001), pp. 2611–2626. DOI: 10.1063/1.1389284.

[45] N. E. Murray and L. S. Ukeiley. "Modified quadratic stochastic estimation of resonating subsonic cavity flow". In: *Journal of Turbulence* 8 (Jan. 1, 2007). Publisher: Taylor & Francis, N53. DOI: 10.1080/14685240701656121.

[46] N. E. Murray and L. S. Ukeiley. "Estimation of the Flowfield from Surface Pressure Measurements in an Open Cavity". In: *AIAA Journal* 41.5 (May 2003). Publisher: American Institute of Aeronautics and Astronautics, pp. 969–972. DOI: 10.2514/2.2035.

[47] J. P. Bonnet et al. "Stochastic estimation and proper orthogonal decomposition: Complementary techniques for identifying structure". In: *Experiments in Fluids* 17.5 (Sept. 1, 1994), pp. 307–314. DOI: 10.1007/BF01874409.

[48] C. E. Tinney et al. "On spectral linear stochastic estimation". In: *Experiments in Fluids* 41.5 (Nov. 1, 2006), pp. 763–775. DOI: 10.1007/s00348-006-0199-5.

[49] R. J. Adrian. "On the role of conditional averages in turbulence theory." In: 1975.

[50] T. C. Tung and R. J. Adrian. "Higher-order estimates of conditional eddies in isotropic turbulence". In: *The Physics of Fluids* 23.7 (July 1, 1980), pp. 1469–1470. DOI: 10.1063/1.863130.

[51] R. J. Adrian. "Conditional eddies in isotropic turbulence". In: *The Physics of Fluids* 22.11 (Nov. 1, 1979), pp. 2065–2070. DOI: 10.1063/1.862515.

[52] L. M. Hudy, A. Naguib, and W. M. Humphreys. "Stochastic estimation of a separated-flow field using wall-pressure-array measurements". In: *Physics of Fluids* 19.2 (Feb. 27, 2007), p. 024103. DOI: 10.1063/1.2472507.

[53] C. W. Rowley and S. T. Dawson. "Model Reduction for Flow Analysis and Control". In: *Annual Review of Fluid Mechanics* 49.1 (Jan. 3, 2017). Publisher: Annual Reviews, pp. 387–417. DOI: 10.1146/annurev-fluid-010816-060042.

[54] J.-C. Loiseau and S. L. Brunton. "Constrained sparse Galerkin regression". In: *Journal of Fluid Mechanics* 838 (Mar. 2018). Publisher: Cambridge University Press, pp. 42–67. DOI: 10.1017/jfm.2017.823.

[55] J.-C. Loiseau, B. R. Noack, and S. L. Brunton. "Sparse reduced-order modelling: sensor-based dynamics to full-state estimation". In: *Journal of Fluid Mechanics* 844 (June 2018). Publisher: Cambridge University Press, pp. 459–490. DOI: 10.1017/jfm.2018.147.

[56] B. Kramer et al. "Sparse Sensing and DMD-Based Identification of Flow Regimes and Bifurcations in Complex Flows". In: *SIAM Journal on Applied Dynamical Systems* 16.2 (Jan. 2017). Publisher: Society for Industrial and Applied Mathematics, pp. 1164–1196. DOI: 10.1137/15M104565X.

[57] P. Saini, C. M. Arndt, and A. M. Steinberg. "Development and evaluation of gappy-POD as a data reconstruction technique for noisy PIV measurements in gas turbine combustors". In: *Experiments in Fluids* 57.7 (July 12, 2016), p. 122. DOI: 10.1007/s00348-016-2208-7.

[58] J. Yu and J. S. Hesthaven. "Flowfield Reconstruction Method Using Artificial Neural Network". In: *AIAA Journal* 57.2 (Feb. 2019). Publisher: American Institute of Aeronautics and Astronautics, pp. 482–498. DOI: 10.2514/1.J057108.

[59] K. Lee and K. Carlberg. *Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders*. June 5, 2019. DOI: 10.48550/arXiv.1812.08373.

[60] K. Champion et al. "Data-driven discovery of coordinates and governing equations". In: *Proceedings of the National Academy of Sciences* 116.45 (Nov. 5, 2019), pp. 22445–22451. DOI: 10.1073/pnas.1906995116.

[61] M. Milano and P. Koumoutsakos. "Neural Network Modeling for Near Wall Turbulent Flow". In: *Journal of Computational Physics* 182.1 (Oct. 10, 2002), pp. 1–26. DOI: 10.1006/jcph.2002.7146.

[62] K. Fukami, K. Fukagata, and K. Taira. "Super-resolution reconstruction of turbulent flows with machine learning". In: *Journal of Fluid Mechanics* 870 (July 2019). Publisher: Cambridge University Press, pp. 106–120. DOI: 10.1017/jfm.2019.238.

[63] C. Hao et al. "Progress of Convolution Neural Networks in Flow Field Reconstruction". In: *Chinese Journal of Theoretical and Applied Mechanics* 54.9 (Sept. 18, 2022), pp. 2343–2360. DOI: 10.6052/0459-1879-22-130.

[64] N. B. Erichson et al. "Shallow neural networks for fluid flow reconstruction with limited sensors". In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 476.2238 (June 10, 2020). Publisher: Royal Society, p. 20200097. DOI: 10.1098/rspa.2020.0097.

[65] X. Peng et al. "A hybrid deep learning framework for unsteady periodic flow field reconstruction based on frequency and residual learning". In: *Aerospace Science and Technology* 141 (Oct. 1, 2023), p. 108539. DOI: 10.1016/j.ast.2023.108539.

[66] S. Barwey et al. "Using Machine Learning to Construct Velocity Fields from OH-PLIF Images". In: *Combustion Science and Technology* 194.1 (Jan. 2, 2022). Publisher: Taylor & Francis, pp. 93–116. DOI: 10.1080/00102202.2019.1678379.

[67] S. Barwey, V. Raman, and A. M. Steinberg. "Extracting information overlap in simultaneous OH-PLIF and PIV fields with neural networks". In: *Proceedings of the Combustion Institute* 38.4 (Jan. 1, 2021), pp. 6241–6249. DOI: 10.1016/j.proci.2020.06.180.

[68] X. Deng et al. "Dual-path flow field reconstruction for a scramjet combustor based on deep learning". In: *Physics of Fluids* 34.9 (Sept. 1, 2022), p. 095118. DOI: 10.1063/5.0111759.

[69] J. Li et al. "Integrated graph deep learning framework for flow field reconstruction and performance prediction of turbomachinery". In: *Energy* 254 (Sept. 1, 2022), p. 124440. DOI: 10.1016/j.energy.2022.124440.

[70] M. Raissi, P. Perdikaris, and G. E. Karniadakis. *Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations*. Nov. 28, 2017. DOI: 10.48550/arXiv.1711.10561.

[71] M. Raissi, P. Perdikaris, and G. E. Karniadakis. *Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations*. Nov. 28, 2017. DOI: 10.48550/arXiv.1711.10566.

[72] 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". In: *Journal of Computational Physics* 378 (Feb. 1, 2019), pp. 686–707. DOI: 10.1016/j.jcp.2018.10.045.

[73] Z. Mao, A. D. Jagtap, and G. E. Karniadakis. "Physics-informed neural networks for high-speed flows". In: *Computer Methods in Applied Mechanics and Engineering* 360 (Mar. 1, 2020), p. 112789. DOI: 10.1016/j.cma.2019.112789.

[74] X.-d. Bai, Y. Wang, and W. Zhang. "Applying physics informed neural network for flow data assimilation". In: *Journal of Hydrodynamics* 32.6 (Dec. 1, 2020), pp. 1050–1058. DOI: 10.1007/s42241-020-0077-2.

[75] M. Raissi, A. Yazdani, and G. E. Karniadakis. *Hidden Fluid Mechanics: A Navier-Stokes Informed Deep Learning Framework for Assimilating Flow Visualization Data*. Aug. 13, 2018. DOI: 10.48550/arXiv.1808.04327.

[76] M. Raissi, A. Yazdani, and G. E. Karniadakis. "Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations". In: *Science* 367.6481 (Feb. 28, 2020). Publisher: American Association for the Advancement of Science, pp. 1026–1030. DOI: 10.1126/science.aaw4741.

[77] H. Wang, Y. Liu, and S. Wang. "Dense velocity reconstruction from particle image velocimetry/particle tracking velocimetry using a physics-informed neural network". In: *Physics of Fluids* 34.1 (Jan. 25, 2022), p. 017116. DOI: 10.1063/5.0078143.

[78] S. Xu et al. "A practical approach to flow field reconstruction with sparse or incomplete data through physics informed neural network". In: *Acta Mechanica Sinica* 39.3 (Nov. 14, 2022), p. 322302. DOI: 10.1007/s10409-022-22302-x.

[79] S. Cai et al. "Flow over an espresso cup: inferring 3-D velocity and pressure fields from tomographic background oriented Schlieren via physics-informed neural networks". In: *Journal of Fluid Mechanics* 915 (May 2021). Publisher: Cambridge University Press, A102. DOI: 10.1017/jfm.2021.135.

[80] H. Ouyang et al. "Reconstruction of hydrofoil cavitation flow based on the chain-style physics-informed neural network". In: *Engineering Applications of Artificial Intelligence* 119 (Mar. 1, 2023), p. 105724. DOI: 10.1016/j.engappai.2022.105724.

[81] C. Wei and R. Ooka. "Indoor airflow field reconstruction using physics-informed neural network". In: *Building and Environment* 242 (Aug. 15, 2023), p. 110563. DOI: 10.1016/j.buildenv.2023.110563.

[82] E.-Z. Rui et al. "Reconstruction of 3D flow field around a building model in wind tunnel: a novel physics-informed neural network framework adopting dynamic prioritization self-adaptive loss balance strategy". In: *Engineering Applications of Computational Fluid Mechanics* 17.1 (Dec. 31, 2023). Publisher: Taylor & Francis, p. 2238849. DOI: 10.1080/19942060.2023.2238849.

[83] H. Eivazi and R. Vinuesa. *Physics-informed deep-learning applications to experimental fluid mechanics*. Mar. 29, 2022. DOI: 10.48550/arXiv.2203.15402.

[84] G. Kissas et al. "Machine learning in cardiovascular flows modeling: Predicting arterial blood pressure from non-invasive 4D flow MRI data using physics-informed neural networks". In: *Computer Methods in Applied Mechanics and Engineering* 358 (Jan. 2020), p. 112623. DOI: 10.1016/j.cma.2019.112623.

[85] N. Tathawadekar et al. "Physical Quantities Reconstruction in Reacting Flows with Deep Learning". In: *INTER-NOISE and NOISE-CON Congress and Conference Proceedings*. InterNoise22. Glasgow, Scotland: Institute of Noise Control Engineering, Feb. 1, 2023, pp. 1645–1656. DOI: https://doi-org.tudelft.idm.oclc.org/10.3397/IN_2022_0235.

[86] V. Yadav, M. Casel, and A. Ghani. "Physics-informed recurrent neural networks for linear and nonlinear flame dynamics". In: *Proceedings of the Combustion Institute* 39.2 (Jan. 1, 2023), pp. 1597–1606. DOI: 10.1016/j.proci.2022.08.036.

[87] N. Tathawadekar et al. *Hybrid Neural Network PDE Solvers for Reacting Flows*. Nov. 22, 2021. DOI: 10.48550/arXiv.2111.11185.

[88] C. Chi, S. Sreekumar, and D. Thévenin. "Data-driven discovery of heat release rate markers for premixed NH3/H2/air flames using physics-informed machine learning". In: *Fuel* 330 (Dec. 15, 2022), p. 125508. DOI: 10.1016/j.fuel.2022.125508.

[89] R. Nakazawa et al. "Species reaction rate modelling based on physics-guided machine learning". In: *Combustion and Flame* 235 (2022). DOI: 10.1016/j.combustflame.2021.111696.

[90] W. Ji et al. "Autonomous kinetic modeling of biomass pyrolysis using chemical reaction neural networks". In: *Combustion and Flame* 240 (2022). DOI: 10.1016/j.combustflame.2022.111992.

[91] X. Wang et al. "Flow-field reconstruction in rotating detonation combustor based on physics-informed neural network". In: *Physics of Fluids* 35.7 (July 11, 2023), p. 076109. DOI: 10.1063/5.0154979.

[92] F. Darlik and B. Peters. "Reconstruct the biomass particles fields in the particle-fluid problem using continuum methods by applying the physics-informed neural network". In: *Results in Engineering* 17 (Mar. 1, 2023), p. 100917. DOI: 10.1016/j.rineng.2023.100917.

[93] D. de Boer. "Velocity Reconstruction in Pool Fires using Physics-Informed Machine Learning". Masters thesis. Delft University of Technology, Dec. 13, 2022.

[94] A. Daw et al. *Mitigating Propagation Failures in PINNs using Evolutionary Sampling*. Publication Title: arXiv e-prints ADS Bibcode: 2022arXiv220702338D Type: article. July 1, 2022. DOI: 10.48550/arXiv.2207.02338.

[95] S. Wang, Y. Teng, and P. Perdikaris. "Understanding and Mitigating Gradient Flow Pathologies in Physics-Informed Neural Networks". In: *SIAM Journal on Scientific Computing* 43.5 (Jan. 2021). Publisher: Society for Industrial and Applied Mathematics, A3055–A3081. DOI: 10.1137/20M1318043.

[96] S. Wang, X. Yu, and P. Perdikaris. "When and why PINNs fail to train: A neural tangent kernel perspective". In: *Journal of Computational Physics* 449 (Jan. 15, 2022), p. 110768. DOI: 10.1016/j.jcp.2021.110768.

[97] A. S. Krishnapriyan et al. *Characterizing possible failure modes in physics-informed neural networks*. Nov. 11, 2021. DOI: 10.48550/arXiv.2109.01050.

[98] Q. Zeng et al. *Competitive Physics Informed Networks*. Oct. 12, 2022. DOI: 10.48550/arXiv.2204.11144.

[99] R. van der Meer, C. Oosterlee, and A. Borovykh. *Optimally weighted loss functions for solving PDEs with Neural Networks*. Mar. 24, 2021. DOI: 10.48550/arXiv.2002.06269.

[100] M. A. Nabian, R. J. Gladstone, and H. Meidani. "Efficient training of physics-informed neural networks via importance sampling". In: *Computer-Aided Civil and Infrastructure Engineering* 36.8 (Apr. 9, 2021), pp. 962–977. DOI: 10.1111/mice.12685.

[101] M. D. Zeiler. *ADADELTA: An Adaptive Learning Rate Method*. Dec. 22, 2012. DOI: 10.48550/arXiv.1212.5701.

[102] I. Loshchilov and F. Hutter. *SGDR: Stochastic Gradient Descent with Warm Restarts*. May 3, 2017. DOI: 10.48550/arXiv.1608.03983.

[103] C. L. Wight and J. Zhao. *Solving Allen-Cahn and Cahn-Hilliard Equations using the Adaptive Physics Informed Neural Networks*. July 8, 2020. DOI: 10.48550/arXiv.2007.04542.

[104] A. Daw et al. *Rethinking the Importance of Sampling in Physics-informed Neural Networks*. July 5, 2022. DOI: 10.48550/arXiv.2207.02338.

[105] C. Wu et al. "A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks". In: *Computer Methods in Applied Mechanics and Engineering* 403 (Jan. 1, 2023), p. 115671. DOI: 10.1016/j.cma.2022.115671.

[106] A. D. Jagtap, K. Kawaguchi, and G. E. Karniadakis. "Adaptive activation functions accelerate convergence in deep and physics-informed neural networks". In: *Journal of Computational Physics* 404 (Mar. 1, 2020), p. 109136. DOI: 10.1016/j.jcp.2019.109136.

[107] A. D. Jagtap, K. Kawaguchi, and G. E. Karniadakis. "Locally adaptive activation functions with slope recovery term for deep and physics-informed neural networks". In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 476.2239 (July 2020), p. 20200334. DOI: 10.1098/rspa.2020.0334.

[108] Jagtap, Ameya D and G. E. M. Karniadakis. "Extended Physics-Informed Neural Networks (XPINNs): A Generalized Space-Time Domain Decomposition Based Deep Learning Framework for Nonlinear Partial Differential Equations". In: *Communications in Computational Physics* 28.5 (June 2020), pp. 2002–2041. DOI: 10.4208/cicp.OA-2020-0164.

[109] B. Moseley, A. Markham, and T. Nissen-Meyer. "Finite basis physics-informed neural networks (FBPINNs): a scalable domain decomposition approach for solving differential equations". In: *Advances in Computational Mathematics* 49.4 (July 31, 2023), p. 62. DOI: 10.1007/s10444-023-10065-9.

[110] S. Wang, S. Sankaran, and P. Perdikaris. *Respecting causality is all you need for training physics-informed neural networks*. Mar. 14, 2022. DOI: 10.48550/arXiv.2203.07404.

[111] The OpenFOAM Foundation. *OpenFOAM-7*. 2019. URL: https://openfoam.org.

[112] G. K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge Mathematical Library. Cambridge University Press, 2000.

[113] T. Salimans and D. P. Kingma. *Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks*. June 3, 2016. DOI: 10.48550/arXiv.1602.07868.

[114] M. P. Sitte. "Modelling of Spray Combustion with Doubly Conditional Moment Closure". PhD thesis. University of Cambridge, Sept. 25, 2018.

[115] E. Fernández-Tarrazo et al. "A simple one-step chemistry model for partially premixed hydrocarbon combustion". In: *Combustion and Flame* 147.1 (Oct. 1, 2006), pp. 32–38. DOI: 10.1016/j.combustflame.2006.08.001.

[116] D. P. Kingma and J. Ba. *Adam: A Method for Stochastic Optimization*. Jan. 29, 2017. DOI: 10.48550/arXiv.1412.6980.

[117] H. Baum and B. Mccaffrey. "Fire Induced Flow Field - Theory And Experiment". In: *Fire Safety Science* 2 (1989), pp. 129–148. DOI: 10.3801/IAFSS.FSS.2-129.

A

Formulation of the continuity and Navier-Stokes equations

As discussed in Chapter 5, the momentum loss used to train the PINNs is evaluated by computing the residuals of the governing equations of the pool fire flame. These are the continuity equation and the Cauchy momentum equation, from which the Navier-Stokes equations are derived. To the best of the author's knowledge the specific formulation used in this thesis is not available in any previous work. For this reason the necessary formulation has been derived directly from the general conservation laws. This derivation is presented here for the sake of completeness and to ensure replicability of the results.

The general conservation laws that govern the dynamics of the pool fire flame have been introduced in Chapter 5. They are:

$$\text{Continuity equation:} \quad \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{V}) = 0 \tag{A.1}$$

$$\text{Cauchy momentum equation:} \quad \rho \frac{D\boldsymbol{V}}{Dt} = \nabla \cdot \boldsymbol{\sigma} + \rho \boldsymbol{g} = -\nabla p + \nabla \cdot \boldsymbol{\tau} + \rho \boldsymbol{g} \tag{A.2}$$

These equations must be formulated in cylindrical coordinates and simplified using the relevant assumptions, introduced in Chapter 5 and repeated below:

Table A.1: Assumptions made in the derivation of the momentum loss

Assumption	**Effect**
Steady flow	Time derivatives are zero
Compressible flow	Density gradients are not neglected
Axisymmetric flow	Tangential derivatives are zero
Zero tangential velocity	Tangential velocity is zero
Only body force is gravity	No other body forces
Non-uniform shear viscosity	Viscosity computed from Sutherland's law
Newtonian fluid	Provides a constitutive relation for the stress tensor

Several vector identities in cylindrical coordinates are used in this process. Firstly, the divergence of the velocity vector in cylindrical coordinates is given by:

$$\nabla \cdot \boldsymbol{V} = \frac{1}{r}\frac{\partial}{\partial r}(rV_r) + \frac{1}{r}\frac{\partial V_\theta}{\partial \theta} + \frac{\partial V_z}{\partial z} \tag{A.3}$$

Applying the assumption of axisymmetric flow with zero tangential velocity, and denoting the three velocity components along the (r, θ, z) directions by (v, w, u) this reduces to:

$$\nabla \cdot \boldsymbol{V} = \frac{1}{r}\frac{\partial}{\partial r}(rv) + \frac{\partial u}{\partial z} \tag{A.4}$$

Additionally, the divergence of a tensor field $\mathbf{S}$ in cylindrical coordinates is given by:

$$\begin{aligned}\nabla \cdot \boldsymbol{S} =& \frac{\partial S_{rr}}{\partial r}\mathbf{e}_r + \frac{\partial S_{r\theta}}{\partial r}\mathbf{e}_\theta + \frac{\partial S_{rz}}{\partial r}\mathbf{e}_z \\ &+ \frac{1}{r}\left[\frac{\partial S_{\theta r}}{\partial \theta} + (S_{rr} - S_{\theta\theta})\right]\mathbf{e}_r + \frac{1}{r}\left[\frac{\partial S_{\theta\theta}}{\partial \theta} + (S_{r\theta} + S_{\theta r})\right]\mathbf{e}_\theta + \frac{1}{r}\left[\frac{\partial S_{\theta z}}{\partial \theta} + S_{rz}\right]\mathbf{e}_z \\ &+ \frac{\partial S_{zr}}{\partial z}\mathbf{e}_r + \frac{\partial S_{z\theta}}{\partial z}\mathbf{e}_\theta + \frac{\partial S_{zz}}{\partial z}\mathbf{e}_z\end{aligned} \tag{A.5}$$

where $(\mathbf{e}_r, \mathbf{e}_\theta, \mathbf{e}_z)$ are the unit vectors along the (r, θ, z) directions, respectively. Applying the relevant assumptions and vector identities, the residual of the continuity equation (Equation A.1) reduces to:

$$\begin{aligned}\mathcal{E}_{\text{cont}} &= \rho\left[\frac{\partial V_r}{\partial r} + \frac{\partial V_z}{\partial z} + \frac{V_r}{r}\right] + V_r\frac{\partial \rho}{\partial r} + V_z\frac{\partial \rho}{\partial z} \\ &= \rho\left[\frac{\partial v}{\partial r} + \frac{\partial u}{\partial z} + \frac{v}{r}\right] + v\frac{\partial \rho}{\partial r} + u\frac{\partial \rho}{\partial z} \quad = 0\end{aligned} \tag{A.6}$$

In order to derive the Navier-Stokes equations from the Cauchy momentum equation (Equation A.2), a constitutive relation is needed to relate the stress tensor to the gradients of the velocity. For a Newtonian fluid this constitutive relation takes the form:

$$\sigma_{ij} = \delta_{ij}[-p + (\Lambda - \frac{2}{3}\mu)\Phi] + 2\mu e_{ij} \tag{A.7}$$

where δ_{ij} is the Kronecker delta, $\Phi = \nabla \cdot \boldsymbol{V}$ is the dilatation, μ is the viscosity, e_{ij} is the stress rate tensor, and Λ is the second coefficient of viscosity, also known as the volume or dilatation viscosity, which is assumed to be $\Lambda = -2\mu/3$. Using this constitutive law the components of the stress tensor σ in cylindrical coordinates can be written as:

$$\sigma_{rr} = -p + 2\mu\frac{\partial V_r}{\partial r} + \Lambda\left\{\frac{1}{r}\frac{\partial\left(rV_r\right)}{\partial r} + \frac{1}{r}\frac{\partial V_\theta}{\partial \theta} + \frac{\partial V_z}{\partial z}\right\} \tag{A.8}$$

$$\sigma_{\theta\theta} = -p + 2\mu\left(\frac{1}{r}\frac{\partial V_\theta}{\partial \theta} + \frac{V_r}{r}\right) + \Lambda\left\{\frac{1}{r}\frac{\partial\left(rV_r\right)}{\partial r} + \frac{1}{r}\frac{\partial V_\theta}{\partial \theta} + \frac{\partial V_z}{\partial z}\right\} \tag{A.9}$$

$$\sigma_{zz} = -p + 2\mu\frac{\partial V_z}{\partial z} + \Lambda\left\{\frac{1}{r}\frac{\partial\left(rV_r\right)}{\partial r} + \frac{1}{r}\frac{\partial V_\theta}{\partial \theta} + \frac{\partial V_z}{\partial z}\right\} \tag{A.10}$$

$$\sigma_{r\theta} = \sigma_{\theta r} = \mu\left(\frac{1}{r}\frac{\partial V_r}{\partial \theta} + \frac{\partial V_\theta}{\partial r} - \frac{V_\theta}{r}\right) \tag{A.11}$$

$$\sigma_{rz} = \sigma_{zr} = \mu\left(\frac{\partial V_r}{\partial z} + \frac{\partial V_z}{\partial r}\right) \tag{A.12}$$

$$\sigma_{\theta z} = \sigma_{z\theta} = \mu\left(\frac{1}{r}\frac{\partial V_z}{\partial \theta} + \frac{\partial V_\theta}{\partial z}\right) \tag{A.13}$$

The dynamic viscosity μ is computed from the temperature field using Sutherland's law, as done by Sitte and Doan [26]:

$$\mu = A_s\sqrt{T}/\left(1 + T_s/T\right) \tag{A.14}$$

where $A_s = 1.671 \times 10^{-6}$ kg/(m s K$^{1/2}$) and $T_s = 170.67$ K. Lastly, the material derivative of velocity in cylindrical coordinates is given by:

$$\begin{aligned}\frac{D\boldsymbol{V}}{Dt} =& \left[\frac{\partial V_r}{\partial t} + V_r\frac{\partial V_r}{\partial r} + \frac{V_\theta}{r}\frac{\partial V_r}{\partial \theta} - \frac{V_\theta^2}{r} + V_z\frac{\partial V_r}{\partial z}\right]\mathbf{e}_r \\ &+ \left[\frac{\partial V_\theta}{\partial t} + V_r\frac{\partial V_\theta}{\partial r} + \frac{V_\theta}{r}\frac{\partial V_\theta}{\partial \theta} + \frac{V_\theta V_r}{r} + V_z\frac{\partial V_\theta}{\partial z}\right]\mathbf{e}_\theta \\ &+ \left[\frac{\partial V_z}{\partial t} + V_r\frac{\partial V_z}{\partial r} + \frac{V_\theta}{r}\frac{\partial V_z}{\partial \theta} + V_z\frac{\partial V_z}{\partial z}\right]\mathbf{e}_z\end{aligned} \tag{A.15}$$

The Cauchy momentum equation (Equation A.2) can now be reduced to the Navier-Stokes equations in r and z by combining the expression for the material derivative of velocity (Equation A.15), the

components of the stress tensor (Equations A.8–A.13), the expression for the divergence of a tensor field (Equation A.5) and finally applying the assumptions listed in Table A.1. The resulting residuals of the Navier-Stokes equations in the r and z directions are:

$$\begin{aligned}\mathcal{E}_{\text{R-mom}} &= \rho\left[V_r\frac{\partial V_r}{\partial r} + V_z\frac{\partial V_r}{\partial z}\right] + \frac{\partial p}{\partial r} + \frac{1}{r}(\tau_{\theta\theta} - \tau_{rr}) - \frac{\partial \tau_{rr}}{\partial r} - \frac{\partial \tau_{zr}}{\partial z} \\ &= \rho\left[v\frac{\partial v}{\partial r} + u\frac{\partial v}{\partial z}\right] + \frac{\partial p}{\partial r} + \frac{1}{r}(\tau_{\theta\theta} - \tau_{rr}) - \frac{\partial \tau_{rr}}{\partial r} - \frac{\partial \tau_{zr}}{\partial z} = 0\end{aligned} \tag{A.16}$$

$$\begin{aligned}\mathcal{E}_{\text{Z-mom}} &= \rho\left[V_r\frac{\partial V_z}{\partial r} + V_z\frac{\partial V_z}{\partial z}\right] + \frac{\partial p}{\partial z} - \frac{\tau_{rz}}{r} - \frac{\partial \tau_{rz}}{\partial r} - \frac{\partial \tau_{zz}}{\partial z} - \rho g_z \\ &= \rho\left[v\frac{\partial u}{\partial r} + u\frac{\partial u}{\partial z}\right] + \frac{\partial p}{\partial z} - \frac{\tau_{rz}}{r} - \frac{\partial \tau_{rz}}{\partial r} - \frac{\partial \tau_{zz}}{\partial z} - \rho g_z = 0\end{aligned} \tag{A.17}$$

The residuals of the Navier-Stokes equations (Equation A.16 and Equation A.17) and the continuity equation (Equation A.6) are used to evaluate the momentum loss used to train the PINNs. These residuals are defined everywhere in the domain and are therefore functions of the spatial coordinates (r, z). In order to compute the momentum loss the mean squared errors of these residuals are computed and then added together. The verification of the formulation of the continuity and Navier-Stokes derived here is discussed in detail in Chapter 5 (Section 5.4).