TUM School of Computation, Information and Technology
Technische Universität München

Modelling Flame Dynamics with Deep Learning Methods

Nilam Tathawadekar

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung des akademischen Grades einer

Doktorin der Ingenieurwissenschaften (Dr.-Ing.)

genehmigten Dissertation.

Vorsitz: Prof. Dr. Debarghya Ghoshdastidar

Prüfende der Dissertation:

1. Prof. Dr. Nils Thuerey
2. Prof. Dr. Wolfgang Polifke
3. Assistant Prof. Dr. Nguyen Anh Khoa Doan

Die Dissertation wurde am 07.12.2023 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 10.07.2024 angenommen.

Abstract

Modelling flame dynamics is an essential topic in the combustion community and is challenging because of the complicated, nonlinear dynamics involved. In this dissertation, we will explore neural network-based modelling of the flame dynamics for various applications, such as describing the nonlinear flame response of premixed laminar flame, forecasting flame dynamics over a long time range, and controlling inlet velocity to achieve the desired flame shape. The primary goal of our work is to explore the applicability of neural network (NN) models in assisting existing computational fluid dynamics (CFD) solvers to tackle the challenges in flame dynamics modelling.

Modelling the nonlinear flame response of a premixed flame comes with a very high computational cost due to the large number of CFD simulations required. We propose a neural network-based model to reliably learn the nonlinear flame response of a laminar premixed flame while carrying out only one unsteady CFD simulation. The system is excited with a broadband, low-pass filtered velocity signal that exhibits a uniform distribution of amplitudes within a predetermined range. The obtained time series of flow velocity upstream of the flame and heat release rate fluctuations are used to train the nonlinear model using a multi-layer perception (MLP). In addition to accurately predicting the flame describing function (FDF), the trained neural network model also captures the presence of higher harmonics in the flame response. As a result, when coupled with an acoustic solver, the obtained neural network model is better suited than a classical FDF model to predict limit cycle oscillations characterized by more than one frequency. Having demonstrated the potential of neural networks in modelling the flame response of laminar flames, their application to turbulent flames is important for practical relevance. Modelling the flame response of turbulent flames via data-driven approaches is a challenging task due, among others, to the presence of combustion noise. Our work analyses the ability of neural network models to evaluate the linear and nonlinear flame response of turbulent flames. Initially, the NN is trained to evaluate and interpolate the linear flame response model when presented with data obtained at various thermal conditions. Later, the NN is trained to infer the nonlinear flame response model when presented with a time series exhibiting sufficient large amplitudes.

Further, we study the problem of predicting flame dynamics for a given initial condition when the underlying chemistry is unknown. Modelling complex dynamical systems

with only partial knowledge of their physical mechanisms is a crucial problem across all scientific and engineering disciplines. Purely data-driven (PDD) approaches, which only make use of an artificial neural network and data, often fail to accurately simulate the evolution of the system dynamics over a sufficiently long time and in a physically consistent manner. Therefore, we propose a hybrid approach that uses a neural network model in combination with an incomplete partial differential equations (PDE) solver that provides known, but incomplete physical information. In this study, we demonstrate that the results obtained from the incomplete PDEs can be efficiently corrected at every time step by the proposed hybrid neural network – PDE solver model, so that the effect of the unknown physics present in the system is correctly accounted for. For validation purposes, the obtained simulations of the hybrid model are successfully compared against results coming from the complete set of PDEs describing the full physics of the considered system. We demonstrate the validity of the proposed approach on a reactive flow, an archetypal multi-physics system that combines fluid mechanics and chemistry, the latter being the physics considered unknown. Experiments are made on planar and Bunsen-type flames at various operating conditions. The hybrid neural network - PDE approach correctly models the flame evolution of the cases under study for significantly long time windows, yields improved generalization, and allows for larger simulation time steps. Lastly, we show that the resulting hybrid solver provides a flexible building block for controlling the evolution of flame shapes. The control of reacting flows is very challenging due to the chaotic nature of flows, the strong nonlinearity of the chemical reactions and the complex interplay between flow and chemistry. The hybrid solver can efficiently predict the states of the reactive flows and readily enable the control of the flow inlet velocities to arrive at a desired flame shape. Therefore, we employ a pre-trained hybrid solver to train a second controller network that learns to steer the flame simulation such that the observed states are matched.

We demonstrate that the neural networks are powerful models and when systematically combined with the appropriate fluid solver can be successfully applied to model flame dynamics. Our methods showcase diverse possibilities of neural networks in modelling reactive flow simulations, forecasting the flame dynamics for unseen operating conditions, and controlling the flame dynamics.

Zusammenfassung

Das Verständnis von Flammendynamiken ist ein essenzielles Thema in der Verbrennungsbranche und ihre Modellierung stellt aufgrund der komplizierten, nichtlinearen Dynamik eine Herausforderung dar. In dieser Dissertation werden wir die auf neuronalen Netzen basierte Modellierung der Flammendynamik für verschiedene Anwendungen untersuchen, wie z. B. die Beschreibung des nichtlinearen Verbrennungsverhaltens einer vorgemischten laminaren Flamme, die Vorhersage der Flammendynamik über einen langen Zeitraum und die Steuerung der Einlassgeschwindigkeit, um die gewünschte Flammenform zu erreichen. Das Kernziel unserer Arbeit ist die Erforschung der Anwendbarkeit von neuronalen Netzen zur Unterstützung bestehender Methoden in der numerischen Strömungsmechanik, um diese Herausforderungen bei der Modellierung der Flammendynamik zu bewältigen.

Die Modellierung des nichtlinearen Flammenverhaltens einer vorgemischten Flamme erfordert zahlreiche CFD-Simulationen und deren Berechnung ist sehr kostenintensiv. Wir entwickeln ein Modell basierend auf neuronalen Netzen, das das nichtlineare Verbrennungsverhalten einer laminaren vorgemischten Flamme zuverlässig erlernt, während nur eine einzige instationäre CFD-Simulation durchgeführt wird. Das System wird mit einem tiefpassgefilterten Breitband-Geschwindigkeitssignal angeregt, welches eine gleichmäßige Verteilung der Amplituden innerhalb eines vorgegebenen Bereichs aufweist. Die resultierend Zeitsequenzen des Strömungsfeldes stromaufwärts der Flamme und die Schwankungen der Wärmeabgabe werden zum Trainieren des nichtlinearen Modells verwendet, welches auf einem Multilayer Perceptron basiert. Neben der genauen Vorhersage der Flammenbeschreibungsfunktion erfasst das trainierte neuronale Netz auch die höheren Harmonischen in der Flammenantwort. In Verbindung mit einer akustischen Simulation ist das erhaltene neuronale Netz daher besser als ein klassisches FDF-Modell in der Vorhersage von Grenzzyklusschwingungen, die durch ein breites Frequenzspektrum gekennzeichnet sind. Nachdem das Potenzial neuronaler Netze für die Modellierung laminarer Flammen aufgezeigt wurde, ist ihre Anwendung auf turbulente Flammen von praktischer Bedeutung. Die Schwierigkeit der datenbasierten Modellierung des turbulenten Flammenverhaltens zeichnet sich unter anderem durch Verbrennungsrauschen in den Daten aus. Unsere Arbeit untersucht die Fähigkeit von neuronalen Netzen ebendieses lineare und nichtlinearen Flammenverhaltens turbulenter Flammen auszuwerten. Zunächst

wird das neuronale Netz darauf trainiert, das lineare Flammenantwortmodell auszuwerten und zu interpolieren. Die Grundlage hierfür bilden Daten, die bei verschiedenen thermischen Bedingungen gewonnen wurden. Später wird das NN darauf trainiert, das nichtlineare Flammenantwortmodell abzuleiten, wenn Zeitreihen mit ausreichend großen Amplituden vorliegen.

Außerdem untersuchen wir das Problem der Vorhersage der Flammendynamik für eine gegebene Anfangsbedingung und unbekannte zugrunde liegende Chemie. Die Modellierung komplexer dynamischer Systeme mit nur unvollständiger Kenntnis ihrer physikalischen Mechanismen ist ein fundamentales Problem in allen wissenschaftlichen und technischen Disziplinen. Rein datengetriebene Ansätze, bei denen nur ein künstliches neuronales Netz und Daten verwendet werden, sind oft nicht in der Lage, die Entwicklung der Systemdynamik über einen ausreichend langen Zeitraum und in physikalisch konsistenter Weise zu simulieren. Daher schlagen wir einen hybriden Ansatz vor, der ein neuronales Netz in Kombination mit einem numerischen Verfahren für partielle Differentialgleichungen (PDG) verwendet, der bekannte, aber unvollständige physikalische Informationen liefert. In dieser Studie zeigen wir, dass die aus den unvollständigen PDGs gewonnenen Ergebnisse in jedem Zeitschritt durch das vorgeschlagene hybride Modell aus neuronalem Netz und PDG-Löser effizient korrigiert werden können, sodass die Auswirkungen der unbekannten physikalischen Gegebenheiten im System korrekt berücksichtigt werden. Zu Validierungszwecken werden die mit dem hybriden Modell erzielten Simulationen mit den Ergebnissen verglichen, die sich aus dem vollständigen System von PDGs ergeben, die die gesamte Physik des betrachteten Systems beschreiben. Wir demonstrieren die Gültigkeit des vorgeschlagenen Ansatzes an einer reaktiven Strömung. Diese stellt ein grundlegenden Multiphysiksystem das, das Strömungsmechanik und Chemie kombiniert, wobei letztere als die unbekannte Physik gilt. Es werden Experimente mit planaren und Bunsen-Flammen unter verschiedenen Betriebsbedingungen durchgeführt. Der hybride Ansatz aus neuronalem Netz und PDG modelliert die Flammenentwicklung in den untersuchten Fällen für deutlich verlängerte Zeitfenster korrekt, bietet eine verbesserte Generalisierung und ermöglicht größere Simulationszeitschritte. Schlussendlich zeigen wir, dass die resultierende hybride Methode einen flexiblen Baustein für die Steuerung der Flammenentwicklung darstellt. Die Steuerung von reaktiven Strömungen ist aufgrund der chaotischen Natur der Strömungen, der starken Nichtlinearität der chemischen Reaktionen und des komplexen Zusammenspiels zwischen Strömung und Chemie eine große Herausforderung. Der hybride Solver kann die Zustände der reaktiven Strömungen effizient vorhersagen und ermöglicht die Steuerung der Einlassgeschwindigkeiten der Strömung, um eine gewünschte Flammenform zu erreichen. Daher verwenden wir einen vortrainierten hybriden Solver, um ein zweites Reglernetzwerk zu trainieren, das lernt, die Flammensimulation so zu steuern, dass die beobachteten Zustände übereinstimmen.

Wir zeigen, dass neuronale Netze leistungsstarke Modelle sind und bei systematischer Kombination mit einem geeigneten numerischen Verfahren erfolgreich zur Modellierung der Flammendynamik eingesetzt werden können. Unsere Methoden zeigen die vielfältigen Möglichkeiten neuronaler Netze bei der Modellierung reaktiver Strömungssimulationen, der Vorhersage der Flammendynamik für unbekannte Betriebsbedingungen und der Steuerung der Flammendynamik.

Acknowledgements

As I stand on the summit of this academic endeavour, I owe the view not only to my own steps but to the steady hands and guiding spirits of those who accompanied me on this ascent. This note of gratitude only begins to describe my appreciation of a few of those many, who have contributed to the successful completion of this dissertation along with my doctoral journey.

First and foremost, I am immensely thankful to my advisor, Prof. Nils Thuerey, for his steadfast support, guidance, and patience throughout this research endeavour. His expertise on the subject, dedication to this project, and structured feedback from time to time have been invaluable in shaping the direction and quality of this dissertation. I am grateful to him for agreeing to guide me and for the substantial time and effort invested in overseeing my academic progress.

A big thanks goes to Dr. Nguyen Anh Khoa Doan and Dr. Camilo F. Silva, who supported me with their knowledge during my research. Their advice in the field of combustion and flame dynamics was of great assistance. Their constructive feedback has played a pivotal role in refining the content and methodology of this study. I am truly fortunate to have had such co-authors and mentors. I would also like to acknowledge the members of my dissertation committee, Prof. Wolfgang Polifke and Dr. Nguyen Anh Khoa Doan, for their time and expertise in reviewing and critiquing this work.

I am grateful to my colleagues and peers who have provided valuable insights and stimulating discussions that have enriched the development of my research. Your encouragement and support have been a source of motivation. I am thankful to Dr. Mark Lynass, Dr. Lorenz Drack, Dr. Maximilian Zahn, Dr. Krzysztof Kostrzewa, Dr. Günter Wilfert, Dr. Thomas Ripplinger, Michael McCartney, my colleagues at GE Aviation, Garching, for helping me shape up the initial part of my work, to name a few. I would also like to thank my colleagues at the Technical University of Munich, Björn List, Philipp Holl, Liwei Chen, Benjamin Holzschuh, You Xie, Georg Kohl, Kiwon Um, Lukas Prantl, Patrick Schnell, Marton Szep, Erik Franz, Qiang Liu, Stephan Rasp, Felix Köhler, Brener d'Lélis, Rene Winchenbach, Shuvayan Brahmachary, Alexander Eder, Sagar Kulkarni and Alper

Ösün. A special thanks to Susanne Weitz and Sebastian Wohner for their support in all the administrative tasks.

The one person who has probably been the sole witness of the highs and lows of this enriching journey, my significant other - Anveet, is the one person who I am the most indebted to. Anveet, you have been my guidepost and my anchor through the challenging times and my biggest cheerleader through it all. To say the least, this would not have been possible without you.

There is no support system like family. I take this opportunity to express my gratitude to my Aai-Baba (Nilakshi-Nandkishor) and my brother Nilesh for standing by me throughout my life, right up to this point. Thank you for keeping me grounded and being my pillar of strength. I would also like to take this opportunity to thank others in my corner - my friends, who helped me maintain an equilibrium between work and pleasure, but especially Rucha, who made sure that this journey was filled with fun.

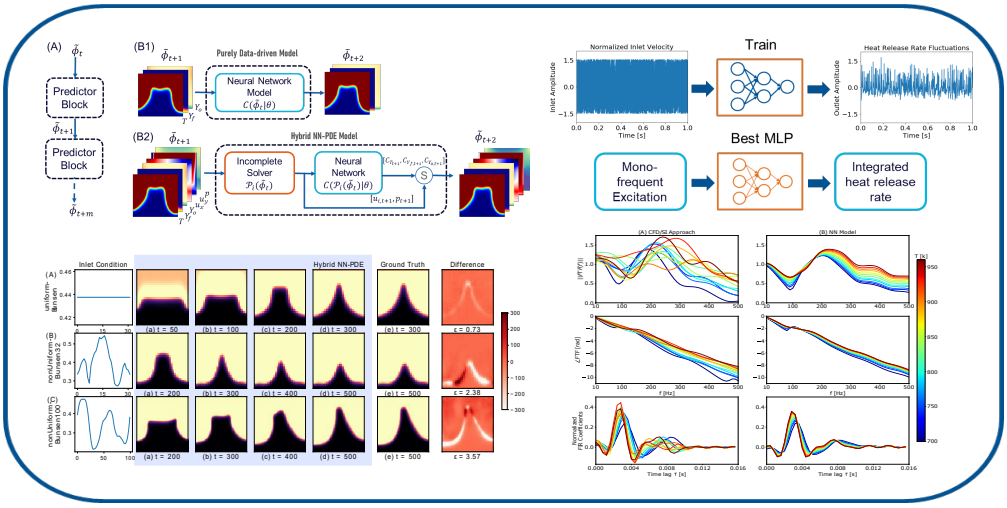
And then there are those countless nameless others, who have done their bit to see me to the finish line. Thank you all for being a part of this significant milestone in my academic journey.

Contents

List of Figures

List of Tables

1 Introduction



1.1 Motivation

Climate change and environmental degradation are an existential threat to the world. The effects of climate change and global warming are raising new awareness of their influence on our everyday existence. Power generation and transportation are some of the significant contributors to greenhouse gas emissions. Indeed, as of 2022, in the European Union (EU), 70 % of the energy is still produced by the combustion of fossil fuels [8, 9]. Thus, great efforts need to be spent to make combustion greener and safer than in the past or find substitute solutions for energy conversion or power generation. Among the different sectors, aviation and power generation are challenging sectors to address due to their high energy consumption and the absence of easily expandable electrification solutions [10]. Gas turbines are a dominant technology in the fields of power generation and aviation. They are known for their efficiency and flexibility in converting fuel into

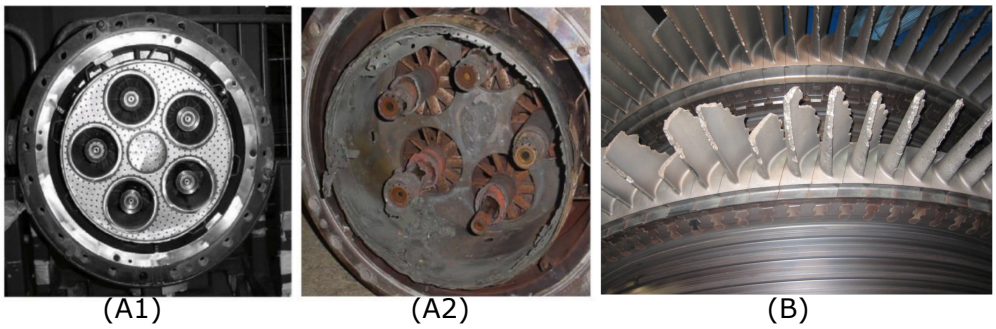


Figure 1.1: Effect of combustion instabilities. Burner assembly (A1) before and (A2) after thermo-acoustic instability [1]. (B) Turbine blade damage caused by combustion instability.

mechanical energy. However, traditional gas turbines that burn fossil fuels contribute to greenhouse gas emissions, which are a major driver of the climate crisis due to the release of carbon dioxide and other pollutants. As the European Green Deal aims for no net emissions of greenhouse gases by 2050, the development of gas turbines needs substantial research on fuels, combustion techniques and more efficient burners with low emissions of greenhouse gases, pollutants and noise. This includes technologies like premixed lean-burn combustion and the use of alternative fuels like hydrogen.

These combustion concepts lead to more complex dynamics and coupling between flame and turbulence. Achieving such concepts requires a redesign of combustion systems; for example, hydrogen fueled combustion requires flame stabilization and high mixing of fuel and air [11, 12]. However, these redesigns give way to instabilities. Lean combustion systems are prone to thermo-acoustic instabilities in which self-amplifying feedback establishes between the unsteady heat release generated by the flame and the acoustic pressure fluctuations [13]. Without proper mixing, hydrogen fueled flames can be difficult to control and stabilize, leading to issues like flame blowoff, flashbacks (flame propagating back into the fuel-air mixture), and flame instability [14, 15]. These instabilities may cause severe damage to the engine such that the unconditional avoidance of such instabilities limits the operational flexibility of engines. During the development and testing of new combustion technologies, instabilities are observed (examples shown in figure 1.1), and corrective actions are taken to prevent damages. Generally, it is a retrofit or redesign activity specific to the engine designs [16]. Although these methods are effective, they are neither fool-proof nor cost efficient. Therefore, there is a need to develop tools that can predict these instabilities in the early design phase of the combustor.

Numerical analysis plays a central role in understanding and modelling combustion systems. Reactive flows are an archetypal multi-physics, multi-scale phenomena involving fluid mechanics and chemistry. Resolving the Navier-Stokes equations lies at the core of these problems, where additionally the transport of different species of relevance must also be accounted for, together with their production or consumption often following complex reaction mechanisms [17]. Reactive flows are modelled within the framework

of computational fluid dynamics (CFD) to analyze or predict combustion instabilities. Due to the wide range of length and time scales involved, huge computational resources are required to enable numerical analysis. Despite the improvement in computation hardware and algorithms, it is still highly challenging to efficiently analyse the combustion systems. Therefore, it is a very important direction of research for the combustion community to develop highly accurate and efficient reduced order models to describe and replace a certain aspect of the overall system. Deep learning methods have achieved rapid developments and great success in many fields, such as computer vision [18, 19, 20] and natural language processing [21, 22, 23]. While these are very interesting techniques, their application to the field of fluid dynamics and, in particular, to reactive flow simulations is challenging. Since ensuring the conservation of physical quantities plays a vital role, obtaining physically accurate results becomes one of the key challenges. Researchers have just begun to explore applications of deep learning models in the field of fluid dynamics [24, 25, 26, 27, 28]. Some early works [29, **?**, 30] apply neural network models to control combustion instabilities of Rijke tube. This dissertation focuses on more sophisticated premixed flame setups and the integration of neural network models with the CFD solver. With advances in numerical simulations, data required to predict flame dynamics is made readily available. These applications help to reveal the ability and the generality of the proposed methods to model and control the reactive flow simulations. This work focuses on modelling flame dynamics using deep learning methods. Flame dynamics is the dynamics response of flame shape, flame position and flame release rate to flow perturbations [31]. This work explores how deep learning methods can be used for different areas of flame dynamics. It includes study of flame propagation i.e. how flame front moves through a combustible mixture; study of flame instabilities i.e. predicting conditions for which flame becomes unstable; study of flame structure i.e. predicting shape of a flame, including its temperature profile and species distribution.

1.2 Scope of the work

In this dissertation, chapter 1 motivates the need for flame dynamics modelling. Chapter 2 discusses relevant background information on the topics of fluid dynamics and deep learning models. An overview of related work is presented in chapter 3. We propose a neural network-based approach to model the nonlinear flame response of Bunsen-type laminar flame in chapter 4. Chapter 5 extends the applicability of the proposed approach to more challenging and practical turbulent flames. Then we demonstrate, in chapter 6, the applicability of a convolutional neural network (CNN) based model to reconstruct fields of reactive flow simulation. In chapter 7, we propose a hybrid framework combining a neural network model with a differentiable PDE solver to model flame dynamics of a Bunsen-type flame. We illustrate the application of learned models to control flame dynamics. Finally, we discuss the potential outlook for future work based on the proposed methods in chapter 8. An overview of the dissertation structure is shown in figure 1.2.

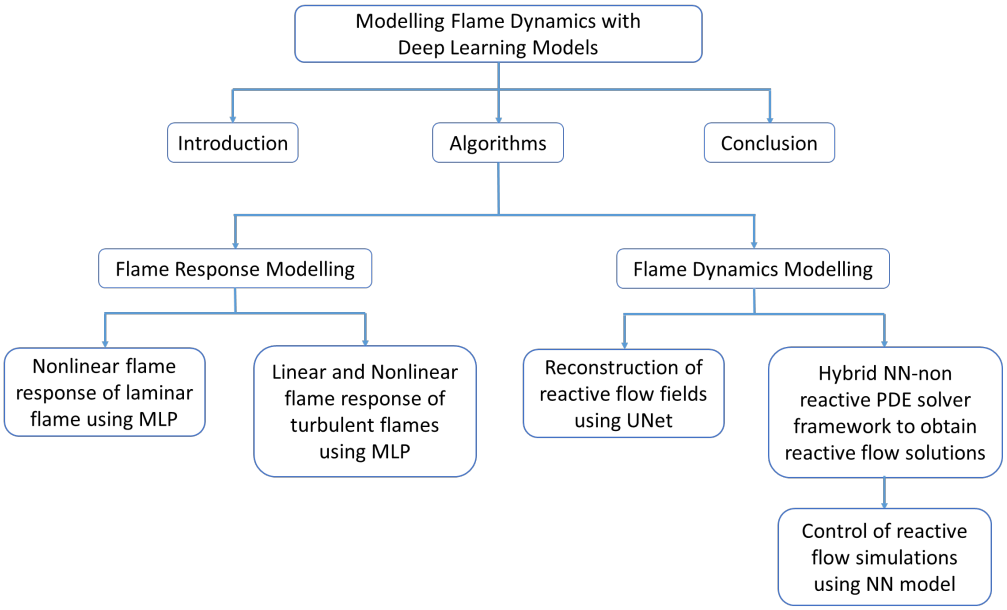


Figure 1.2: An overview of the dissertation structure.

Nonlinear flame response modelling is important to predict thermo-acoustic instabilities in lean, premixed combustion systems. A well-established strategy to model a nonlinear flame response is by means of the flame describing function. It characterizes the fluctuations of heat release rate, produced as a response to upstream velocity perturbations, which are characterized by a well-defined frequency and amplitude. As traditional approaches rely on multiple CFD runs in order to obtain the nonlinear flame response, they are generally unaffordable. It is, therefore, crucial to develop methodologies where nonlinear flame response can be obtained by a single CFD simulation. Neural network models are capable of capturing nonlinear relationships between the data. Therefore, the neural network is used as a post-processing step to the single, unsteady, broadband excited CFD simulation to model the nonlinear flame response. We demonstrate the applicability of the proposed framework on the laminar multi-slit burner investigated by Kornilov et al. [32] and Jaensch et al. [33]. The trained neural network model, which captures the flame response, is integrated with the acoustic network model to predict thermo-acoustic instabilities. This motivates us to extend the proposed neural network-based approach to challenging turbulent flames. We successfully train a multi-layer perceptron to obtain a linear and nonlinear flame response of two turbulent, swirl-stabilized combustors, namely the fully premixed EM2C [5, 34] and ETH burners [6]. These works showcase the potential of a multi-layer perceptron to predict one-dimensional heat release rate fluctuations in response to the perturbations in inlet flow velocity. Further, a convolutional neural network is used to reconstruct the two-dimensional heat release rate flow fields of the reactive flows.

After successfully applying the neural network model trained on the data coming from CFD solver, we explore integrating the neural network model with a differentiable

CFD solver to model the solutions of reactive flow simulations. Modelling reactive flow simulations require to resolve Navier-Stokes equations and transport of different species following complex chemical mechanisms. Often times the underlying chemistry is not well known or expensive to compute. Neural network models can be used to model the effects of the chemical kinetic mechanisms. Convolutional neural network models have been effective in modelling spatio-temporal relations, especially in the field of fluid dynamics [35, 36]. However, these purely data-driven approaches can fail to produce physically accurate solutions. Therefore it is necessary to use a methodology that guarantees the conservation of underlying physical quantities. We propose a hybrid framework that combines a neural network model with a non-reactive partial differential equations solver to account for the effects of unknown chemistry present in the system. Experiments are made on planar and Bunsen-type flames at various operating conditions. The hybrid neural network - PDE approach correctly models the flame evolution from given initial state due to various inlet, boundary conditions, of the cases under study for significantly long time windows. We show that the resulting hybrid solver provides a flexible building block for adjacent tasks. Specifically, we demonstrate this for controlling the evolution of flame shapes via continuous control. The hybrid solvers can efficiently predict the states of the reactive flows and readily enable the control of the flow inlet velocities to arrive at a desired flame shape. Therefore, we employ a pre-trained hybrid solver to train a second controller network that learns to steer the flame simulation such that the observed states are matched. Our experimental evaluation demonstrates the diverse possibilities of the hybrid NN-PDE solver, such as, performing accurate long term predictions, generalization to unseen operating conditions and enabling flame shape control.

All the proposed methods have the potential to be adapted to more complex, multi-physics problems for forecasting, parameter identification and control.

1.3 Publication List

This dissertation explains and concludes work from the following manuscripts:

1. **Tathawadekar, N.**, Doan, N. A. K., Silva, C. F., & Thuerey, N. (2021). Modeling of the nonlinear flame response of a Bunsen-type flame via multi-layer perceptron. *Proceedings of the Combustion Institute*, 38(4), 6261-6269.

2. **Tathawadekar, N.**, Silva, C., Sitte, P., & Doan, N. A. K. (2023, February). Physical quantities reconstruction in reacting flows with deep learning. *INTER-NOISE and NOISE-CON Congress and Conference Proceedings.* 265(6), 1645-1656. Institute of Noise Control Engineering.

3. **Tathawadekar, N.**, Doan, N. A. K., Silva, C. F., & Thuerey, N. (2023). Incomplete to complete multiphysics forecasting - a hybrid approach for learning unknown phenomena. *Data-Centric Engineering*, 4, E27. doi:10.1017/dce.2023.20

4. **Tathawadekar, N.**, Ösün, A., Eder, A. J., Silva, C. F., & Thuerey, N. (2024) Linear and nonlinear flame response prediction of turbulent flames using neural network models. *International Journal of Spray and Combustion Dynamics*. doi:10.1177/17568277241262641

2 Fundamentals

2.1 Fluid dynamics

Numerical techniques for reactive flows are essential in the field of chemical weather forecasting [37], combustion analysis in gas turbines and rocket engines [17]. Reactive flow simulations require the study of fluid dynamics as well as the chemistry of combustion. The following section describes the conservation equations needed for reactive flows, specifically the Navier-Stokes equations and the treatment of multi-species chemical mechanisms. Later we focus on the fundamentals of flame modelling. We describe challenges faced by existing numerical methods in capturing flame response accurately and establish the scope of this dissertation.

2.1.1 Navier-Stokes Equations

Reactive flows involve multiple species reacting through chemical reactions. The Navier-Stokes equations apply for such multi-species, multi-physics phenomena but they require some additional terms. The species are characterized through their mass fractions Y_k for $k = 1$ to N, where N is the number of species in the reacting mixture. The equations can be written as,

$$\begin{aligned}
\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_i}{\partial x_i} &= 0 \\
\frac{\partial \rho u_j}{\partial t} + \frac{\partial \rho u_i u_j}{\partial x_i} &= -\frac{\partial p}{\partial x_j} + \frac{\partial \tau_{ij}}{\partial x_i} \\
\frac{\partial \rho Y_k}{\partial t} + \frac{\partial \rho u_i Y_k}{\partial x_i} &= \dot{\omega}_k + \frac{\partial}{\partial x_i}\left(\rho D_k \frac{\partial Y_k}{\partial x_i}\right) \\
\rho C_p \left(\frac{\partial T}{\partial t} + \frac{\partial u_i T}{\partial x_i}\right) &= \dot{\omega}'_T + \frac{\partial}{\partial x_i}\left(\lambda \frac{\partial T}{\partial x_i}\right)
\end{aligned} \tag{2.1}$$

where density ρ, the three dimensional velocity field u_i, pressure p, temperature T, the mass fractions Y_k are primary variables. τ, D_k, λ are the strain rate tensor, the diffusion

coefficient of species k and the mixture thermal conductivity. In addition, C_p denotes the mixture specific heat capacity. $\dot{\omega}_k$ and $\dot{\omega}'_T$ are the species reaction rate and heat release rate, respectively.

In this section, we discuss the equations and relevant simplifications used to produce the simulation data of reactive flows. These equations are derived from the work of Poinsot and Veynante [17]. Readers are encouraged to refer to their work for more detailed discussions. The first two equations are the total mass conservation equation and conservation of momentum equations, respectively. They are unchanged compared to non-reacting flows. The mass conservation equation for species k is written as,

$$\frac{\partial \rho Y_k}{\partial t} + \frac{\partial}{\partial x_i}(\rho(u_i + V_{k,i})Y_k) = \dot{\omega}_k \tag{2.2}$$

where $V_{k,i}$ is the i-component of the diffusion velocity V_k of species k and $\dot{\omega}_k$ is the reaction rate of species k [17]. By definition:

$$\sum_{k=1}^{N} Y_k V_{k,i} = 0 \qquad \sum_{k=1}^{N} \dot{\omega}_k = 0 \tag{2.3}$$

The diffusion velocities V_k of the N species are obtained by solving the system of equations [38] given by,

$$\nabla X_p = \sum_{k=1}^{N} \frac{X_p X_k}{\mathcal{D}_{pk}}(V_k - V_p) + (Y_p - X_p)\frac{\nabla P}{P}\frac{\rho}{p}\sum_{k=1}^{N} Y_p Y_k (f_p - f_k) \quad p = 1, N \tag{2.4}$$

where $\mathcal{D}_{pk} = \mathcal{D}_{kp}$ is the binary mass diffusion coefficient of species p into species k and X_k is the mole fraction of species k; $X_k = Y_k W / W_k$. For multi-species system, equation 2.4 is replaced by the Hirschfelder and Curtiss [39] approximation which is the best first-order approximation to the exact solution of system 2.4.

$$V_k X_k = -D_k \nabla X_k \quad \text{with} \quad D_k = \frac{1 - Y_k}{\sum_{j \neq k} X_j / \mathcal{D}_{jk}} \tag{2.5}$$

The coefficient $\mathcal{D}_{jk}$ is a diffusion coefficient of species k into the rest of the mixture [17]. If the mixture contains only two species, then equation 2.5 reduces to Fick's law [40], otherwise it leads to the third equation in 2.1. The fourth equation in 2.1 is the energy conservation equation. For a reacting flow, the energy is represented as a sum of sensible and chemical energy. $\dot{\omega}'_T$ is the heat release due to combustion $\dot{\omega}'_T = -\sum_{k=1}^{N} \Delta h^o_{f,k} \dot{\omega}_k$. The formation enthalpy $h^o_{f,k}$ is the enthalpy needed to form 1 kg of species k at the reference temperature $T_0 = 298.15K$ [17]. Now, we look into the details of chemical reactions and how associated quantities are computed.

Consider a chemical system of N species reacting through M reactions,

$$\sum_{k=1}^{N} \upsilon'_{kj} \mathcal{M}_k \rightleftharpoons \sum_{k=1}^{N} \upsilon''_{kj} \mathcal{M}_k \quad \text{for} \quad j = 1, M \tag{2.6}$$

where $\mathcal{M}_k$ is a symbol for species k, υ'_{kj} and υ''_{kj} are the molar stoichiometric coefficients of species k in reaction j [17]. Mass conservation enforces:

$$\sum_{k=1}^{N} \upsilon'_{kj} W_k = \sum_{k=1}^{N} \upsilon''_{kj} W_k \quad \text{or} \quad \sum_{k=1}^{N} \upsilon_{kj} W_k = 0 \quad \text{for} \quad j = 1, M \tag{2.7}$$

where $\upsilon_{kj} = \upsilon''_{kj} - \upsilon'_{kj}$. For simplicity, only mass reaction rates are used. For species k, this rate $\dot{\omega}_k$ is the sum of rates $\dot{\omega}_{kj}$ produced by all M reactions,

$$\dot{\omega}_k = \sum_{j=1}^{M} \dot{\omega}_{kj} = W_k \sum_{j=1}^{M} \upsilon_{kj} Q_j \tag{2.8}$$

where Q_j is the rate of progress of reaction j [17]. The progress rate Q_j of reaction j is written as,

$$Q_j = K_{fj} \Pi_{k=1}^{N} [X_k]^{\upsilon'_{kj}} - K_{rj} \Pi_{k=1}^{N} [X_k]^{\upsilon''_{kj}} \tag{2.9}$$

where K_{fj} and K_{rj} are the forward and reverse rates of reaction j [17]. The kinetic rates are expressed using molar concentrations $[X_k] = \rho Y_k / W_k = \rho_k / W_k$. The rate constants K_{fj} and K_{rj} are usually modelled using the empirical Arrhenius law,

$$K_{fj} = A_{fj} T^{\beta_j} \exp\left(-\frac{E_j}{RT}\right) = A_{fj} T^{\beta_j} \exp\left(-\frac{T_{aj}}{T}\right) \tag{2.10}$$

where A_{fj} is the pre-exponential constant, β_j is the temperature exponent, T_{aj} is the activation temperature and $E_j = RT_{aj}$ is the activation energy [17].

In a premixed combustor, fuel and oxidizer are mixed before they enter the combustion chamber. The computation of premixed flames with complex chemistry and transport is complicated. The simplified premixed flames with one-step chemistry are presented here. For a one-step reaction, chemistry proceeds only through one irreversible reaction i.e. $M = 1$ in equation 2.6.

$$\sum_{k=1}^{N} \upsilon'_{kj} \mathcal{M}_k \rightarrow \sum_{k=1}^{N} \upsilon''_{kj} \mathcal{M}_k. \tag{2.11}$$

The reverse reaction rate is zero; $K_{r1} = 0$. The forward reaction rate is $K_{f1} = A_1 T^{\beta_1} e^{-T_{a1}/T}$. The reaction rates $\dot{\omega}_k$ for each species are linked to the progress rate Q_1 at which the single reaction proceeds by equation 2.8,

$$\frac{\dot{\omega}_k}{W_k \upsilon_{k1}} = Q_1 \quad \text{or} \quad \dot{\omega}_k = W_k \upsilon_{k1} Q_1. \tag{2.12}$$

The heat release term $\dot{\omega}'_T$ in the temperature equation becomes,

$$\dot{\omega}'_T = -\sum_{k=1}^{N} \Delta h^o_{f,k} \dot{\omega}_k = -Q_1 \sum_{k=1}^{N} \Delta h^o_{f,k} W_k \upsilon_{k1} = |\upsilon_{F1}| Q^m Q_1 \tag{2.13}$$

where Q^m is the molar heat of the reaction. Q^m measures the heat released by the complete combustion of 1 mole of fuel,

$$Q^m = -\sum_{k=1}^{N} \Delta h^o_{f,k} W_k \frac{\upsilon_{k1}}{|\upsilon_{F1}|} = -\sum_{k=1}^{N} \Delta h^{o,m}_{f,k} \frac{\upsilon_{k1}}{|\upsilon_{F1}|} = \sum_{k=1}^{N} \Delta h^{o,m}_{f,k} \frac{\upsilon_{k1}}{\upsilon_{F1}} \tag{2.14}$$

because $\upsilon_{F1} = \upsilon''_{F1} - \upsilon'_{F1}$ is negative. Poinsot et al. [17] define the quantity Q,

$$Q = \frac{Q^m}{W_F} = \sum_{k=1}^{N} \Delta h^o_{f,k} \frac{W_k \upsilon_{k1}}{W_F \upsilon_{F1}} \tag{2.15}$$

to have positive Q values. The relationship between the heat release rate and the fuel reaction rate becomes,

$$\dot{\omega}'_T = -Q\dot{\omega}_F \tag{2.16}$$

In chapter 7, following assumptions are used to create reactive flow simulation data. The fuel reaction rate $\dot{\omega}_F$ of the reaction 2.11 is assumed to be limited by the fuel mass fraction (Y_F) and not by the oxidizer mass fraction (Y_O) [17]. This assumption corresponds to the study of a very lean flame in which oxidizer mass fraction remains approximately constant. Only one species Y_F has then to be considered to evaluate the fuel reaction rate. Therefore, the simplified fuel reaction rate is,

$$\begin{gathered} \upsilon'_{F1} = 1; \quad \upsilon''_{F1} = 0; \quad \upsilon_{F1} = -1; \quad \upsilon'_{O1} = \upsilon''_{O1} = \upsilon_{O1} = 0 \\ \dot{\omega}_F = A_1 T^{\beta_1} \upsilon_{F1} \rho Y_F e^{-T_{a1}/T} \end{gathered} \tag{2.17}$$

Even though these assumptions may seem strong, they preserve many essential features of flames such as intense nonlinear heat release, variable density and temperature [17].

2.1.2 Fundamentals of Flame Modelling

In order to meet low NOx emission targets for modern gas turbines used for power generation, lean premixed combustion technology is applied. The main drawback of this system is thermo-acoustic oscillations arising from the interactions between an oscillatory combustion process and, natural acoustic modes of the combustor [13]. The flame moves in response to the fluctuations in the velocity (u') impinging on the flames. (1) These flow rate fluctuations induce fluctuations in the heat release rate ($\dot{Q}'$). (2) The resulting heat release rate fluctuations excite acoustic pressure oscillations (p'). (3) The acoustic oscillations generate the velocity fluctuations [13] described in point (1). This feedback cycle may lead to large-amplitude pressure and velocity oscillations which may cause severe damage to the combustion system. Therefore, it is desirable to predict the thermo-acoustic stability behaviour over the whole operational envelope in the design phase of a combustion system. Because the initial amplitudes of most instabilities are generally quite small, their characteristics are described by the linear-wave equation [13]. To predict the amplitude of the thermo-acoustic instabilities, a low-order model of acoustic network of the system can be combined with an appropriate model of flame response

[41, 42]. The flame response estimates the response of flame subjected to the velocity perturbations [42].

In the limit of small perturbation, a linear model can capture the flame response. A well-established strategy to model such linear flame response is by means of the flame transfer function (FTF). This transfer function relates the fluctuations of heat release rate, $\dot{Q}'$, produced as a response to upstream velocity perturbations, u', by

$$\text{FTF} : \mathcal{F}(\omega) = \frac{\dot{Q}'(\omega)/\bar{\dot{Q}}}{u'(\omega)/\bar{u}} \tag{2.18}$$

where $\bar{(\cdot)}$ and $(\cdot)'$ denote temporal averaging and fluctuations, respectively. $\omega = 2\pi f$ is the angular frequency. In linear stability analysis, the flame transfer function can be cast in the following form:

$$\mathcal{F}(\omega) = G(\omega)e^{i\omega t} \tag{2.19}$$

where gain and phase are modelled separately.

$$\begin{aligned} p' &\propto e^{-i\omega t} \\ \omega &= \omega_r + i\omega_i \\ p' &\propto e^{-i\omega_r t}e^{\omega_i t} \end{aligned} \tag{2.20}$$

$$\omega_i \begin{cases} > 0 & \text{growth} \\ = 0 & \text{neutral} \\ < 0 & \text{damped} \end{cases}$$

The pressure fluctuations are proportional to the $e^{i\omega t}$. ω is composed of angular frequency (ω_r) and growth rate (ω_i). If the imaginary part of the frequency ω is positive, amplification of the small perturbations is realized and combustion instability is expected. The instability frequency is obtained from the real part of ω. The limit cycle is obtained when the growth rate equals the damping rate. Linear analysis however cannot predict amplitudes of self-excited oscillations correctly [43, 44].

The flame describing function is introduced to model nonlinear flame response by Noiray et al. [41]. The main idea is to extend the definition of a flame transfer function to include flame response to both input frequency and amplitude. The FDF:

$$\text{FDF} : \mathcal{F}(\omega_r, |u'|) = \frac{\dot{Q}'(\omega_r, |u'|)/\bar{\dot{Q}}}{u'(\omega_r, |u'|)/\bar{u}} = G(\omega_r, |u'|)e^{i\psi(\omega_r, |u'|)} \tag{2.21}$$

extends the transfer function concept to the nonlinear case. It is a family of transfer functions which describe the system at various levels of amplitude. FDF is a weakly nonlinear function, i.e., it assumes that only the fundamental frequency determines the dynamics of the system, while higher harmonics generated in the nonlinear process are of sufficiently low amplitude to have a negligible effect on the system stability. Instability in the combustion system for particular acoustic perturbation can be identified by solving the nonlinear dispersion equation. Noiray et al. [41] showed that the FDF can be used to

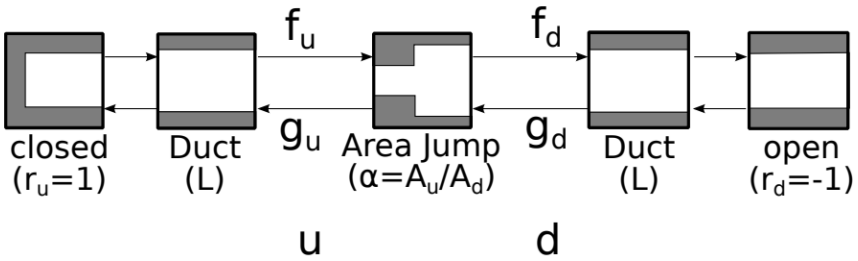


Figure 2.1: An example of a simple low order acoustic network. Adapted from [2]

model growth rates and modal frequencies as a function of the perturbation amplitude level. It can be used to predict the exact amplitudes of oscillation and frequencies at the limit cycles. There are different ways to obtain this flame response. Experimentally, the flame response is obtained using acoustic forcing provided by speakers mounted in the upstream or downstream duct [45, 46, 47, 48]. For numerical analysis, CFD simulations are performed at various frequencies and amplitudes to obtain the flame response [49, 50].

The flame transfer function and the flame describing function capture the flame response to incoming velocity perturbations. Thus it models the interaction in step (1) of the feedback loop described at the start of this subsection. The acoustic waves in step (2) can be modelled with an acoustic network model [2]. It models the propagation of acoustic perturbations by a set of linear, time-invariant differential equations. The combustor configuration can be divided into several elements and each element is separately modelled. In the end, all models are interconnected to obtain the acoustic network model of the combustor. Figure 2.1 shows an example of a low order acoustic network of a simple area jump with two duct sections terminated by a closed respectively open end. Thermoacoustic oscillations are modelled by combining the model of flame dynamics in a loop with the acoustic network model.

2.2 Deep Learning Methods

Machine learning is a statistical tool with increased emphasis on the use of computers to estimate complicated functions. Compared to the traditional statistical methods that fit the training data, machine learning methods differ in finding patterns that generalize to new data. Various algorithm components, such as an optimization algorithm, a loss (cost) function, a model, and a dataset, are combined to build a machine learning algorithm. A dataset consists of a collection of examples that a machine learning algorithm should process. Most machine learning algorithms can be divided into the categories of supervised and unsupervised learning, depending on the nature of the dataset available. Unsupervised learning algorithms use a dataset containing only examples and learn useful properties of the structure of the dataset. Supervised learning algorithms use a dataset containing examples and their associated label or target. We refer to such data as input-output pairs. In this work, we only consider supervised learning algorithms. Given a training set of examples of inputs X and outputs y, the learning algorithm es-

timates a probability distribution $p(y|X)$. Simple machine learning algorithms face the challenge of generalizing to new examples when working with high-dimensional data. The mechanisms used to achieve generalization in traditional machine learning are insufficient to learn complicated functions in high-dimensional spaces. Such spaces also often impose high computational costs.

Deep learning was designed to overcome these and other obstacles. Deep learning is a specific kind of machine learning. Modern deep learning provides a powerful framework for supervised learning. Deep learning models are built using layers of artificial neurons that are loosely inspired by the neurons in a human brain. At the core of deep learning are neural networks, which consist of layers of nodes (neurons) that transform input data into output predictions. Each neuron receives input, applies a weight to it, and passes the result through a non-linear activation function. By adding more layers and more neurons within a layer, a deep network can represent functions of increasing complexity. Most tasks consist of mapping an input vector to an output vector. In this chapter, we begin by describing the multi-layer perceptron that is used to represent these functions. Next, we introduce the convolutional neural networks, which efficiently processes the two or multi-dimensional image data. Over the past decade, deep learning has had incredible success in various fields [21, 18, 51]. Deep learning is revolutionizing many fields, such as image processing, natural language modelling, autonomous driving, climate modelling, and medical image processing, to name a few. Therefore, this dissertation explores the use of deep learning techniques to model flame dynamics.

2.2.1 Multi-Layer Perceptron

Multi-layer perceptron or feed-forward neural networks, are the quintessential deep learning models. The goal of a multi-layer perceptron is to approximate some function f^*. A multi-layer perceptron defines a mapping $y = f(X;\theta)$ and learns the value of the parameters θ that result in the best function approximation. These models are called feedforward networks because information flows from input X to the output y, through the network. There are no feedback connections in which the output of the network is fed back as input. These networks are called *multi-layer* because they are constructed by composing together multiple layers. Each layer represents a function and multiple of them are chained together to form the best function approximation f. The final layer is called the output layer.

Figure 2.2 shows a simple MLP model with one hidden layer. The features of the input example X_i are stacked together to form the input layer. The final layer formed by a single node is called an output layer. In supervised learning, the training dataset contains values of input examples $X = \{X_1, X_2, ..., X_n\}$ and corresponding outputs $y = \{y_1, y_2, ..., y_n\}$. In the training set, true values of the intermediate layers are not observed. Therefore, these middle layers are called as hidden layers. The hidden layers and output layer have parameters associated with them, called as weights W and biases b. For the example shown in figure 2.2, the outputs of the hidden layer and output layer

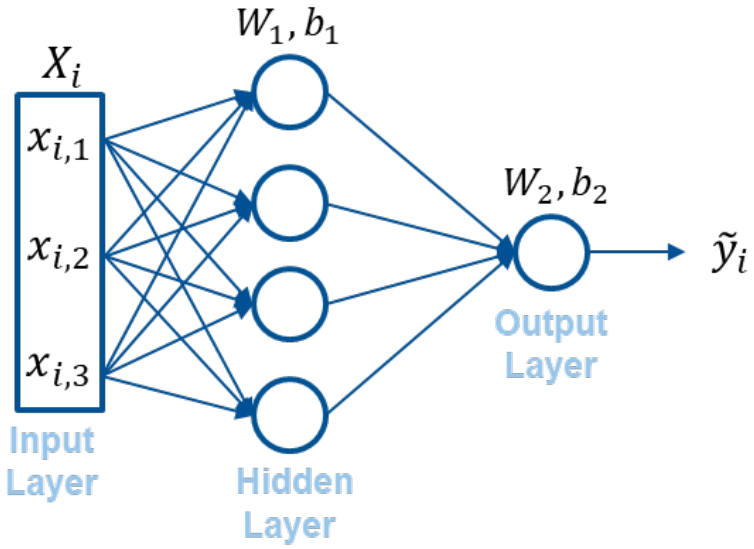


Figure 2.2: A generic representation of multi-layer perceptron

are computed as follows,

$$\begin{aligned} Z_i &= \sigma(W_1^T X_i + b_1) \\ \tilde{y_i} &= \sigma(W_2^T Z_i + b_2) \end{aligned} \tag{2.22}$$

where σ indicates the activation function. The activation function can be used to introduce nonlinearity to the affine transformation of $W_1^T X_i + b_1$. Some examples of nonlinear activation function include, sigmoid ($\sigma(z) = \frac{1}{1+e^{-z}}$) which restricts the output in the range of $[0, 1]$, tanh ($\sigma(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$) restricts it to $[-1, 1]$, ReLU ($\sigma(z) = max(0, z)$) maps it to positive values. Therefore, the key design considerations to build a neural network model are the number of hidden layers, the number of neurons in each hidden layer, the type of activation function to be used. These are called hyper-parameters of the model. The neural network models are trained using iterative, gradient-based optimization techniques. A suitable loss function is defined to compute the distance between the predicted output ($\tilde{y}$) and the actual output (y). Gradient descent is used to minimize the loss function. It calculates the derivative of the loss function with respect to the trainable weights via chain rule and updates the weights. Therefore, the error is propagated back to each neuron and its corresponding weights are adjusted accordingly. This process of updating weights is called back-propagation.

2.2.2 Convolutional Neural Networks

Convolutional neural networks are another type of neural network model that have been tremendously successful at various tasks, such as image classification [52], object detection [53] and speech recognition [54]. CNNs provide an ideal architecture for detecting and learning key features in images and time-series data. They specialize in processing

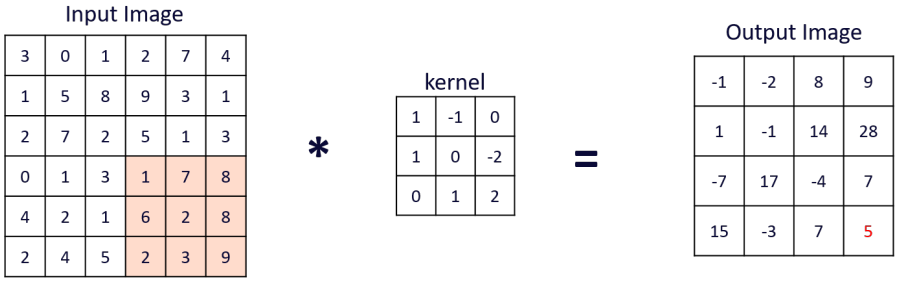


Figure 2.3: A generic example of a convolutional operation

data that has a grid-like topology, such as images and videos. MLP ignores the locality of reference in image data, both computationally and semantically. Thus, fully connected neurons are wasteful for the task of image classification or object detection which are dominated by spatially local input patterns. With MLP, the number of trainable parameters grow rapidly with the increase in data dimensions and the number of layers. The convolutional neural network mainly consists of convolutional layers and pooling layers. A convolutional layer performs a dot product between two matrices, where one matrix is the set of learnable parameters, also known as a kernel, and the other matrix is the restricted portion of the receptive field. The kernel is spatially smaller than an image. During the forward pass, the kernel slides across the height and width of the image, producing the image representation of that receptive field. Figure 2.3 shows an example of such a convolution operation. In CNN, the convolution operation is performed discretely since both the inputs and the kernels are spatially discrete. The one-dimensional discrete convolution $f * g$ of two functions where $f, g : \mathbb{D} \to \mathbb{C}$ can be written as,

$$(f * g)(x) = \sum_{j \in \mathbb{D}} f(j)\, g(x - j). \tag{2.23}$$

Unlike MLP, a CNN has common weights and bias values that are the same for all hidden neurons in a given layer. Convolutional neural networks have sparse interaction, achieved by small kernels. This leads to a small number of trainable parameters, making CNNs efficient for high-dimensional data. The trainable parameters of the kernel are shared; i.e. same kernel is applied to the whole input to obtain the output. This makes CNNs equivariant to translation. If we change the input in some way, the output will also get changed in the same way. The pooling layer replaces the output of the network at certain locations by deriving summary statistics of the nearby outputs. This helps in reducing the spatial size of the representation, which decreases the required amount of computation and weights. The pooling operation is processed on every slice of the representation individually. This reduces the size of the representation and also makes it possible to build connections for points far away from each other. There are many types of pooling functions that can be used. However, the most popular process is max pooling, which reports the maximum output from the neighbourhood. It helps to make the detected features more robust. Since convolution is a linear operation, activation functions are used to introduce nonlinearity as discussed in section 2.2.1.

Convolutional neural networks have become foundational in computer vision and image processing, achieving remarkable success in various real-world applications, such as medical imaging, audio processing, object detection, and synthetic data generation.

3 Literature Review

This literature review summarizes various approaches used to model the flame response, amplitudes of thermo-acoustic instabilities and flame dynamics. We discuss the progress made so far in obtaining these models and the challenges faced by existing methods.

3.1 Thermo-acoustic System Identification

Thermo-acoustic instability arises from the positive interaction between acoustic waves and unsteady combustion and manifests as self-excited acoustic oscillations [13]. To predict the amplitude of these acoustic oscillations at the design phase, low-order methods have proven to be useful [41, 48, 42] if combined with appropriate models for the nonlinear acoustic flame response. A well-established strategy to model such a nonlinear flame response is by means of the flame describing function. Conventionally, to obtain this FDF, the inlet velocity is excited harmonically and both the frequency and amplitude of the harmonic are varied. Although useful, the FDF approach is restrictive as it assumes that the flame responds exclusively at the same frequency as the input signal u'. Accordingly, it neglects the possible apparition of harmonics in the response $\dot{q}'$. Such an assumption may lead to an inaccurate prediction of thermo-acoustic limit cycles [55, 56]. To extend this FDF framework, Haeringer et al. [57] proposed an extended FDF (xFDF), which includes additional transfer functions relating higher harmonics of the heat release rate to the forcing velocity. The extended flame describing function performs better than the conventional FDF in predicting limit cycle amplitudes, frequencies and time series shapes. However, the same number of CFD simulations are required for xFDF compared to the conventional approach. Alternatively, Orchini et al. [58] introduced the flame double input describing function (FDIDF), where the FDF is extended by forcing the flame with two amplitudes and two frequencies. The FDIDF is a more expensive object to calculate, but yields an improved stability prediction and the nonlinear flame response is approximated more precisely than the conventional approach.

A more reliable yet more complex alternative to the FDF-acoustic network approach is the hybrid CFD/low-order acoustic approach. It consists of coupling 'on the fly'

computational fluid dynamics to an acoustic solver. While the former simulates the flame acoustic response $\dot{q}'$ to a given input velocity u', the latter provides the fluctuations u' as the result of the acoustic source $\dot{q}'$ and the propagation of acoustic waves and corresponding reflection at the domain boundaries. This method was demonstrated to predict limit cycle amplitudes accurately [4]. Unlike FDF, this approach captures the nonlinear interaction between fundamental frequency and associated harmonics to predict the self-excited oscillations.

Obtaining the FDF (or xFDF) by means of CFD is usually computationally expensive. This is because, under traditional methods, many independent simulations are necessary, each one associated with a given amplitude and frequency. The computational costs for the double input FDF are even higher if the same strategy is followed. The computational costs associated with the hybrid CFD/low-order model are directly linked to the costs of CFD, which may be considerable for highly resolved flames. As a consequence, nonlinear thermoacoustic studies, where the nonlinear flame response plays a central role, are generally unaffordable if all the calculation chain rely on numerical simulations. It is therefore crucial to develop methodologies, where the FDF - or eventually more complex flame models - can be obtained by a *single* CFD simulation, where a carefully designed input signal u' is applied during a short physical time window.

Polifke et al. [59] proposed a combined application of computational fluid dynamics and system identification (CFD/SI) to characterize the flame response to incoming flow perturbations. The CFD simulation is performed by exciting a flame with a low-pass filtered, broadband signal of upstream acoustic velocity perturbations. The resulting time series of fluctuating heat release rate $\dot{Q}'$ and inlet velocity u' are used to identify the linear flame response, i.e. the impulse response, and corresponding frequency response using correlation analysis. Even though this approach models an FTF from a single simulation, it is limited to the *linear* flame response. Modelling the nonlinear flame response [60, 41] remains unattainable for such kind of correlation approaches.

3.2 Deep Learning for Flame Response Modelling

The foundation of the SI methods is the linear time-invariant systems. Therefore, they are incapable of modelling the nonlinear flame response and in turn the amplitudes of self-excited oscillations. The universal approximation theorem states that an MLP has the ability to learn any nonlinear function in its subspace [61]. This forms the basis to use MLPs for the modelling of the nonlinear flame response. Selimefendigil et al. [3] uses a feed-forward network to model limit cycle oscillations of linearly unstable thermo-acoustic systems. A feed-forward neural network is used to model the nonlinear behaviour of the heat source in a Rijke tube, using the broadband excited input-output dataset generated with unsteady CFD simulation. The tanh activation function of the neural network model is approximated with polynomials, resulting in a Volterra series representation of input-output relations. The kernels are transformed into the frequency domain to obtain the higher-order transfer functions. Overall, limit cycles are captured accurately when compared with the Galerkin time domain simulation for a Rijke tube.

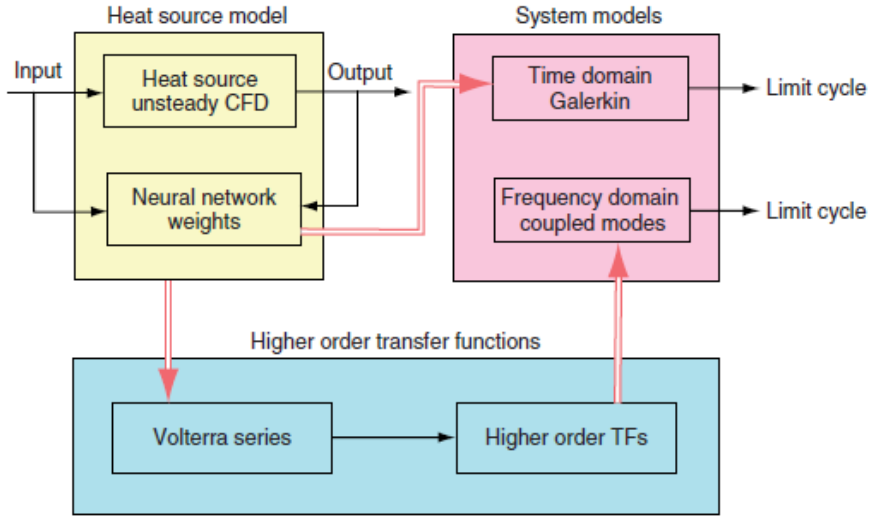


Figure 3.1: Modelling of nonlinear heat source using feed-forward neural network model. Adapted from Selimefendigil et al. [3]

Selimefendigil et al. [62] uses the CFD/SI approach to model the unsteady heat transfer of a cylinder in pulsating cross flow. Förner and Polifke [63] modelled the nonlinear acoustic behaviour of Helmholtz resonators using a data-based, reduced-order model. The data coming from CFD simulation is used to train a local-linear neuro-fuzzy network. All these studies have extended the CFD/SI approach to nonlinear regimes using neural networks in the context of nonreacting flows. A similar approach was also proposed by Jaensch et al. [33] for reacting flows, where they attempted to model the FDF of the Kornilov's flame [32] using neural network models. The results of that study were not satisfactory, perhaps because of the lack of regularization techniques, such as dropout or L-norms, or because of the use of shuffling in the data [64]. Nonetheless, it suggested different ways to improve the results, such as using a more sophisticated and robust model.

The study presented in Chapter 4 [65] uses the data collected by broadband forcing and makes use of a machine learning-based approach to model the nonlinear acoustic response of flames. This demonstrated the use of multi-layer perceptron (MLP), the same type of neural network (NN) model used in Förner and Polifke [33], to characterize the nonlinear flame response of a laminar flame. When combined with an acoustic solver, the trained neural network model was able to predict limit-cycle amplitudes accurately. Following work by Yadav et al.[66] applied the long short-term memory (LSTM) model, a type of recurrent neural network, to predict the nonlinear flame response of a laminar flame. They show that LSTM may be advantageous, as they require shorter time series if compared with the MPL counterparts. However, that work lacks validation of LSTM models to predict flame responses for amplitudes higher than 0.5 and to predict limit cycle oscillations when coupled with a given acoustic solver. Rywik et al. [67] pro-

posed an alternative approach to model the nonlinear flame response using multi layer perceptron-convolutional autoencoder (MLP-CAE) model. It models the flow field data to capture the nonlinear spacial dynamics. This approach can be useful to study the flame response of acoustically noncompact flames. As a future direction, it would be interesting to extend this work for more challenging turbulent flames.

3.3 Deep Learning for Fluid Dynamics

Deep learning in the field of fluid dynamics is rapidly developing, driven by large volumes of data made available by experiments and numerical simulations. Deep learning can be efficient in extracting patterns within the data and translating it into knowledge about the underlying fluid dynamics. Deep learning methods have been widely used to model the solutions of partial differential equations [68, 69, 70, 71] and in particular, the Navier-Stokes equations [24, 72].

3.3.1 Purely Data-Driven Models

A simple approach in modelling any physical system consists of training a deep learning model using data coming from either experiments or numerical simulation [24, 72]. These models solely use deep learning techniques with appropriate data to make predictions and hence are called as *purely data-driven models*. Thuerey et al. [24] applied a convolutional neural network to learn Reynolds-Averaged Navier-Stokes (RANS) solutions of airfoil flows. The proposed approach is very generic and applicable to a wide range of PDE boundary value problems on Cartesian grids. Stachenfeld et al. [73] demonstrate that a generic CNN-based model may predict turbulent dynamics on coarse grids more accurately than classical numerical solvers. The proposed approach is effectively applied to a wide range of physical domains, which can be represented as grids. These classical neural networks map between finite-dimensional spaces and can only learn solutions tied to a specific discretization, which can be excessively limiting. Therefore, approaches based on learning operators are receiving increased attention. Lu et al. [74] proposed a novel architecture based on fully connected neural networks called DeepONets to learn diverse linear or nonlinear explicit and implicit operators. Neural operators [75, 76, 77], specifically, the Fourier neural operator (FNO) of Li et al.[78] introduce an interesting line of work by learning mesh-free, infinite-dimensional operators with neural networks, but do not necessarily offer advantages for longer term predictions. These models can be very fast and may not suffer from the time-step stability issues associated with traditional numerical solvers. Nevertheless, as these purely data-driven approaches lack the physical understanding of the system being modelled, they generally fail in generalizing to other operating conditions [35, 36]. To leverage the potential of deep learning in physical simulations, it is, therefore, necessary to incorporate some physical information within the deep learning framework.

3.3.2 Physics-Informed Deep Learning Models

Deep learning models can enforce physical constraints partially through the loss function [71, 25, 66] or changes in neural network architecture [79]. Kim et al. [35] define a novel loss function to ensure the conservation of mass i.e. to ensure divergence-free motion for incompressible flows. A well-known framework called Physics-Informed Neural Network (PINN) [25] uses neural networks as methods for solving PDEs. It minimizes the residual of the underlying governing laws by taking advantage of automatic differentiation to compute exact, mesh-free derivatives. However, these approaches struggle to enforce physical constraints such as boundary conditions or predict the strong unsteadiness and chaotic nature of flows [80, 81]. PINN only learns the solution function of a single PDE instance and needs re-optimization for other instances or PDEs. To alleviate this issue, Wang et al. [82] extended the DeepONet framework by imposing the underlying physical laws via soft penalty constraints during training. Although physics-informed DeepONet imposes PDE losses on operator learning, they are not discretization invariant. To tackle this issue, the physics-informed neural operator framework was proposed by Li et al. [83] that uses available data and physics constraints to learn the solution operator of parametric PDEs. However, all these approaches require explicit knowledge of the underlying PDEs to accurately train the network. For the physical systems where only partial knowledge of their physical mechanism is known, these approaches may fail to converge to the solutions of the complete PDE solver. Minimizing the residual of only the known but incomplete PDEs will not lead to an accurate prediction of the solutions of a complete PDE system.

3.3.3 Neural Networks with Differentiable PDE Solvers

In recent years, the development of differentiable PDE solvers has led to an interesting line of research. Thuerey et al. [84] have developed an open-source physics simulation toolkit called *PhiFlow* for optimization and machine learning applications. *SU2* [85] is an open-source collection of software tools for analysing PDEs and PDE-constrained optimization problems on unstructured meshes with state-of-the-art numerical methods. *SciML* [86] is a collection of tools for solving equations and modelling systems. These tools provide differentiable functions for physical simulations, which enable close integration with deep learning frameworks by leveraging their automatic differentiation functionality. Hybrid approaches that combine machine learning techniques with numerical PDE solvers [87, 88] have attracted a significant amount of interest due to their capabilities for generalization [89]. In this context, neural networks are typically used to model or replace a part of the conventional PDE solver to improve aspects of the solving process. For example, Tompson et al. [90], Özbay et al. [91] and Ajuria Illarramendi et al. [92] proposed a convolutional neural network-based approach to solve the Poisson equation in CFD simulation. In recent years, a number of deep learning-based models have been introduced to accurately model turbulent flows [93, 73, 94]. Um et al. [27] and Belbute-Peres et al. [95] showed the advantages of training neural networks with differentiable physics to correct the numerical errors that arise in the discretization of

PDEs. Um et al. [27] integrates the convolutional neural network with the differential PDE solver from the *phiflow* framework [96, 84] to reduce numerical errors in PDEs due to spatial scales. They showcase improved generalization for out-of-distribution samples and obtain stable, long-term recurrent predictions. Belbute-Peres et al. [95] combined graph neural networks with *SU2* framework for differential CFD simulator to recover fine mesh solutions from coarse mesh inputs. These methods are shown to outperform a purely data-driven model. Sirignano et al. [97] and Kochkov et al. [28] additionally correct for closure error in turbulence modelling by integrating the neural network with a differentiable CFD simulator. These approaches demonstrate the capabilities of neural networks to correct errors in a fast, under-resolved simulation. The unresolved physics in turbulence modelling is attributed to spatial filtering. Given additional computing resources, one can arrive at accurate solutions by running the known PDEs at higher resolution. Note that in chapter 7, we investigate a more challenging case of unknown physics in a multi-physics system, where increasing the spatial or temporal resolution of the known, but incomplete PDEs will not lead to the accurate solution of the complete PDEs.

Yin et al. [98] and Takeishi and Kalousis [99] have introduced frameworks for augmenting incomplete physical dynamics with neural network models. These approaches demonstrate the applicability on ODE/PDE systems, which are weakly nonlinear and the unknown dynamics are linear combinations of the underlying flow fields. We expand on these works to explore the significantly more challenging scenario of reactive flows. These are characterised by multi-physics, multi-species systems with nonlinear advective terms, which model the transport of a flow state by the velocity of the flow, and strongly nonlinear dynamics, described by exponential source terms that exhibit nonlinear combinations of the flow fields. Compared to the work by Yin et al. [98], we do not use an additive approach to learn the solutions of the complete PDE system. Instead, we use the output of an incomplete PDE solver as an input to the neural network model to correct for the effects of unknown terms.

4 Modelling of Thermo-acoustic Oscillations

This work demonstrates the ability of neural networks to reliably learn the nonlinear flame response of a laminar premixed flame, while carrying out only one unsteady CFD simulation. The system is excited with a broadband, low-pass filtered velocity signal that exhibits a uniform distribution of amplitudes within a predetermined range. The obtained time series of flow velocity upstream of the flame and heat release rate fluctuations are used to train the nonlinear model using a multi-layer perceptron. Several models with varying hyperparameters are trained and the dropout strategy is used as regularizer to avoid overfitting. The best performing model is subsequently used to compute the flame describing function using mono-frequent excitations. In addition to accurately predicting the FDF, the trained neural network model also captures the presence of higher harmonics in the flame response. As a result, when coupled with an acoustic solver, the obtained neural network model is better suited than a classical FDF model to predict limit cycle oscillations characterized by more than one frequency. The latter is demonstrated in the final part of the present study. We show that the RMS value of the predicted acoustic oscillations together with the associated dominant frequencies are in excellent agreement with CFD reference data. Following work was presented in the *38th International Symposium on Combustion* and is published in *Proceedings of the Combustion Institute* by Tathawadekar et al. [65]. It is reproduced here under the terms of the Creative Commons Attribution 4.0 License.

4.1 Numerical Setup

The experimental and CFD setup studied in this work is shown in Fig. 4.1. It is the laminar multi-slit burner investigated by Kornilov et al. [32] where a premixed methane-air mixture with an equivalence ratio of 0.8 is used. The velocity perturbation is performed by using a loudspeaker operated by a single tone generator. Here, the numerical setup and results of Jaensch et al. [33] are used: a 2D CFD domain with symmetric boundary conditions to reduce the computational cost, the two-step chemical scheme as detailed in Selle et al. [100] and OpenFOAM, a low-Mach number solver, are used for the simu-

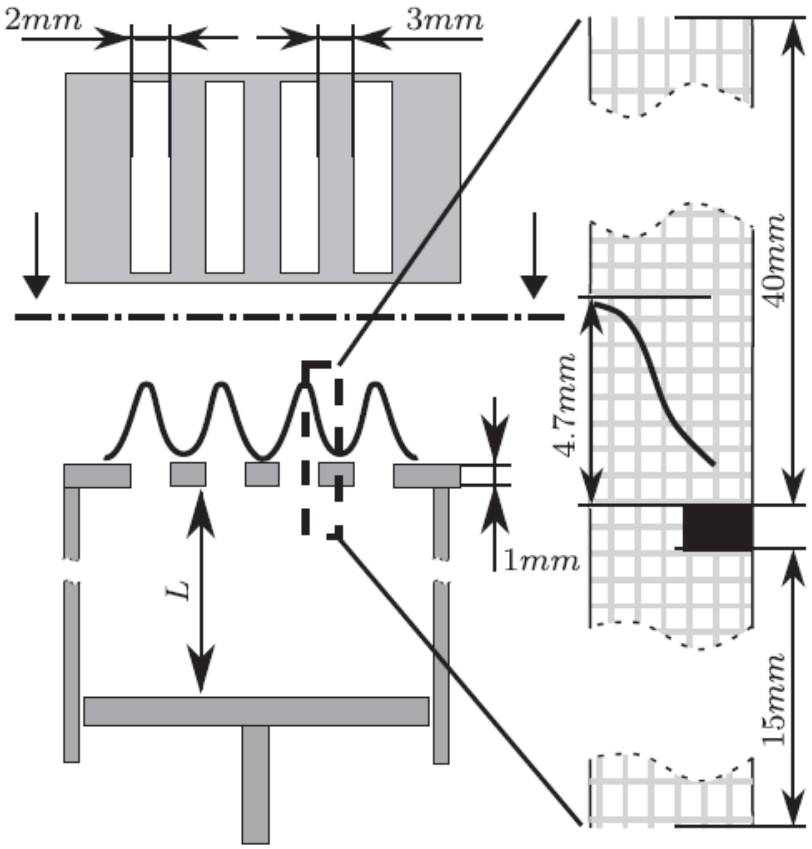


Figure 4.1: Left: Experimental configuration. Right: CFD domain. Figure adapted from [4].

lation. Additionally, no turbulent combustion model is used here as the flame is laminar and all species transport equations are fully resolved. This approach with a low-Mach formulation, coupled with an acoustic network model via the global heat release rate and the fluctuation of the axial velocity at a reference position upstream of the flame, has already been validated against one based on resolving the fully compressible Navier–Stokes equations, coupled to the low-order model via the characteristic wave amplitudes at the inlet boundary by Jaensch et al.[4]. Given that there is no turbulence, all flow quantities are also perfectly resolved. The grid was chosen following the grid independence study performed in [4]. At the inflow, a mean inlet velocity of 0.4 m/s and inlet temperature of 293 K are imposed. The plate on which the flame is stabilized is modelled as a no-slip wall with a fixed temperature of 373K, as measured in the experiment by Kornilov et al. [32]. A structured grid with 122,300 cells was used. The grid is uniform with a cell size of 0.025 mm in the region of the steady-state position of the flame, which ensures that the flame is fully resolved, and the area of contractions. Outside this region, the cells were stretched in the axial direction [4]. The CFD simulation is run with an adaptive time-stepping scheme with an average timestep $\Delta t = 10^{-6}$.

4.2 Data Generation and Processing

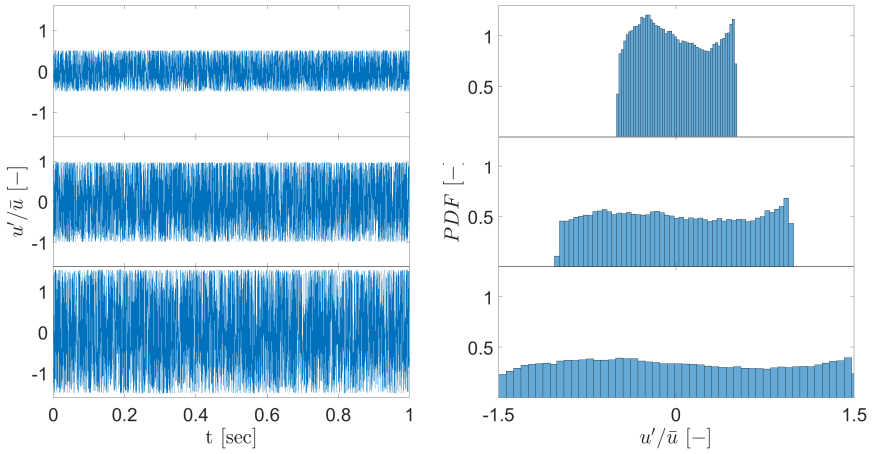


Figure 4.2: Input signals used to train and validate the NN models. Dataset A, B, C from top to bottom. Left: input time series $u'/\bar{u}$, Right: probability density function of the input data.

The time series for the training of the NN used in this work are those obtained by Jaensch et al. [33]. The excitation signals used are of broadband nature as shown in Fig. 4.2. The signals allow for a broadband, low-pass filtered excitation spectrum with constant amplitude across the frequency range of interest [101]. A non-Gaussian simulation method is applied to generate the random signal time series based on a prescribed power spectrum and a cumulative distribution function [102]. u' is the area averaged velocity measured upstream of the burner plate. It is normalized by extracting the temporal average. The normalized excitation amplitudes are 0.5, 1.0 and 1.5 for the three datasets. These are subsequently called datasets A, B and C, respectively. Despite having similar PDF of amplitudes, all three forcing signals are generated independently and are statistically independent and uncorrelated. The heat release rate fluctuations are integrated over the entire flame and then normalized to get the $\dot{q}'$ time-series. It should be noted that the accuracy of the NN model depends on the accuracy of the training data used. Hence, the quality of the CFD simulation needs to be ensured. In the present study, given the laminar state of the flame and the fact that the CFD resolves all thermochemical and flow time and lengthscales, the data is accurate to be used for the training of the NN model.

4.3 Neural Network Architecture

In this section, we formalize the problem statement and provide the implementation details of the neural network model.

Given a time series data $\mathcal{D} = \{u'(t), \dot{q}'(t)\}_{t=0}^{m\Delta t}$, where $u'(t) \in \mathbb{R}$ and $\dot{q}'(t) \in \mathbb{R}$, the objective is for the neural network to find a mapping $\mathcal{F}$ such that

$$\dot{q}'(t) \approx \mathcal{F}(u'(t)|u'(t-\Delta t), ..., u'(t-n\Delta t)) \quad \forall t \tag{4.1}$$

where $\Delta t = 15\mu s$ denotes the sampling time. The main assumption is that the output at any point in time t, i.e. $\dot{q}'(t)$ depends not only on the input u' at time t, but also on some history of input as formulated in Eq. 4.1, which reflects the time-lagged nature of the system. Here, we consider at least 10 ms of history as it is the characteristic time scale of the response of the system [103]. To solve this problem a Multi-Layer Perceptron is used and its architecture is described next.

4.3.1 Non-linearity through Activation Functions

A multi-layer perceptron is a feed-forward network mapping the input to the output as represented in Fig. 4.3 where an input layer, two hidden layers and an output layer are represented. Here, the input $\underline{u'} = [u'(t), u'(t-\Delta t), ..., u'(t-n\Delta t)]$ is mapped to the output $\dot{q}'(t)$ via the concatenation of nonlinear functions. The predicted output of the MLP is denoted by $\dot{q}'_p(t)$. It is termed as a feed-forward network as the output is not fed back into the input layer. Each hidden layer consists of neurons which are fully connected. This means that each and every neuron in layer l_{i-1} is connected to all neurons in layer l_i through a weight matrix W_i. Hence, the intermediate output of the hidden layer i can be written as $Z_i = W_i^T X_{i-1} + b_i$ where X_{i-1} is the output of layer $i-1$ and b_i is a bias term. In MLP, the non-linearity is introduced by using an activation function g and, therefore, $X_i = g_i(Z_i)$ becomes the final output of layer i. The activation function is one of the most important choices in the design of MLP and

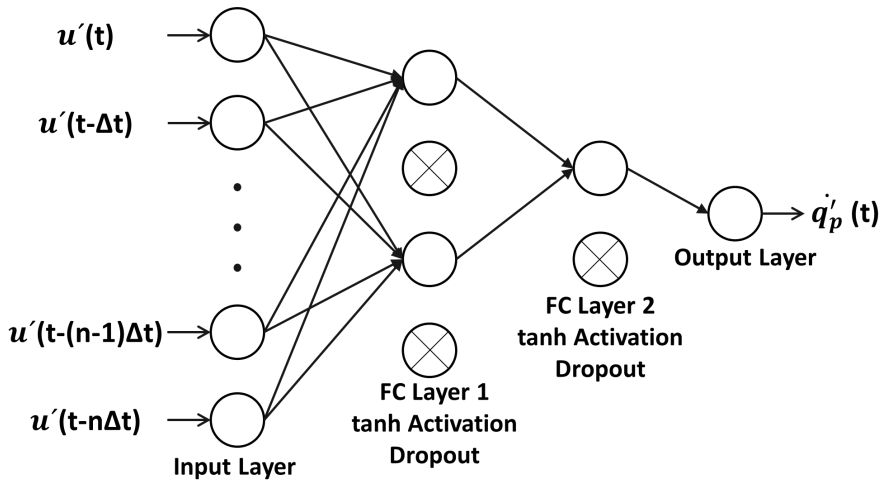


Figure 4.3: Typical representation of a neural network model with two hidden layers and dropout regularizer.

the main options are usually the logistic function (sigmoid), hyperbolic tangent (tanh)

or rectified linear unit (ReLU). Here, the tanh activation function is chosen because it generally allows for a faster convergence of the training of the NN than the sigmoid function [104]. Though ReLU is found to be better than tanh for many applications [105, 52], functions approximated with tanh activation units are smoother than ReLU which makes tanh more appropriate for the present work. The tanh activation function is used for all hidden layers, whereas the linear activation function is adopted for the output layer.

4.3.2 Dropout as Regularizer

The performance of many machine learning algorithms, neural networks included, suffers from overfitting [33]. That is why it is necessary to regularize the model to reduce the risk of overfitting and improve the generalization error. One way to "regularize" a model is to combine several models with different architectures or different data [106]. However, it is computationally expensive to find optimal hyperparameters for each architecture and there may not be enough data available to train the network on different subsets of the data. The dropout regularizer addresses both of these issues. The key idea is to randomly drop units along with their input and output connections from the neural network during the training as shown in Fig. 4.3. This prevents the units from co-adapting too much [106]. The dropout rate is one of the design parameters and, in this work, it is set at 0.5 for the hidden layers. That means that neurons in the hidden layers are dropped with a probability of 0.5 during training. This was found to be an efficient value for this work.

4.3.3 Neural Network Architecture

To train the neural network, a loss function based on the mean squared error (MSE) is used:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(\dot{q}'_p(i) - \dot{q}'(i)\right)^2 \tag{4.2}$$

where $\dot{q}'_p$ is the prediction from the neural network. The network parameters (weight matrix W and bias b) are updated in response to the loss function using the Adam optimizer with the learning rate as one of the hyperparameters.

In total, in order to get optimal performance, the design of the MLP requires to choose the values of these hyperparameters: the number of hidden layers, the number of hidden units and the learning rate of the optimizer. Generally, a grid search is widely used for hyperparameter optimization but it is computationally expensive. So, here, a random search is used as it is found to be efficient and inexpensive compared to grid search [107]. Neural networks with up to 4 hidden layers and a maximum of 200 neurons in the first hidden layer were explored and the learning rate was log-linearly sampled from the range $[0.0001, 0.001]$.

The present study uses python - TensorFlow (`http://tensorflow.org/`) as a framework. Contrary to the modelling approach of Jaensch et al. [33], where they use the default implementation of artificial neural networks in Matlab R2015b. Additionally, we

explore here deeper (more no. of hidden layers) and wider (more no. of neurons per layer) networks compared to [33]. Regularization is also found to be crucial to avoid overfitting. We use dropout as a regularizer, whereas [33] did not use any form of regularization. These differences between our work and [33] may explain the significant differences in the corresponding results.

4.4 Training Details

Figure 4.2 shows the three datasets used to train and test the neural network models. Each dataset contains 1s of inlet flow velocity fluctuations and associated heat release rate fluctuations data. For the training and testing, each dataset is divided into 0.7s for training and 0.3s for testing, where the last 20% of the training data is used for validation. As described in section 4.3, the input data is pre-processed to include the history of inputs. This transforms the input data into a higher dimensional space and is fed to the neural network in batches. The model with the smallest MSE on the validation set is taken to be the best and its performance is evaluated against the test set. For the three datasets, the models are trained for different hyperparameters settings and the best performing model is stored for further use. Table 4.1 shows the number of hidden layers, neurons per layer and number of trainable parameters (weights) for the best performing neural network. A dropout layer is used between each hidden layers. The last layer of the network is the output layer with 1 neuron. Figure 4.4 shows the prediction of the best trained model against the reference from CFD on test data for all three datasets. This figure highlights the good agreement between the NN model and the CFD on the dataset available.

Table 4.1: Details of the best performing MLP for all datasets.

Dataset	# Hidden Layers	# Neurons per layer	Total weights
A	2	[73 36]	75774
B	2	[92 46]	96417
C	3	[128 64 32]	138497

4.5 Results and Discussion

4.5.1 Forced Response

To further assess the trained neural networks, the FDF is obtained from them and compared to CFD results. To do so, the trained neural networks are forced with mono-frequent excitation of different amplitudes. Frequencies from 10 to 500 Hz and amplitude levels of 0.5, 1.0 and 1.5 are considered. Figure 4.5 shows the comparison of the FDF computed from the best neural network models and the CFD simulation. It is seen that all the NN models capture the FDF at amplitude = 0.5. However, only the NN trained on dataset C captures the FDF accurately for all amplitudes. The networks trained

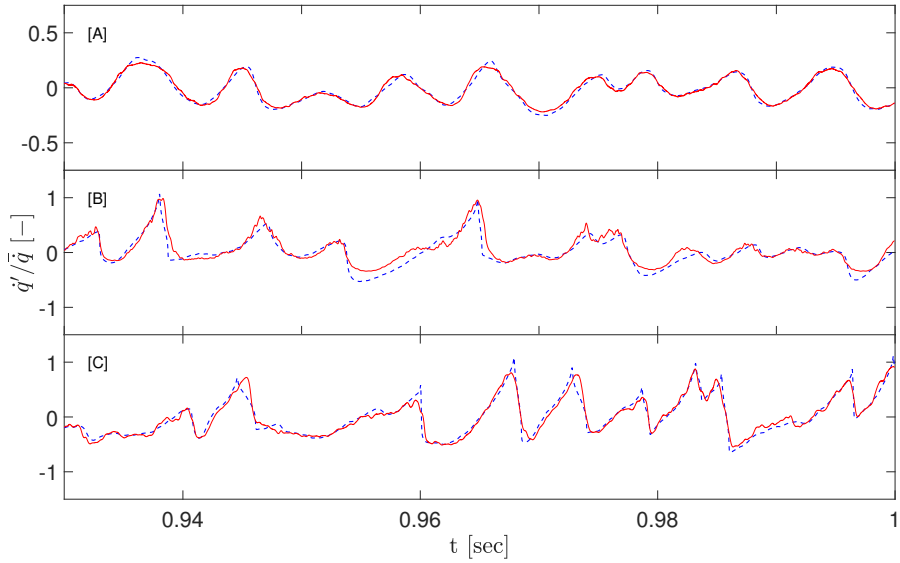


Figure 4.4: Validation of the forced response of the best fit neural network model against broadband time series for dataset A, B, C from top to bottom. Solid line: prediction of heat release rate fluctuations by NN model on test data. Dashed line: CFD reference.

on datasets A and B fail to capture the higher amplitude flame response because those datasets lack the necessary amplitude information as shown in Fig. 4.2. For dataset C, all amplitude levels are present and quite uniformly distributed. Therefore, that dataset contains all the necessary information that the NN needs to learn the nonlinear flame response. However, as the phase does not vary much with different amplitude levels, the networks trained on all datasets are able to predict the phase of the FDF satisfactorily. The shaded regions in Fig. 4.5 show the bounds of prediction, using the estimates of the 5 best performing NNs. The dataset C exhibits tighter bounds, which indicates less uncertainty around the prediction.

As discussed in the introduction, the FDF neglects the excitation of higher harmonics and the ability of the NN to do this also is assessed here. The trained NN-model is excited with 3 different sinusoidal signals: each with a frequency of 100 Hz and amplitudes of 0.05, 0.1 and 0.5 respectively and the results are compared with the equivalent excited CFD simulation. Figure 4.6 shows the FFT contents of the heat release rate fluctuations at the fundamental frequency and its higher harmonics. At 0.05, the output is purely sinusoidal. For amplitudes of 0.1 and 0.5, higher harmonics start to appear which is a trend that the NN-model also predicts. This behaviour is consistent across all other frequencies. This shows that the proposed neural network is capable of learning more complex interactions between harmonics than the FDF.

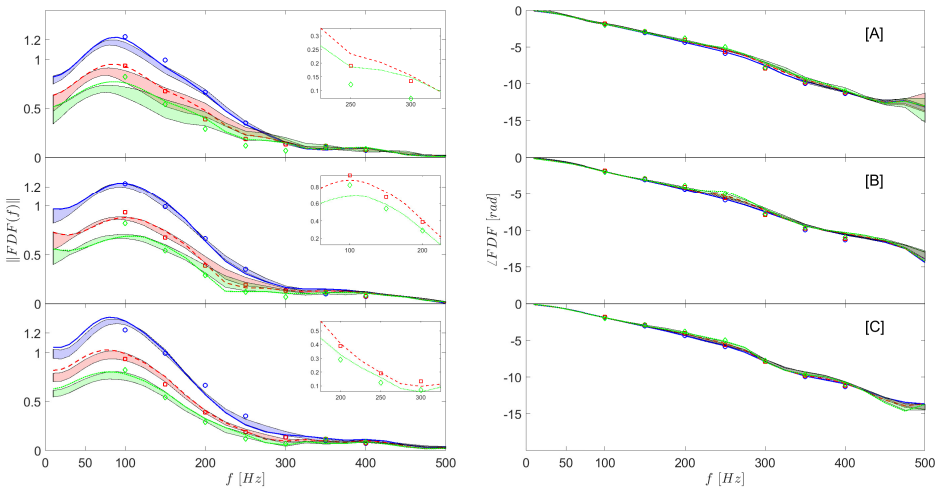


Figure 4.5: Comparison of the FDF computed from the best NN model and from the CFD simulation for dataset A, B, C from top to bottom. Left: Gain; Right: Phase; Lines: estimate by the best NN fit model. Markers: CFD reference using mono-frequent excitation. Shaded area: Bounds of prediction by top 5 NN models. Excitation amplitudes: 0.5 (solid line, circle), 1.0 (dashed line, square), 1.5 (dotted line, diamond).

4.5.2 Coupling of Neural Network and Acoustic Network

From the previous sections, it was observed that only the neural network trained with dataset C can appropriately reproduce the FDF for all frequencies and amplitude levels under consideration. The lack of accuracy of the NNs trained with datasets A and B might lead to an inaccurate evaluation of the acoustic limit cycle. Therefore, for the rest of the analysis, only a neural network model trained on dataset C will be used for further processing and to assess whether it can accurately predict self-excited oscillations. To do this, the trained neural network model is integrated into an acoustic network model through flow velocity fluctuations and heat release rate fluctuations [108], as shown in Fig. 4.7, where the overall network represents a flame in a duct. Additional details about the acoustic network model can be found in Emmert et al. [2].
In this section, the results of the coupled neural network and acoustic network models are compared with the hybrid CFD/low-order model by Jaensch et al. [4]. In that work, the flame response was obtained using CFD. To study the stability of the system, the length of the plenum duct is varied from 50 mm to 1000 mm and the amplitudes of the velocity fluctuations are collected.

Figure 4.8 shows the RMS value of the flow velocity fluctuations measured at the upstream side of the burner plate and the dominant frequency of the self-excited oscillations for different plenum lengths. It is observed that the system becomes unstable at

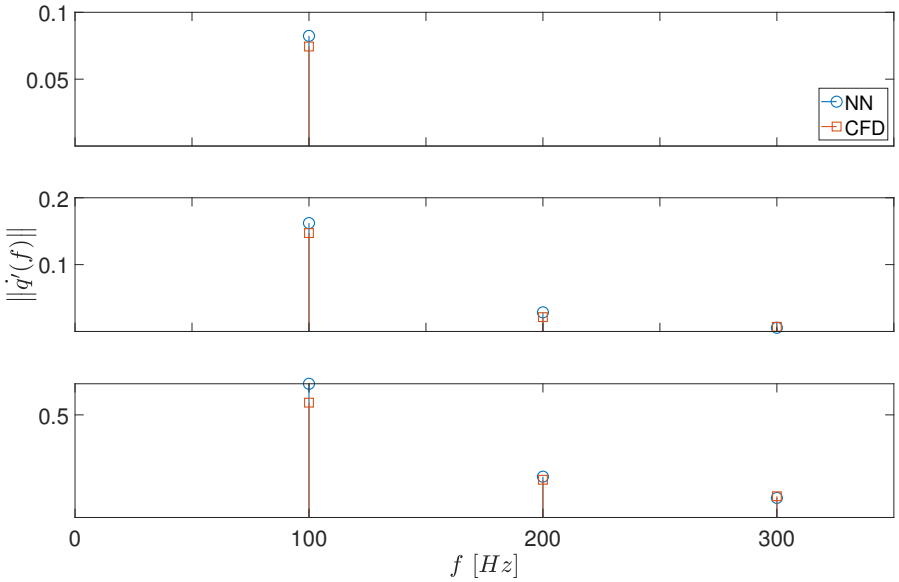


Figure 4.6: Comparison of FFT component of heat release rate fluctuations at different harmonics for an excitation signal with frequency = 100 Hz and amplitudes from top to bottom 0.05, 0.1, 0.5. Circle: NN model. Square: CFD reference.

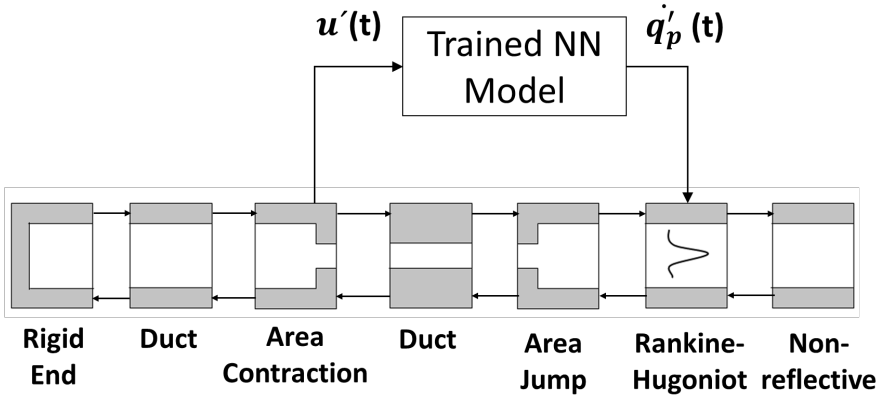


Figure 4.7: Integration of acoustic network and neural network models.

a plenum length of 160 mm which is in agreement with the CFD results from [4]. Additionally, the neural network model shows a good agreement with the CFD results with a slight over-prediction for the longer plenum. Furthermore, the NN model also exhibits the same trend as the CFD results where, initially, the second harmonic is dominant and, as the plenum length increases, the fundamental frequency becomes dominant (see Fig. 4.8 at frequencies for plenum lengths between 200 mm and 300 mm). Finally, the shape and frequency response of the obtained self-excited oscillations for two different plenum lengths are shown in Fig. 4.9. The snapshots of the time series are taken once

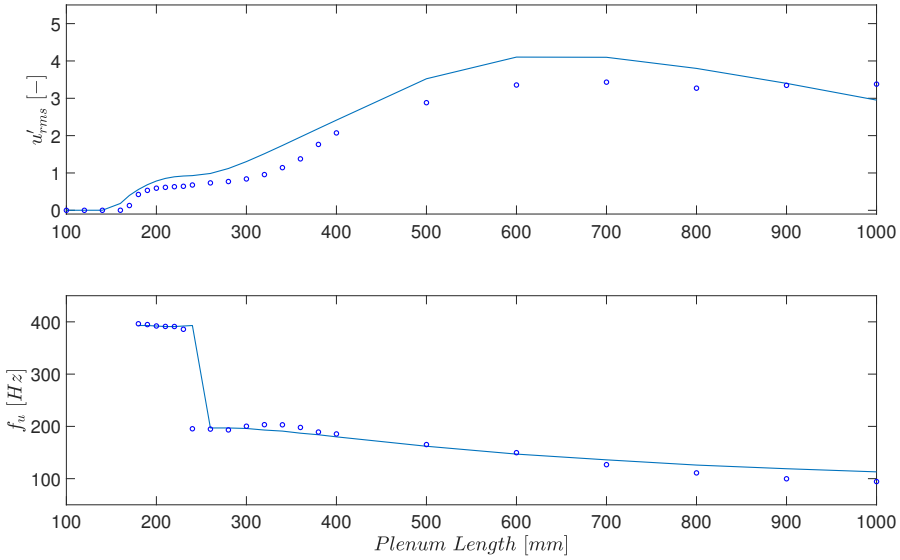


Figure 4.8: Comparison of RMS and dominant frequency of the reference velocity for different plenum lengths. Line: estimate by the NN model. Markers: CFD reference.

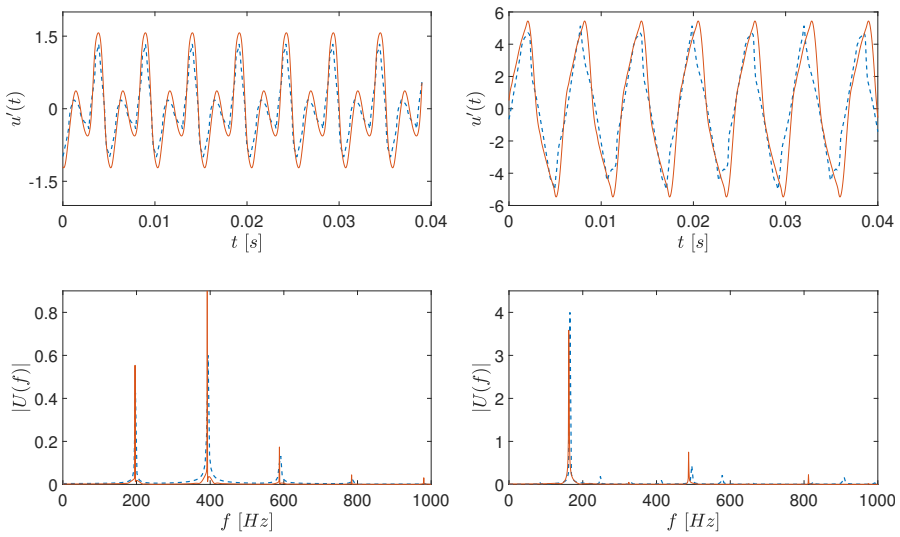


Figure 4.9: Time series (Top), power spectrum (Bottom) of the velocity signal for neural network model integrated with an acoustic network (NN model) and the hybrid/low-order model by Jaensch et al (CFD reference) for two plenum lengths (L). Left: L = 200 mm, Right: L = 500 mm. Solid Line: NN model, Dashed Line: CFD reference.

the steady limit-cycle state is reached. An excellent agreement is observed between the neural network based approach and the CFD for both plenum lengths, with the appropriate frequency being dominant in the response signals. This shows that the NN-based

approach can provide additional information compared to the FDF method as it also produces harmonics in its response.

4.6 Discussion

Multi-layer perceptrons with different number of hidden layers and variable neurons were explored to model the nonlinear flame response of a Bunsen-type flame. The neural network models showcase the ability to learn the FDF for the laminar premixed flame under study while using only one CFD simulation. The simulation is excited with a broadband signal characterized by a uniform distribution of all amplitudes of interest. The trained neural network model captures the flame response not only at the input frequency but also at higher harmonics. When coupled with an acoustic solver, the trained NN could accurately reproduce the bifurcation diagram and the limit-cycle amplitude, shape and frequency contents when compared to the hybrid CFD/acoustic solver of Jaensch et al. [4].

One of the limitations of the proposed methodology is that it uses 1 s of simulation data for training. Obtaining 1 s of simulation data might be computationally time consuming. Therefore, study of length of minimum data required to obtain accurate flame response would be a good future research direction. Another short coming of the proposed model is that the predicted flame response does not obey the low frequency limit (see 4.5). The proposed methodology does not enforce any physical constraints on the predicted integrated heat release rate fluctuations, therefore the low frequency limit is not captured accurately. Following work of Yadav et al. [66] has shown that the loss function can be constrained appropriately to obtain physically accurate flame response at low frequency amplitudes. Such constraints can be added in the future work.

5

Modelling Flame Response of Turbulent Flames

Modelling the flame response of turbulent flames via data-driven approaches is a challenging task due to, among others, the presence of combustion noise. Chapter 4 has shown the potential of neural network methods to infer the linear and nonlinear flame response of laminar flames when externally forced with broadband signals. This work is an extension of that study to analyse the ability of NN models to evaluate the linear and nonlinear flame response of turbulent flames. In the first part of this work, the NN is trained to evaluate and interpolate the linear flame response model when presented with data obtained at various thermal conditions. In the second part, the NN is trained to infer the nonlinear flame response model when presented with a time series exhibiting sufficiently large amplitudes. In both cases, the data is obtained from a large eddy simulation of an academic combustor when acoustically forced by broadband signals. Following work was presented in the *Symposium on Thermoacoustics in Combustion : Industry meets Academia* and is published in *International Journal of Spray and Combustion Dynamics* by Tathawadekar et al. [109].

5.1 Numerical Setup

The present work investigates the flame dynamics of two turbulent, swirl-stabilized combustors, namely the fully premixed EM2C and ETH burners. Schematic representations of both are given in figure 5.1 and 5.2, where the computational domains are highlighted in colour. Geometrical and numerical details of both burners are summarized in table 5.1. The numerical validation of the EM2C burner was carried out by Merk et al. [5] and Kulkarni et al. [34], for more information, the reader is referred to these works. The combustor back plate was defined as a no-slip isothermal wall. In the work of Kulkarni et al. [34], the back plate temperature of the combustor is varied in the range of 700 K to 960 K to quantify the uncertainty in the flame response model. The respective data is used for further analysis to predict the FTF at unseen boundary conditions with the present NN model.

Table 5.1: The two burners under investigation.

Name	EM2C	ETH
LES code	AVBP (fully comp.)	OpenFOAM (low Mach)
Subgrid model [110]	WALE	WALE
Combustion model [111]	DTFLES	DTFLES
Chemistry scheme [112]	2S-BFER	2S-BFER
Mesh size	19×10^6	6×10^6
Swirler	radial	axial
Fuel	CH_4	CH_4
P_{th} [kW]	5.5	48
ϕ [-]	0.82	0.77

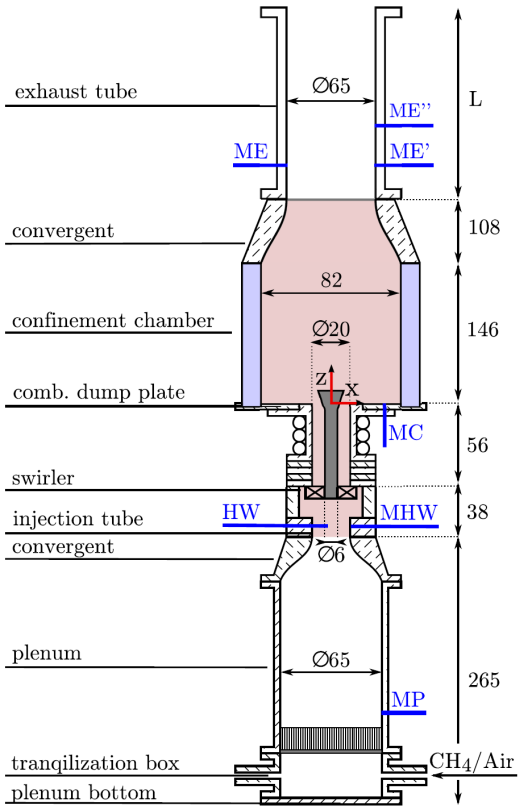


Figure 5.1: Schematic of the EM2C turbulent swirl combustor. Dimensions are given in millimeters. Figure adapted from Merk et al. [5].

The large eddy simulation of the ETH burner [6] is performed using the finite volume-based OpenFOAM 10 library [113] solving the reactive Navier-Stokes equations. To

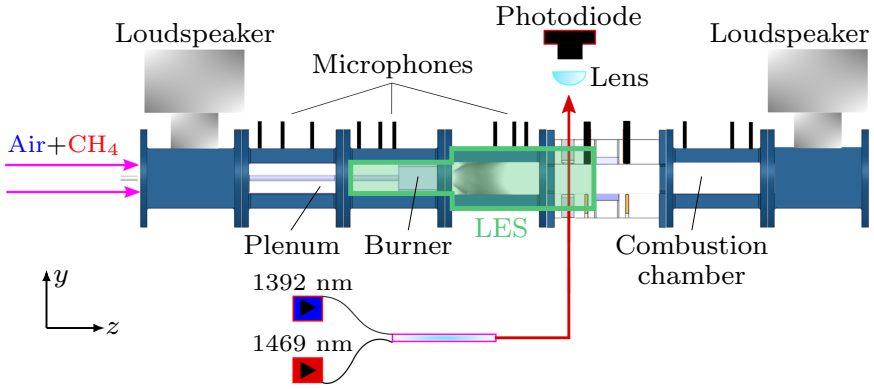


Figure 5.2: Schematic of the ETH turbulent swirl combustor. Figure adapted from Eder et al. [6].

decouple acoustics and flame response and therefore avoid self-excited instabilities or resonances in the system [114], a low Mach number formulation of the `reactingFoam` solver is utilized in the present work. The time integration is achieved using a fully-implicit, second-order accurate Euler scheme. The velocity-pressure coupling is solved with the PISO-consistent algorithm, with three inner iterations. The time step is $\Delta t = 10^{-6}$ s, which assures a Courant-Friedrich-Lewis (CFL) number below 0.3 in the flame region. The numerical domain is discretized with 6 million hexahedral elements and is refined in the swirler and flame region. The boundary conditions consist of velocity inlet and total pressure outlet, while all walls are treated as no-slip. Isothermal walls with a fixed temperature of 700 K are set for the combustor walls and bluff body plane. The small-scale fluid motions are accounted for by the WALE [110] sub-grid scale model. To consider turbulence-flame interaction, an extended version of the dynamic thickened flame model (DTFLES) [111] is utilized. The two-step global mechanism BFER by Franzelli et al. [112] is applied for modelling chemical reactions. The species and energy equations are modelled with a unity Lewis number and a mixture Prandtl number of $\mathrm{Pr} = 0.7$, which is a common approximation in hydrocarbon combustion.

5.2 Methodology

In this section, we formalize the problem statement and provide the pre-processing of data and details of the neural network model.

5.2.1 EM2C burner: problem formulation

Using the EM2C burner, LES simulations are carried out for 12 different back-plate temperatures in the range of 700K to 960K. Each simulation generates the time series of broadband, low-pass filtered velocity signal and corresponding heat release rate fluctuations. The neural network model is trained to predict the heat release rate fluctuations

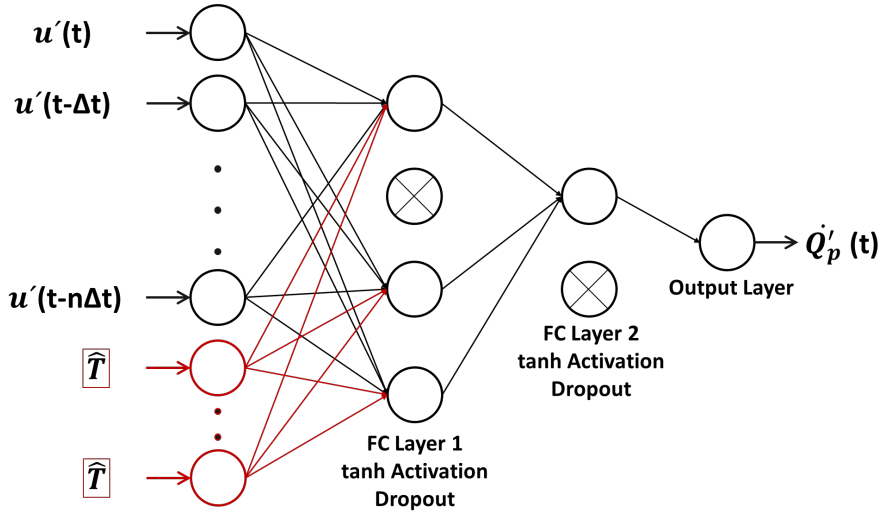


Figure 5.3: Schematic of the neural network model used for both the case studies. The portion in black shows the NN used to model the formulation in Eq. (5.3) for the ETH burner case. For the EM2C case, we provide additional information on the normalized back plate temperature ($\hat{T}$) as formulated in Eq. (5.1).

given the data of inlet velocity and the back plate temperature (T). The heat release rate fluctuations at time t ($\dot{Q}'(t)$) corresponds to the velocity fluctuation not only at time t ($u'(t)$) but also some history of it. Therefore, the mathematical formulation is given as follows :

$$\dot{Q}'(t,T) \approx f(u'(t), T | u'(t-\Delta t), ..., u'(t-n\Delta t)) \quad \forall t \tag{5.1}$$

where the value of n depends on the history of the input signal used. Table 5.2 lists the values used for different cases. To evaluate the ability of a neural network in modelling the flame response of a combustor with unseen back plate temperature, we divide the data into training, validation and test datasets. The data of $T_{\text{train}} = \{700, 786, 800, 860, 926, 960\}$ is used for training the neural network model. The data of $T_{\text{validation}} = \{740, 842, 885\}$ and $T_{\text{test}} = \{760, 823, 905\}$ is used for validation and test. Before feeding it to the NN model, the data is pre-processed as follows. Given the data $\mathcal{D} = \{u'(t), \dot{Q}'(t,T)\}_{t=0}^{m\Delta t}$, for different temperature configurations, the velocity and heat release rate is mean normalized using $\bar{u}$ and $\bar{\dot{Q}}$, respectively. Due to the time-lagged nature of the combustion system, the input time series is transformed to include the history of velocity as shown in Eq. 5.1 and Fig. 5.3. We have data from multiple time series, each corresponding to a simulation with a different back plate temperature. Therefore, the neural network model requires an indicator to distinguish between the data of different temperature configurations. We classify input data by appending it with the normalized temperature value ($\hat{T}$). Therefore, the data of training temperatures is concatenated in addition to the history of velocity fluctuations in the input layer of the neural network model. As shown in Fig. 5.3 the value of the normalized temperature is repeated multiple

times in the input layer to emphasize that feature. It leads to better model performance. In our implementation, we have repeated this value 100 times for all input instances. This pre-processed data is shuffled and then fed into the neural network model.

Table 5.2: Hyper-parameters of the NN model

Case	Learning Rate	Batch Size	train:test	History ($n\Delta t$)
`EM2C`	10^{-3}	3000	NA	10 ms
`ETH Model L`	10^{-5}	2750	70:30	10 ms
`ETH Model NL`	5×10^{-5}	2250	80:20	10 ms

We use a multi-layer perceptron [61] to model the flame response of the turbulent swirl combustor. Fig. 5.3 shows the schematic of a typical multi-layer perceptron with an input layer, two hidden layers and an output layer. The hidden layers and output layer have parameters associated with them, called weights W and biases b. The outputs of i^{th} hidden layer are computed as, $Z_i = \sigma(W_i Z_{i-1} + b_i)$, where Z_{i-1} are the outputs of the previous layer and σ indicates activation function. The activation function can be used to introduce nonlinearity to the affine transformation of $W_i Z_{i-1} + b_i$. Therefore, the key design considerations to build a neural network model are the number of hidden layers, the number of neurons in each hidden layer, and the activation function to be used. These are called hyper-parameters of the model. Several models with varying hyper-parameters are trained, and the model with the least validation error is selected to predict the flame response. The hyper-parameters considered are: no. of hidden layers, no. of neurons per layer, learning rate and batch size. Table 5.2 gives the details of the hyper-parameters used. For all the cases considered in this work, we use a NN model with three hidden layers, $[128, 64, 32]$ neurons in each layer respectively. The learning rate and batch size were identified as important hyper-parameters. The grid search [115] is performed to exhaustively find the best combination of these values. Learning rate is sampled from $\{10^{-5}, 5 \times 10^{-5}, 10^{-4}, 5 \times 10^{-4}, 10^{-3}\}$. The batch size is sampled from the range of $[1500, 4500]$ with a step of 50. We explored values of 10ms, 12ms and 15ms for the history ($n\Delta T$) provided to the neural network model. No. of hidden layers and no. of neurons are efficiently found using Random search [107]. tanh is used as an activation function. The dropout [106] is used as a regularizer and is set to 0.5 for all the hidden layers. The neural network model is trained using the Adam optimizer [116] with a suitable learning rate. The mean squared error loss function is used to model the flame response of EM2C burner with unseen back plate temperature. It is given by,

$$\text{MSE} = \frac{1}{m} \sum_{t=0}^{m\Delta T} (\dot{Q}'(t) - \dot{Q}'_p(t))^2 \tag{5.2}$$

where m is the number of data points and $\dot{Q}'_p(t)$ is the heat release rate predicted by the neural network model.

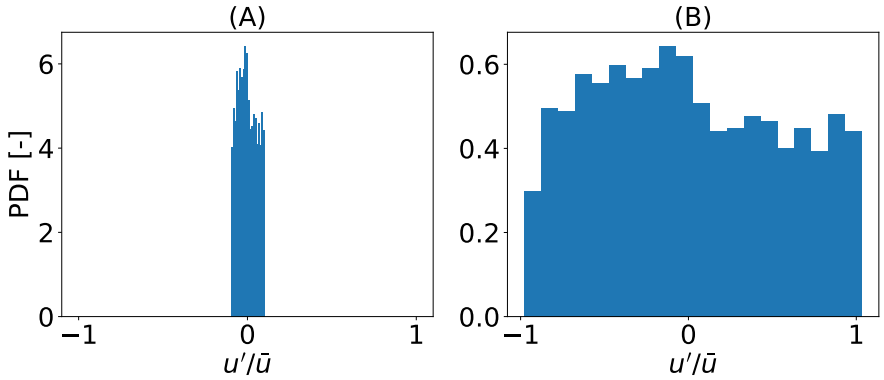


Figure 5.4: Probability density function of the input velocity signal with (A) 0.1 and (B) 1.0 amplitude fluctuations.

5.2.2 ETH burner: problem formulation

For the ETH burner, NN models are trained using the time series of u' and $\dot{Q}'$. The neural network model is trained to solve the following regression problem:

$$\dot{Q}'(t) \approx f(u'(t)|u'(t-\Delta t), ..., u'(t-n\Delta t)) \quad \forall t \tag{5.3}$$

The signal's sampling rate (ΔT) is $10^{-6}s$ and is increased to $10^{-5}s$ for efficient processing by the NN model. The length of the time series is 300 ms. Two datasets of different excitation amplitudes, 0.1 and 1.0, are generated, as shown in Fig. 5.4. We train two NN models, one to predict the linear flame response and another to predict the nonlinear flame response. We will refer to these models as "Model L" and "Model NL" respectively, throughout the paper. Model L uses data coming from 0.1 excitation amplitude. When trained on only 1.0 amplitude data, Model NL fails to accurately capture the low amplitude response. Therefore, data from 0.1 and 1.0 excitation amplitudes are concatenated to train Model NL. The input and output data are mean normalized using $\bar{u}$ and $\bar{\dot{Q}}$, respectively. The velocity data is transformed to include the history of velocity observations as shown in Eq. (5.3) and Fig. 5.3. The time series data is divided into the training, validation, and test datasets without shuffling. Table 5.2 lists the training and test split up used. The training data is divided into the training and validation datasets using 80 : 20 split up. Similar to the EM2C setup, a multi-layer perceptron is used to model the linear and nonlinear flame response of the ETH burner. Table 5.2 lists the hyper-parameters used to train Model L. The Adam optimizer with the learning rate of 10^{-5} is used to train the MLP. The mean squared error is used as a loss function.

Model NL, which combines the data from two different amplitude levels, provides unsatisfactory results when trained with the MSE loss functions due to the amplitude imbalance problem. The MSE loss function computes the average of the squared differences between the predicted values and the true target values as seen in Eq. (5.2). However, the squared term in the loss function can lead to the problem of the model being more sensitive to large errors (associated with larger amplitudes) than to smaller

errors (associated with smaller amplitudes). During back-propagation, gradients are scaled by the derivative of the loss function. The gradients can become much larger for larger errors, thus emphasizing the model's focus on large amplitudes. This can lead the model to prioritize reducing errors for larger amplitudes, which can predict large amplitudes more accurately than smaller ones. To alleviate this issue, we tested different loss functions that do not overly amplify the errors. Such loss functions include mean absolute percentage error (MAPE), normalized mean squared error (NMSE), normalized root mean squared error (NRMSE), and Huber loss. Taking inspiration from these loss functions, we propose a novel loss function that performs best for the underlying data. A novel loss function called *mean square root error (MSQRE)* is formulated as:

$$\text{MSQRE} = \frac{1}{m} \sum_{t=0}^{m\Delta T} \sqrt{|\dot{Q}'(t) - \dot{Q}'_p(t)|}. \tag{5.4}$$

5.3 Results

In this section, we showcase results obtained on two tasks of flame response modelling. First, we assess the model performance in predicting the FTF of a turbulent flame at unseen thermal boundary conditions. Then, we showcase the ability of NN models to capture the FTF and FDF of a turbulent flame.

5.3.1 Predicting FTF of a turbulent flame at unseen boundary conditions

Time series data obtained from 6 LES simulations, at different thermal boundary conditions (T_{train}), are used to train the neural network model. It is validated on 3 different combustor configurations ($T_{\text{validation}}$). The best performing model gives the MSE of 0.0068. This trained neural network is used to predict the FTF of a turbulent flame with unseen back plate temperature (T_{test}). To compute the FTF, mono-frequent harmonic excitation signals with 0.1 amplitude are generated in the frequency range from 10 Hz to 500 Hz with 10 Hz resolution. The trained neural network model is used to predict the output for each excitation signal. The fast Fourier transform (FFT) is applied to obtain the gain and phase information of the frequency of interest. Fig. 5.5(A) shows the prediction at 760K, which belongs to the test set. The FTF obtained by the NN model is compared against the FTF coming from the CFD/SI approach [117]. Note that the latter requires data at that temperature: the CFD/SI approach is not capable of evaluating the flame response at unseen conditions. In other words, there is no established method that permits to 'interpolate' an impulse response, obtained by the CFD/SI approach, from data obtained at adjacent operating conditions. We observe that the NN model captures the flame response quite accurately. In order to obtain the uncertainty in FTF prediction, we generate multiple realizations of training data using temporal slicing. We slice the input-output time series of 350 ms length into the 5 overlapping time series of length 150 ms each. 5 neural network models are trained using these datasets. Additionally, we train a neural network model with complete data of 350 ms. The gain and phase predictions obtained from these 6 models are used to calculate upper and lower

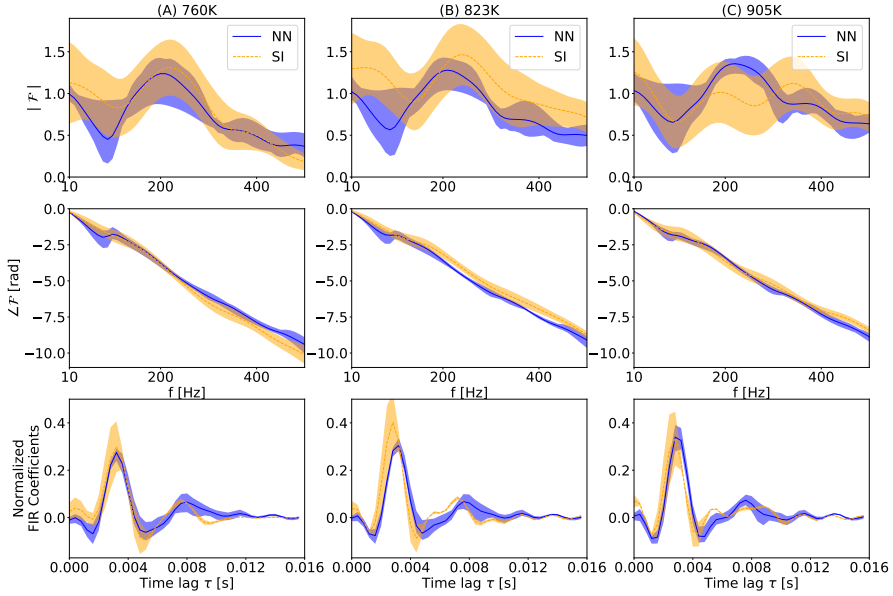


Figure 5.5: EM2C burner: Comparison of gain, phase and normalized FIR coefficients of the linear flame response predicted by CFD/SI and NN approach at unseen, interpolated back plate temperatures of (A) 760K; (B) 823K; (C) 905K. The shaded region in blue shows the bounds of prediction by the NN model using different realizations of the training data.

limits in the FTF prediction. Figures 5.5 (B) and (C) show the prediction bounds of gain and phase obtained by the neural network model for additional test cases of 823K and 905K.

Another equivalent approach to comparing the similarity in the predicted flame response is by means of the Finite Impulse Response (FIR) model [118, 119, 120]. It describes the effect of an impulse perturbation of velocity on the heat release by the flame. It is formulated as,

$$\dot{Q}'(t) = \sum_{i=0}^{n_b} h_i u'(t - i\Delta t) \tag{5.5}$$

where n_b is the number of impulse response coefficients h_i. The third row of Fig. 5.5 compares the estimated FIR coefficients at three corresponding temperatures obtained by the CFD/SI approach and NN model. The FIR is obtained by performing the inverse z-transform of the calculated FTF. The FIR obtained from the NN model is in good agreement with the CFD/SI approach.

In the first two rows of Fig. 5.6, we collect the gain and phase for all 12 configurations using the CFD/SI approach and the neural network model. The phase difference in the resulting NN predictions (Fig. 5.6 (B)) displays that increasing the back plate temperature of the EM2C burner results in a decrease in the phase difference. This finding is in good agreement with [34]. According to that study, higher back plate temperatures cause a reduction in the phase because such high temperatures promote closer stabiliza-

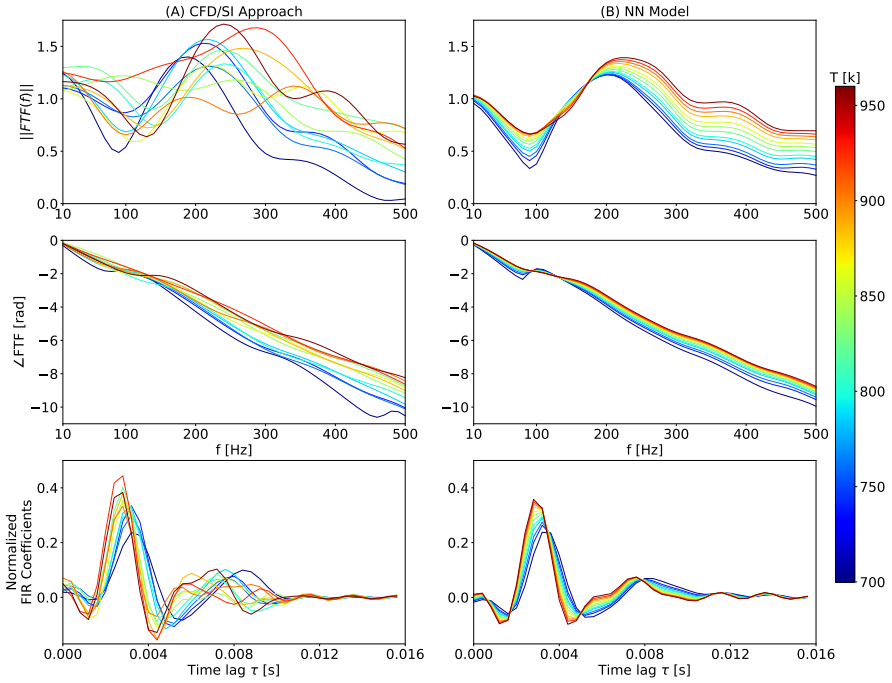


Figure 5.6: EM2C burner: Comparison between mean gain, phase and normalized FIR coefficients of the linear flame response at 12 different temperatures using CFD/SI and NN technique.

tion of the flame to the burner mouth and, overall, a shorter flame length. Furthermore, gain curves reveal that increasing back plate temperatures of the EM2C burner shifts the flame response gain to higher frequencies slightly until 250 Hz. In addition, increased back plate temperatures of the EM2C burner amplify the flame response between 250 Hz and 500 Hz.

A collection of impulse responses, each of them characterizing the acoustic flame response at a given temperature, is shown in the third row Fig. 5.6. Each of the shown impulse responses of the NN model are obtained by applying the inverse z-transform to the frequency responses shown in the first two rows of Fig. 5.6. From the results of Fig. 5.6, it is understood that the flame responds faster and stronger for larger temperatures T. Note also that the valleys of the impulse response are also stronger for higher plate temperatures T. This implies that the flame fluctuations – increase and decrease of the integrated flame surface area – are more preponderant in the case of higher T. For smaller plate temperatures T, crests and valleys strength decrease and are shifted to the right (larger times). Similar observations are obtained when looking at the impulse responses obtained by system identification, although in this case, the gradual change of the impulse response (when shifting from larger to lower values of T) is not as apparent. The agreement of results between NN and the benchmark case (SI) demonstrates the validity of NN results when inferring models of the flame response at temperatures that are not present in the training data.

The reader might wonder how to interpret the mistmatch betwen the CFD/SI and NN when looking at the gain of the flame response. Upon studying the first row of Fig. 5.6, it becomes evident that any resemblance between CFD/SI and NN results is challenging to discern. It is important to note that CFD/SI cannot be considered a reference truth, as this technique can be significantly affected by noise, particularly combustion noise present in the data. This effect is especially pronounced in the EM2C burner under investigation, as demonstrated by Kulkarni et al. [34], and reflected in the high uncertainties observed (refer to the top row in Fig. 5.5). Therefore, any comparison of flame response gain between the LES/SI and NN methods must take into account the underlying uncertainties. In contrast, the level of uncertainty is lower when investigating the phase of the flame response and the flame impulse response, allowing for a comparison between the two methods.

5.3.2 Modelling nonlinear flame response

For ETH burner, the time series obtained using a normalized excitation amplitude of 0.1 (Fig. 5.4 (A)) is used to model the linear flame response of the flame. 30 % of the data is used for testing and 70 % is used for training. Further, 20 % of the training data is used for validation. A neural network model is trained using these time series. The model with the least validation error of 0.0014 is chosen to further predict the FTF. Hyperparameters of the selected model are listed in Tab. 5.2 and are referred to as Model L. Model L is subjected to mono-frequent forcing at 0.1 amplitude and frequencies in the range of 10Hz to 500Hz. Figure 5.7 compares the FTF obtained by the NN model, CFD/SI method, and experimental data available [121]. Overall, a very good agreement

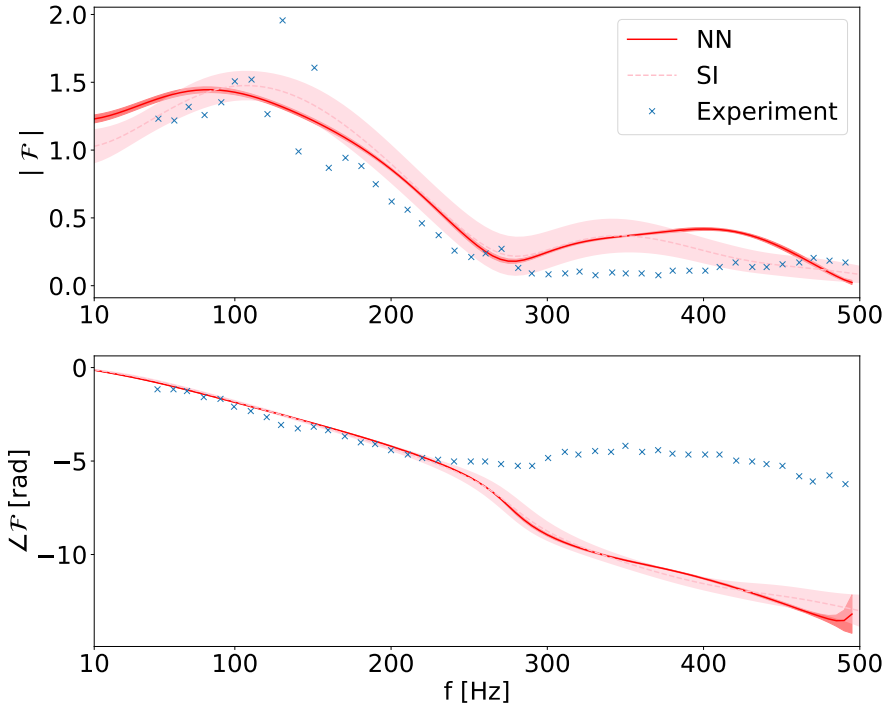


Figure 5.7: ETH burner: Comparison of the FTF predicted by the NN model with CFD/SI and experiment. The shaded region in red shows the bounds of prediction by the top 5 NN models.

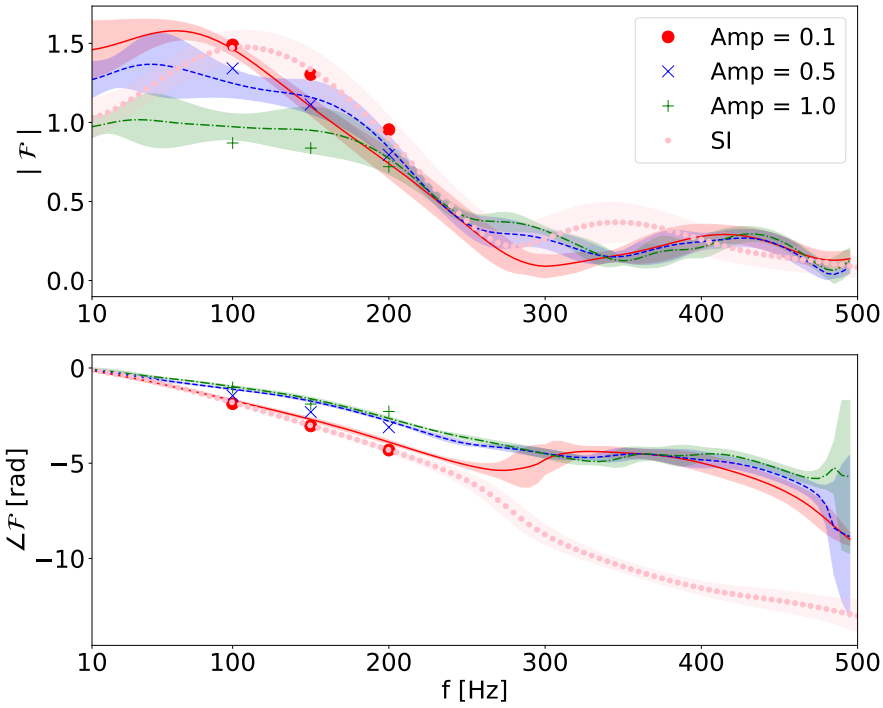


Figure 5.8: ETH burner: Comparison of the FDF computed from the NN model with LES. Solid lines: estimate by the NN model. Markers: ground truth from LES using mono-frequent excitation. The shaded region shows the bounds of prediction by the top 5 NN models.

is found for the gain and phase prediction done by the NN model with the other data sets.

Now, we assess the ability of neural networks to model the nonlinear flame response. We combine the 0.1 and 1.0 excitation amplitude data to train the neural network model. Combining data of 0.1 amplitudes provides NN with more data to train on and improves its prediction at low amplitude levels. The model with the least validation error of 0.2608 is selected to predict the FDF. This task is more complicated than the FTF prediction due to the variation in gain and phase response to the different amplitudes. Figure 5.8 compares the FDF predicted by the neural network model with the data from LES simulations at 0.1, 0.5, and 1.0 amplitude levels. Gains predicted by NN models match accurately with the CFD simulations. A good agreement is observed for the gain predicted by the NN model at 0.1 amplitude level with the CFD/SI approach. Similarly, the NN model captures the variation in phase for different amplitude levels of 0.1 and 1.0. The accurate predictions in gain and phase at low amplitude levels stem from the novel MSQRT loss function used to train the neural network model. Furthermore, we show the bounds of prediction by selecting the top 5 best-performing neural network models from the hyper-parameter study. The predictions by these neural network models are used to obtain the mean and standard deviation in gain and phase predictions shown

in Fig. 5.8. Therefore, we show that the neural network model successfully predicts the linear and nonlinear flame response of the ETH burner under study.

5.4 Discussion

In this work, we demonstrated the ability of the neural network model to predict the linear and nonlinear flame response of turbulent flames for two different burner configurations. For the EM2C burner, the neural network is trained on the data from a few back plate temperature configurations. The trained neural network model can predict the linear flame response at unseen burner configurations with different back plate temperatures. The neural network model with sufficient training data can eliminate the need to run additional CFD simulations to capture the FTF at unseen boundary conditions, thus reducing the computational cost. For the ETH burner, the neural network model is shown to capture the nonlinear flame response. Compared with the data from the mono-frequent excitations, predictions from the NN model are in good agreement. A novel loss function is shown to be useful to improve flame response predictions.

6

Reconstruction of Reactive Flow Fields

Performing measurements in reacting flows is a challenging task due to the complexity of measuring all quantities of interest simultaneously or limitations in the optical access. To compensate for this, recent advances in deep learning have shown a strong potential in augmenting the information content in datasets composed of partial measurements by reconstructing the quantities that could not be measured. This work analyses the use of such deep learning tools. Convolutional neural networks are used to reconstruct the heat release rate (HRR) from velocity measurements in a methane/air premixed flame under harmonic excitation. The CNNs are trained from complete datasets at some specific frequencies and amplitudes of excitation and their ability to reconstruct the HRR for different operating conditions with good accuracy is demonstrated. The work reproduced in this chapter was presented and published in INTER-NOISE and NOISE-CON Congress and Conference Proceedings, InterNoise22, by Tathawadekar et al. [122].

6.1 Test case: Bunsen flame under excitation

The Bunsen flame test case is shown in Fig. 4.1. It is the laminar multi-slit burner investigated by Kornilov et al. [32], where a premixed methane-air mixture with an equivalence ratio of 0.8 is used and is subjected to single tone velocity perturbation with a loudspeaker. The numerical setup and results similar to the one in [65, 33] are used: a 2D CFD domain with symmetric boundary condition in the transverse direction and inflow/outflow boundary condition in the streamwise direction is considered. The two-step chemical scheme as detailed in [100] and OpenFOAM [123] are used for the simulation. Additionally, no turbulent combustion model is necessary here as all species transport equations are fully resolved. A two-step mechanism is a simplified chemistry model. Specifically, the continuity, Navier-Stokes, species mass fraction and enthalpy equations [124] are solved with the CFD solver:

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_i}{\partial x_i} = 0 \tag{6.1}$$

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial \rho u_i u_j}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} + \rho g_i \tag{6.2}$$

$$\frac{\partial \rho Y_k}{\partial t} + \frac{\partial \rho u_i Y_k}{\partial x_i} = \frac{\partial}{\partial x_i}\left[\rho D \frac{\partial Y_k}{\partial x_i}\right] + \rho \dot{\omega}_k \tag{6.3}$$

$$\frac{\partial \rho h}{\partial t} + \frac{\partial \rho u_i h}{\partial x_i} = \frac{\partial}{\partial x_i}\left[\rho \alpha \frac{\partial h}{\partial x_i}\right] \tag{6.4}$$

where ρ is the density, u_i denotes the velocity in the i-th direction, p the pressure, Y_k is the species mass fraction, $\dot{\omega}_k$ its reaction rate and h is the specific enthalpy of the mixture. τ_{ij} is the molecular stress tensor, D is the molecular diffusivity and $\alpha = \lambda/(\rho C_p)$ is the thermal diffusivity, where C_p is the specific heat coefficient. The reaction rate and species transport and thermodynamics properties are taken from the two-step chemical mechanism detailed in [100]. It should be noted that for this case of an excited laminar flame, gravity effects are neglected and therefore $g_i = 0$. The heat release rate is computed as [124]:

$$\dot{q} = -\sum_{k=1}^{N_s} \rho \dot{\omega}_k \Delta h^0_{f,k} \tag{6.5}$$

where $\Delta h^0_{f,k}$ is the formation enthalpy of specie k and N_s is the number of species.

The flame is laminar and the grid is composed of 122 300 cells with a cell size of 0.025 mm in the flame region and area of contractions and cell stretched in the axial direction. This ensures that all flow and reaction lengthscales are fully resolved [4]. At the inflow, a mean inlet velocity of 0.4 m/s and inlet temperature of 293 K are imposed. The plate on which the flame is stabilized is modelled as a no-slip wall with a fixed temperature of 373 K, as measured in the experiment [32]. The CFD simulation is run with an adaptive time-stepping scheme with an average timestep $\Delta t = 10^{-6}$. Given that this CFD set-up has already been validated in previous studies with respect to experimental data, these validation steps are not repeated here. The interested reader is referred to [65, 33] for additional details on the numerical set-up and its validation.

This initial set-up is then subjected to acoustic excitation at the inlet, where a normalized harmonic streamwise velocity fluctuation is imposed. The frequencies considered for excitation are f =100, 150 and 200 Hz with a normalized amplitude of A =10, 50, 100, 125 and 150% of the mean inlet velocity. Different combinations of cases will be used to train the deep learning framework and it will then be tested on a different condition to assess its reconstruction performance.

A typical time-sequence of the varying HRR (normalized by its maximum value) over one period is shown in Fig. 6.1 for the case with $f = 100$ Hz and $A = 100\%$. It can be seen that the initial flame gets extremely elongated under the large amplitude of excitation before regaining its original shape.

6.2 Reconstruction method with CNNs

In this section, the reconstruction objective is the inference of the heat release rate field from the velocity field. To achieve this, a deep learning architecture based on the U-

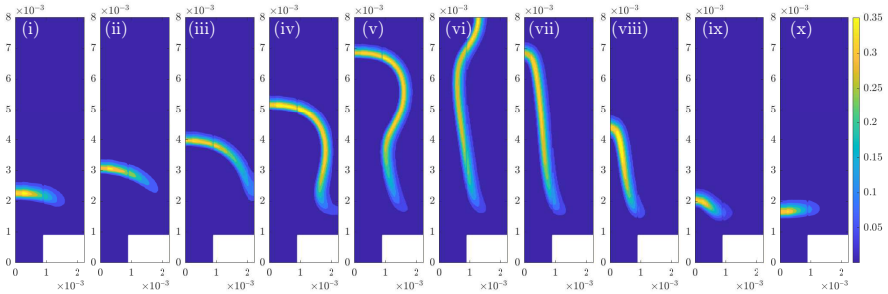


Figure 6.1: Evolution of HRR over one period of excitation for the case with $f = 100$ Hz and $A = 100\%$. Snapshots are spaced by a time-duration which is one-tenth of the excitation period.

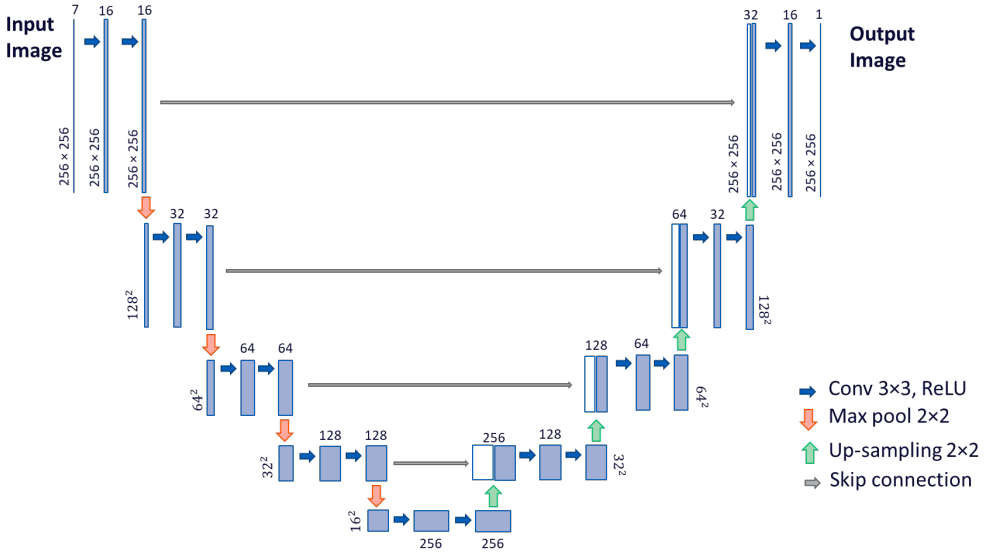


Figure 6.2: Schematic of the U-net architecture

net will be used [125]. This architecture is presented in Fig. 6.2. It is composed of a series of CNNs which perform multi-level filtering of the input fields and recombine these into one output field. For this reconstruction problem, it will be considered that the U-net takes as input the full (2-component) velocity fields at 3 different successive time instants in the past (i.e. $\mathbf{u}(x,t)$, $\mathbf{u}(x,t-\Delta t)$, $\mathbf{u}(x,t-2\Delta t)$) and the mean HRR field (i.e. $\langle \dot{q}(x) \rangle_T$, where $\langle \cdot \rangle_T$ represents the time-averaging operation over the period T of the velocity fluctuation) and outputs the HRR fluctuations $d\dot{q} = \dot{q} - \langle \dot{q} \rangle_T$. This combination of information allows the U-net to estimate velocity gradients, which are necessary to estimate the HRR fluctuation. The network is a special case of an encoder-decoder architecture. In the encoding part, the input data is down-sampled by a factor of 2 using max-pooling layers. This helps to extract large-scale and abstract information. As image size decreases the spatial information is translated into extracted features in

the growing number of feature channels. In the decoding part of the network, spatial resolution increases with up-sampling layers and the number of features reduces. The skip connections help in transferring low-level information on the encoding side to the decoding layers.

In this setup, the U-net architecture requires the data of HRR fluctuations for training which may seem limiting. However, we will show that the U-net is also able to infer the HRR fluctuations for operating conditions not present in the training dataset. This situation would mimic an experimental campaign where measurements of HRR are only available for a few operating conditions while the velocity measurements would be available for all of them. It should be noted that the requirement of having the mean HRR as an input is not excessively constraining given that it can be straightforwardly obtained using luminescent photographs for example.

6.3 Results

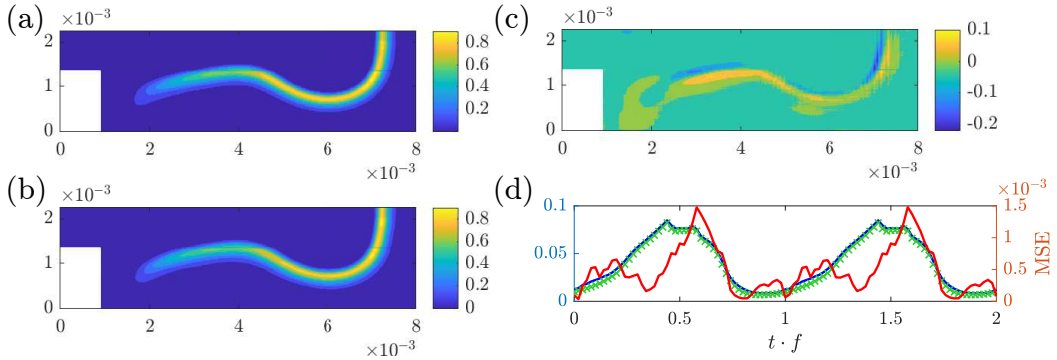


Figure 6.3: (a) Actual HRR field, (b) Reconstruction from U-net and (c) difference between the two. (d) Time-evolution of (left axis) the total HRR fluctuation (blue line: U-net, cross: exact) and (right axis) mean squared error for the amplitude interpolation problem (for the case $f = 100$ Hz and $A = 125\%$) .

Two different reconstruction problems are considered. In the first "amplitude interpolation" case, the U-net architecture is trained with the data at specific amplitudes of excitation (10, 50, 100, 150%) and frequencies of 100 and 200 Hz and the U-net is used to reconstruct the HRR fluctuation for the other cases at different amplitudes of excitation (125%), for the same frequencies. In the second "frequency interpolation" case, the U-net is trained with the case at 2 different amplitudes of excitation (50 and 100%) for frequencies 100 and 200 Hz and it is tested for a case with frequency of 150 Hz and amplitude 50%.

A typical reconstructed HRR field for a representative timestep for the "amplitude" case is shown in Fig. 6.3 alongside the original data, where the U-net reconstruct the HRR field for a case whose amplitude was not present in the dataset ($A = 125\%$). It can be seen that the HRR profile is correctly reconstructed from the input (the velocity fields during several past time steps). Especially the formation of the HRR "cusps", which is

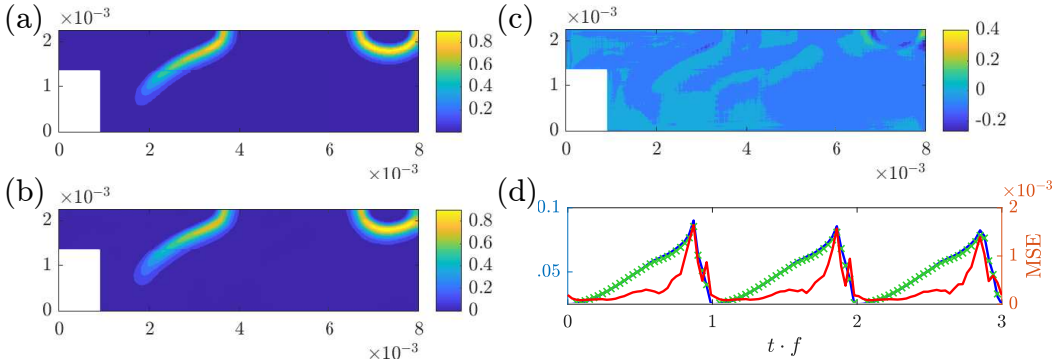


Figure 6.4: (a) Actual HRR field, (b) Reconstruction from U-net and (c) difference between the two. (d) Time-evolution of (left axis) the total HRR fluctuation (blue line: U-net, cross: exact) and (right axis) mean squared error for the frequency interpolation problem (for the case $f = 150$ Hz, and $A = 50\%$).

due to the very large amplitude of the excitation, is recovered. The time evolution of the mean-squared error (MSE) between the predicted HRR field and the exact one is also shown in Fig. 6.3d where it can be seen that throughout the prediction, the MSE remains small. In addition, an important feature of the flame for thermoacoustic studies is the total HRR fluctuation (the HRR integrated over the entire domain, $\dot{Q} = \int_V \dot{q}\, dV$). It is computed using the HRR reconstruction by the U-net and is shown in Fig. 6.3d. It can be seen that it closely matches the actual total HRR fluctuation. This indicates that the U-net 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. The results for the other test case ($f = 200$ Hz and $A = 125\%$) exhibited a similar level of accuracy and are not shown here for brevity.

A similar analysis was performed when considering the frequency interpolation problem, i.e. by requiring the U-net to reconstruct the HRR profile for a case whose frequency of excitation was not present in the training dataset. This is shown in Figs 6.4a-c where the exact, reconstructed HRR profiles and their difference are shown. Similarly to the other case, one can observe that the reconstructed HRR matches well the actual one. The U-net is also able to reconstruct the HRR accurately in an integral sense (see the time-evolution of the MSE and total HRR in Fig. 6.4d). Similar accuracy was also found for the other test case in this problem set-up ($f = 150$ Hz with $A = 100\%$) and the associated results are not shown for brevity.

6.4 Discussion

This work presented a deep learning-based method for the reconstruction of unmeasured quantities in reacting flows using available training data. A U-net architecture is shown to be successful at reconstructing the HRR fluctuation in an acoustically excited flame from past velocity information. The U-net was trained with the HRR dataset for some specific frequencies and amplitudes of excitation and could accurately reconstruct

the HRR for other frequencies and amplitudes, not in that training dataset. These results demonstrate the potential of deep learning techniques in supplementing information when only partial data is available. In the next chapter, we will further explore the exciting problem of modelling reactive flow simulations when underlying physics is unknown.

7 Modelling of Flame Dynamics

Modelling complex dynamical systems with only partial knowledge of their physical mechanisms is a crucial problem across all scientific and engineering disciplines. Purely data-driven approaches, which only make use of an artificial neural network and data, often fail to accurately simulate the evolution of the system dynamics over a sufficiently long time and in a physically consistent manner. Therefore, we propose a hybrid approach that uses a neural network model in combination with an incomplete partial differential equations solver that provides known, but incomplete physical information. In this study, we demonstrate that the results obtained from the incomplete PDEs can be efficiently corrected at every time step by the proposed hybrid neural network – PDE solver model, so that the effect of the unknown physics present in the system is correctly accounted for. For validation purposes, the obtained simulations of the hybrid model are successfully compared against results coming from the complete set of PDEs describing the full physics of the considered system. We demonstrate the validity of the proposed approach on a reactive flow, an archetypal multi-physics system that combines fluid mechanics and chemistry, the latter being the physics considered unknown. Experiments are made on planar and Bunsen-type flames at various operating conditions. The hybrid neural network - PDE approach correctly models the flame evolution of the cases under study for significantly long time windows, yields improved generalization, and allows for larger simulation time steps. Following work is published in *Data-Centric Engineering* journal by Tathawadekar et al. [126]. It is reproduced here under the terms of the Creative Commons Attribution 4.0 License.

7.1 Preliminaries

Modelling and forecasting of complex physical systems described by nonlinear partial differential equations are central to various domains with applications ranging from weather forecasting [127], design of airplane wings [128, 129], to material science [130]. Typically, a chosen set of PDEs are solved iteratively until convergence of the solution. Modelling complex physical dynamics requires a good understanding of the underlying physical

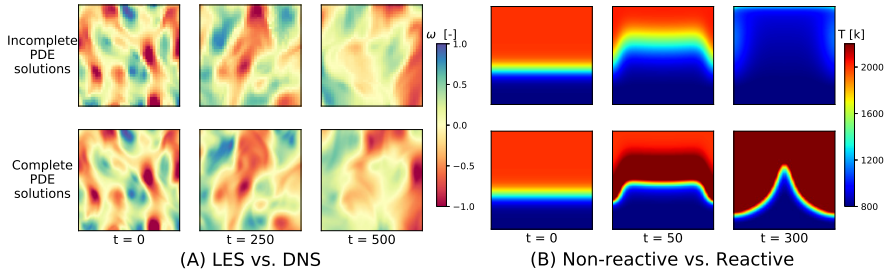


Figure 7.1: (A) The normalized vorticity solutions of complete/DNS (bottom) solver can be reached by increasing the spatial resolution of the incomplete/LES (top) solver [7]. (B) We consider the problem of the incomplete/non-reactive (top) and complete/reactive (bottom) PDE solvers which can yield fundamentally different evolutions, as shown here for a sample temperature field over time.

phenomena. For cases where the complete information on the physics of the system is missing, deep learning models can be employed to complete the physical description when additional data of the system is available. Deep learning methods have shown promises to account for these unknown components of the system [98]. We consider a set of partial differential equations with partially unknown physics represented. The corresponding PDE model for a general state ϕ is given by

$$\begin{aligned}
\frac{\partial \phi}{\partial t} &= \mathcal{P}_c\left(\phi, \frac{\partial \phi}{\partial x}, \frac{\partial^2 \phi}{\partial x^2}, ...\right), && \text{in} \quad \Omega \times (0, \infty) \\
&= \mathcal{F}\left(\mathcal{P}_i\left(\phi, \frac{\partial \phi}{\partial x}, \frac{\partial^2 \phi}{\partial x^2}, ...\right), \mathcal{P}_u\left(\phi, \frac{\partial \phi}{\partial x}, \frac{\partial^2 \phi}{\partial x^2}, ...\right)\right), && \text{in} \quad \Omega \times (0, \infty) \qquad (7.1) \\
\mathcal{G}(\phi) &= 0, && \text{in} \quad \partial\Omega \times (0, \infty) \\
\phi &= h, && \text{in} \quad \Omega \times (0)
\end{aligned}$$

where $\mathcal{P}_c$ represents the physical system with complete information. $\mathcal{F}$ represents a potentially simple function that combines the known but incomplete PDE description, $\mathcal{P}_i$, and unknown physics represented by $\mathcal{P}_u$ to match the solutions of complete PDE system. We take $\mathcal{G}$ and h to be known functions appropriately defining the boundary and initial conditions respectively. Ω is the spatial domain over which we solve the PDE system and $\partial\Omega$ its boundary. The term $\mathcal{P}_u$ can take the form of closure terms, source terms, higher order coupling terms between state variables or terms resulting from a set of unknown ODEs/PDEs depending on the physical system under investigation.

A commonly targeted case is when the governing equations of the complete PDE description are computationally too expensive to solve or not known, with turbulence modelling in computational fluid dynamics being a good example [131, 132]. In CFD, spatial filtering is performed on the original governing PDEs in the context of large eddy simulations. This step introduces unclosed terms in the model equations that correspond to unrepresented physics in equation (7.1), due to the effects of the filtered scales. Figure 7.1 (A) shows instances of normalized vorticity for isotropic decaying turbulence. The

solutions of a fully resolved direct numerical simulation (DNS) could be achieved by increasing the spatial resolution of LES. This is a widely studied problem, where the use of deep learning models is currently being explored [36, 73, 28, 7]. Several of these methods train a neural network with LES as the incomplete system to model the effects of the filtered scales and obtain the solutions of the complete DNS [28, 7]. Instead, we consider a different, more challenging problem related to multi-physics, coupled systems containing dependent variables which describe different physical phenomena. In equation (7.1), $\mathcal{P}_u$ contains all the terms corresponding to different physical phenomena that are not included in the PDEs represented by $\mathcal{P}_i$. This formulation could describe fluid-structure interactions which couple fluid mechanics with structural mechanics with $\mathcal{P}_u$ representing the unknown coupling terms, or aeroacoustics problems which couple fluid dynamics and acoustics [133, 134].

However, in this work, we will focus on reactive flow simulation as the complete PDE description, while a non-reactive flow simulation will represent the incomplete PDE basis. It collects the chemical kinetic mechanisms in the unknown physics term of equation (7.1). In this system, $\mathcal{P}_u$ could take the form of the source terms from unknown chemistry in the Navier-Stokes equations. A *reactive flow* refers to a fluid flow with chemical reactions occurring within a reacting fluid, such as combustion-related flows. A reacting fluid is a mixture of two or more species such as hydrocarbons, oxygen, water, carbon dioxide, etc which undergo chemical reactions [17]. In contrast, a non-reactive flow refers to a fluid flow where no chemical reactions take place. Figure 7.1 (B) shows a visual example of a non-reactive and reactive flow simulations which can be considered as an incomplete and complete PDE systems, respectively. Starting from the same initial condition, the influence of $\mathcal{P}_u$ in this multi-physics system leads to fundamentally different solutions as reacting fluid (shown by blue colour in figure 7.1(B)) advances through the domain without reacting for the non-reactive flow simulation or forms a Λ shaped flame with higher temperature of burned products (shown by dark red colour) for the reactive flow simulation. Therefore, we are targeting a more challenging problem than those tackled in [36, 28, 7, 73], where increasing spatial resolution and/or reducing time-scales of the incomplete PDE solver does not lead to a converged full solution. Rather, the complete and incomplete PDEs produce drastically different solutions due to the unknown physics. The central learning objective is to correct this behaviour and retrieve the evolution that would be obtained with the complete PDE description.

Our work expands on the combination of incomplete PDE solvers and neural networks (NNs) [98, 99] to account for the effects of an incomplete physics model. The NN aims to complete the PDE description, where the differences in complete and incomplete PDE solutions are beyond the effects of spatial and temporal scales. We showcase that combining the trained NN model with a differentiable solver for the incomplete PDE can accurately reproduce the physical solutions of the complete, multi-physics PDE solver with stable long-term rollouts.

Reactive flow modelling has applications in numerous domains such as combustion processes in gas turbines [135, 136], climate modelling [137, 138] and astrophysics simulations [139]. Resolving the Navier-Stokes equations lies at the core of these problems, where additionally the transport of different species of relevance must also be accounted

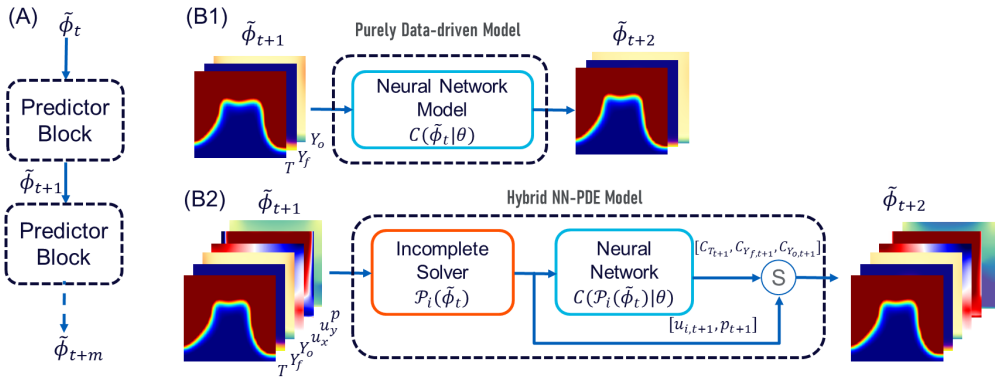


Figure 7.2: (A) Multi-step training framework helps to learn the dynamics of complete PDE solver over longer rollouts. (B1) Details of the input flow state and predictor block used in a purely data-driven approach and (B2) the hybrid NN-PDE approach, where S denotes the concatenation of different fields to obtain the complete flow state $\tilde{\phi}$ at next time step

for, together with their production or consumption often following complex reaction mechanisms [17]. For chemically reacting flows, generation or consumption of multiple species via some chemical reactions are modelled using a net source term. It is a well-known fact that the incorporation of a detailed chemical kinetic mechanism in a reacting flow model can result in a stiff system of governing equations [140, 141, 142].

We showcase the effectiveness of our approach for different cases of planar 2D premixed methane-air flames, and the varying transient evolution of Bunsen-type flames. We show that the proposed approach can handle large domains with highly resolved flames, which are closer to the practical flame domains used in many industrial applications. Specifically, we concentrate on training a NN model to correct the spatio-temporal effects of energy and species transport source terms. We show that in addition to recovering the desired solutions, this approach overcomes inherent problems of temporal stiffness due to the complex reaction mechanism. Lastly, we show that the resulting hybrid solver provides a flexible building block for adjacent tasks. Specifically, we demonstrate this for controlling the evolution of flame shapes via continuous control.

7.2 Methodology

We consider two different sets of PDEs with their associated numerical solver, which we denote as the *incomplete* PDE $\mathcal{P}_i$ and the *complete* PDE $\mathcal{P}_c$. By evaluating $\mathcal{P}_c$ on an input state ϕ_t at time t, we can compute the points of the phase space sequences; $\phi_{t+\Delta t} = \mathcal{P}_c(\phi_t)$. Without loss of generality, we assume a fixed time-step Δt and denote a state $\phi_{t+\Delta t}$ at next time instance as ϕ_{t+1}. Let $\mathcal{X}$ be a Banach space of functions taking values in a spatial domain $\Omega \subset \mathbb{R}^2$. Furthermore, let $C^\dagger : \mathcal{X} \to \mathcal{X}$ be a nonlinear map from ϕ_t to ϕ_{t+1}, defined over a finite time interval $[0, T]$ and an input flow state ϕ. The

learning objective is to find the best possible correction function

$$C(\phi;\theta) : \mathcal{X} \rightarrow \mathcal{X}, \quad \theta \in \Theta \tag{7.2}$$

for some finite dimensional parameter space Θ by choosing $\theta^{\dagger}$ such that $C(.;\theta^{\dagger}) \approx C^{\dagger}$. The neural network models can be employed to learn such mapping as shown in figure 7.2. The model parameters θ are estimated from the complete PDE solution trajectories $(\phi_0, \phi_1, .., \phi_T)$. The learned predictions obtained after repeatedly applying the corrector C and invoking $\mathcal{P}_i$ are denoted by $(\tilde{\phi_0}, \tilde{\phi_1}, .., \tilde{\phi_T})$.

7.2.1 Partial differential equations for reactive flows

In this section, we present the basics of reactive flow simulation and underlying conservation equations. A reactive flow involves multiple species reacting through one or more chemical reactions. Different species, such as hydrocarbons (fuel), oxygen (oxidizer), water, carbon dioxide (products), are characterized through their mass fraction Y_k. The primary variables for two-dimensional (2-D) reacting flow involve density (ρ), 2-D velocity field (u), temperature (T), and the mass fraction Y_k of the N reacting species.

In a premixed combustor, fuel (Y_F) and oxidizer (Y_O) are mixed before they enter the combustion chamber. The computation of premixed flames with complex chemistry is possible but we consider a simplified approach. We assume that chemistry proceeds only through one irreversible reaction i.e. one-step chemistry. If υ'_F and υ'_O are the coefficients corresponding to fuel and oxidizer when considering a one-step reaction of type $\upsilon'_F F + \upsilon'_O O \rightarrow$ products, the mass fractions of fuel and oxidizer correspond to stoichiometric conditions. It is defined as,

$$\left(\frac{Y_O}{Y_F}\right)_{st} = \frac{\upsilon'_O W_O}{\upsilon'_F W_F} = \varphi. \tag{7.3}$$

This ratio φ is called the mass stoichiometric ratio. W_F and W_O represent the molecular weights of the fuel and oxidizer, respectively. The equivalence ratio of a given mixture is then,

$$E = \varphi \left(\frac{Y_F}{Y_O}\right). \tag{7.4}$$

A common example of such reaction is $CH_4 + 2O_2 \rightarrow CO_2 + 2H_2O$, where $\upsilon'_F = 1, \upsilon'_O = 2, W_F = 0.016$ kg/mole$, W_O = 0.032$ kg/mole and therefore $\varphi = 4$. The equivalence ratio is an important parameter in the design of a premixed combustion system. Rich combustion is observed for $E > 1$ (the fuel is in excess) and lean regimes are achieved when $(E < 1)$ (the oxidizer is in excess) [17]. Most practical premixed combustors operate at or below stoichiometry [13].

The physical system we investigate is a laminar premixed methane-air flame with one-step chemistry. It is governed by the following Navier-Stokes equations [17]

$$
\begin{aligned}
\frac{\partial \rho}{\partial t} + \nabla.(\rho u) &= 0 && \text{in} \quad \Omega \times [0,T] \\
\frac{\partial}{\partial t}(\rho u) + \nabla.(\rho u \otimes u) &= -\nabla p + \nabla.\tau && \text{in} \quad \Omega \times [0,T] \\
\rho C_p \left(\frac{\partial T}{\partial t} + \nabla.(u \otimes T) \right) &= \dot{\omega}'_T + \lambda \nabla^2 T && \text{in} \quad \Omega \times [0,T] \\
\frac{\partial \rho Y_k}{\partial t} + \nabla.(\rho u \otimes Y_k) &= \dot{\omega}_k + \rho D_k \nabla^2 Y_k && \text{in} \quad \Omega \times [0,T]
\end{aligned}
\tag{7.5}
$$

where τ, D_k, λ are the strain rate tensor, the diffusion coefficient of species k, and the mixture thermal conductivity. In addition, C_p denotes the mixture specific heat capacity. $\dot{\omega}_k$ and $\dot{\omega}'_T$ are the species reaction rate and heat release rate, respectively. Boundary conditions of each flow state vary depending on the applications and are provided in detail in section 7.3. $\Omega \subset \mathbb{R}^2$ represents the spatial domain.

The reaction rate $\dot{\omega}_k$ for each species are linked to the progress rate Q_1 at which the single reaction proceeds, as: $\dot{\omega}_k = W_k \upsilon_k Q_1$. The simplifications proposed by Williams, 1985 [38] and Mitani, 1980 [143] are used to model the reaction rates $\dot{\omega}_k$ for each species. The progress rate Q_1 is assumed to have the Arrhenius form and is given by,

$$
\begin{aligned}
Q_1 &= B_1 T^{\beta_1} \left(\frac{\rho Y_F}{W_F} \right)^{n_F} \left(\frac{\rho Y_O}{W_O} \right)^{n_O} \exp\left(-\frac{E_a}{RT} \right). \\
\dot{\omega}_F &= \upsilon'_F W_F Q_1; \quad \dot{\omega}_O = \upsilon'_O W_O Q_1
\end{aligned}
\tag{7.6}
$$

where $\dot{\omega}_F$ and $\dot{\omega}_O$ are the reaction rates of fuel and oxidizer, respectively. $\dot{\omega}'_T$ is the heat release due to combustion and is formulated as, $\dot{\omega}'_T = -\sum_{k=1}^{N} \Delta h^o_{f,k} \dot{\omega}_k$. The formation enthalpy $h^o_{f,k}$ is the enthalpy needed to form 1 kg of species k at the reference temperature $T_0 = 298.15K$. The formulation for $\dot{\omega}'_T$ can be further simplified as,

$$
\dot{\omega}'_T = -\sum_{k=1}^{N} \Delta h^o_{f,k} W_k \upsilon_k Q_1 = -\sum_{k=1}^{N} \Delta h^o_{f,k} \frac{W_k \upsilon_k}{W_F \upsilon_F} W_F \upsilon_F Q_1 = -\sum_{k=1}^{N} \Delta h^o_{f,k} \frac{W_k \upsilon_k}{W_F \upsilon_F} \dot{\omega}_F = -Q \dot{\omega}_F.
\tag{7.7}
$$

Therefore, the heat release source term $\dot{\omega}'_T$ and the fuel source term $\dot{\omega}_F$ are linked by the Q, which is the heat of reaction per unit mass. Following [17], parameters corresponding to a real-world methane-air flame are chosen as: $B_1 = 1.0810^7$ uSI; $\beta_1 = 0$; $E_a =$ 83600 J/mole; $n_F = 1$; $n_O = 0.5$; $Q = 50100$ kJ/kg; $C_p = 1450$ J/(kgK). Taken together, the system of equations above is a challenging scenario even for classical solvers, and due to its practical relevance likewise a highly interesting environment for deep learning methods.

7.2.2 Problem formulation

The incomplete PDE solver solves the set of equations (7.5) without the source terms and reaction rates $\dot{\omega}_k$ and $\dot{\omega}'_T$, while the complete PDE solver solves the full set from equation (7.5). The neural network model, denoted by $C(\mathcal{P}_i(\phi)|\theta)$, corrects the incomplete/non-reacting flow states $\mathcal{P}_i(\phi)$ to obtain the complete/reacting states $\mathcal{P}_c(\phi)$ as shown in

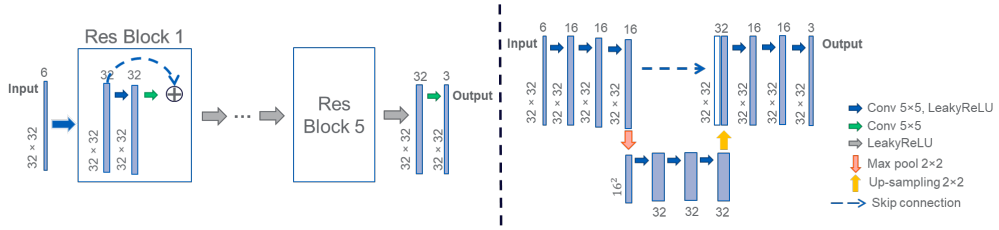


Figure 7.3: Schematic of the convolutional neural networks used. Left: ResNet with 5 Res-Blocks, Right: UNet32 with 2 layers.

figure 7.2 (B2). The neural network is trained to model the effects of the unknown chemistry using parameters θ given an input flow state, $\phi = [u_i, p, T, Y_f, Y_o]$. As seen from equation (7.5), and equation (7.6), the temperature and species mass fraction fields are strongly coupled which significantly increases the prediction problem complexity. A slight error in one of the fields will quickly propagate into the other fields, making the predictions diverge. In the following, a subscript $C_s(\circ)$ will denote that the neural network C generates the field s, e.g., C_T generating the temperature field. The update can be written as, $\tilde{\phi}_{t+1} = [u_{i,t+1}, p_{t+1}, C_T(\mathcal{P}_i(\tilde{\phi}_t)|\theta), C_{Y_f}(\mathcal{P}_i(\tilde{\phi}_t)|\theta), C_{Y_o}(\mathcal{P}_i(\tilde{\phi}_t)|\theta)]$ where $\tilde{\cdot}$ indicates a corrected state. $u_{i,t+1}$ and p_{t+1} are the time-advanced velocity and pressure field, respectively, predicted using the incomplete PDE solver.

7.2.3 Training Methodology

Hybrid NN-PDE approach We employ a hybrid NN-PDE approach that augments a neural network model with a PDE solver [27, 28]. In contrast to previous work, we use the incomplete PDE solver as a basis, and hence the solver does not converge to the desired solutions under refinement, as explained in section 7.1. The neural network is integrated and trained in a loop with the incomplete PDE solver using stochastic gradient descent for m time steps, as shown in figure 7.2 (A,B2). Here, the number of temporal look-ahead steps, m, is an important hyper-parameter of the training process. Higher m provides the network with longer-term feedback at training time through the gradient roll-outs. This gives the model improved feedback on how the time dynamics of the incomplete PDE solver affect the input states, and hence which corrections need to be inferred by the model. In our framework, if an incomplete PDE solver is very close to the complete PDE solver for the given data, the NN model would learn a correction which is closer to an identity. The skip connections in the used convolutional neural network architectures would readily enable such an identity operation. In contrast, when the incomplete solver contributes very little, the proposed approach would start to represent a purely data-driven approach.

Differentiable PDE Solver A central component of the hybrid NN-PDE model is the differentiable solver, which allows us to embed the solver for the incomplete PDE system in the training of a neural network. The differentiable solver acts as additional non-trainable layers in the network. They provide derivatives of the outputs of the simulation with respect to its inputs and parameters. Finite differences can be used to

compute the gradients of the PDE solver, but they are computationally very expensive for high-dimensional PDEs. Differentiable solvers resolve this issue by solving the adjoint problem [144] via analytic derivatives. Here, we use the differentiable PDE solver from the *phiflow* framework of [84] in combination with *Tensorflow* to obtain the non-reactive flow solver and reactive flow solver solutions. The marker-and-cell method [145] is adopted to represent temperature, pressure, density, species mass fraction fields in a centered grid, and velocities in a staggered grid. Basic differential operators such as gradient, divergence, curl, and Laplace operators are implemented in TensorFlow using basic mathematical tensor operations [96]. These differential operators act on a 9 point stencil of grid points and the corresponding derivatives are straight-forward to compute. Advection is implemented with a semi-Lagrangian step, while the Poisson problem for the pressure and its gradient are solved implicitly. Efficient derivatives for all these operations are then combined via back-propagation [146].

Purely data-driven approach A PDD model is used as a baseline. It employs a neural network model to learn the complete flow states $\mathcal{P}_c(\phi)$ given an input flow state ϕ^S where $\phi^S = [T, Y_f, Y_o]$, as shown in figure 7.2 (B1). The new state predicted by the trained neural network model is $\tilde{\phi}^S_{t+1} = C(\tilde{\phi}^S_t|\theta)$. For the hybrid NN-PDE as well as PDD, the neural network part of the predictor block in figure 7.2 consists of a fully convolutional neural network model.

We additionally compare against the *Fourier neural operator* (FNO) of Li et al. [78] as an example of a state-of-the-art neural operator method. The new state predicted by the trained model is given by $\tilde{\phi}^S_{t+1} = C(\tilde{\phi}^S_t|\theta)$ where $\tilde{\phi}^S = [T, Y_f, Y_o]$ and $C(\circ)$ represents the Fourier neural operator.

7.2.4 Training Details

All three approaches use an L_2 based loss that is evaluated for m steps as

$$\mathcal{L}(\theta) = \sum_{n=t+1}^{t+m} \sum_{\phi=\{T,Y_f,Y_o\}} ||\tilde{\phi}_n, \mathcal{P}_c(\phi_n)||_2 \ . \tag{7.8}$$

The network receives the input states as shown in figure 7.2. The output of the neural network model is used to obtain the corrected state $\tilde{\phi}_{t+1}$ as specified above. We constrain the mass fraction fields Y_k to contain physical values in the range $Y_f \in [0, 0.05]$ and $Y_o \in [0, 1]$. All models are trained for 100 epochs with a batch size of 3 and a learning rate of 0.0001. We have used a small batch size of 3 due to the memory requirements as we unroll the NN-PDE framework over long timesteps. Our training procedure uses the Adam optimizer [116]. For all our computations, a *Nvidia Quadro RTX 8000* GPU is used. Figure 7.4 shows the typical training loss curve over 100 epochs.

For PDD and the hybrid NN-PDE approach, we experimented with both the ResNet [147] and the UNet [148] architectures. Figure 7.3 shows the schematic of the neural network models used. We found that the ResNet performed the best for the PDD setting, while for the hybrid NN-PDE approach the UNet performed consistently better. Hence,

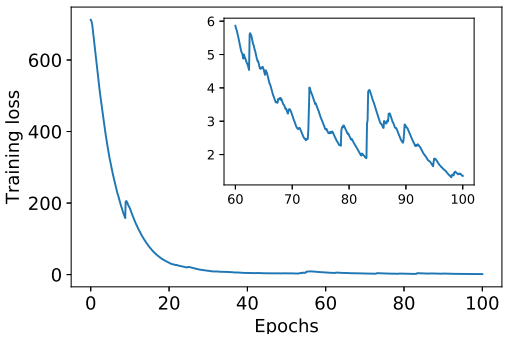


Figure 7.4: Typical L_2 based training loss as defined in equation (7.8) over 100 epochs. The inset figure shows zoomed in loss function curve for last 40 epochs.

the following results will use a ResNet for PDD models, and a UNet for the NN-PDE hybrids.

Table 7.1: Details of various neural network architectures used in this paper.

Hyper-parameters	ResNet	UNet32	UNet100
Kernel size	5	5	5
Latent size	32		
Activation	LeakyReLu	LeakyReLu	ReLu
Loss	MSE	MSE	MSE
# ResBlocks	5		
# UNet layers		2	3
CNN stack depth	2	3	2
Base latent size		16	16
Spatial down-sample by layer		2	2
latent sizes		(32)	(32,64)
# trainable parameters	261,953	136,227	362,371

Table 7.2: Mean and standard deviation of errors over different architectures for purely data-driven and hybrid NN-PDE approaches.

		PDD ResNet	PDD UNet	Hybrid NN-PDE ResNet	Hybrid NN-PDE UNet
MAPE	Planar-v0	6.33 ± 3.05%	7.09 ± 4.40%	1.62 ± 0.44%	**1.40** ± 0.65%
	uniform-Bunsen	7.58 ± 3.73%	2.87 ± 1.53%	2.01 ± 0.99%	**0.72** ± 0.37%
	nonUniform-Bunsen32	12.48 ± 11.31%	19.19 ± 14.92%	3.25 ± 2.35%	**2.04** ± 1.39%
	nonUniform-Bunsen100	-	-	15.68 ± 10.44%	**3.23** ± 3.76%

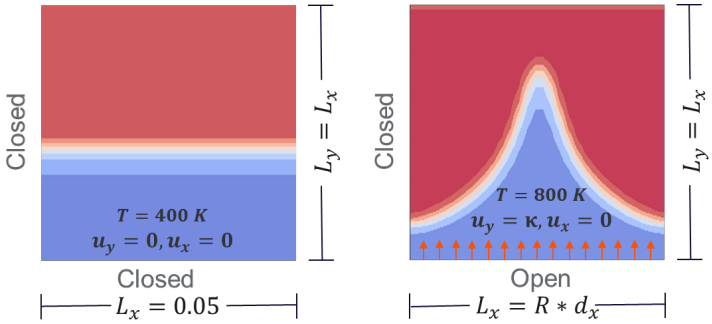


Figure 7.5: Details of the boundary condition for - left: Planar flame; right: a Bunsen-type flame. R represents resolution of the domain

Table 7.3: Details of the boundary conditions used for the planar flame case and various cases of Bunsen-type flames. The uniform-Bunsen case is obtained with κ = constant and nonUniform-Bunsen case is obtained with $\kappa \in \mathbb{R}^n$ as discussed in section 7.3.

Planar flame		
field	inlet	left and right wall
T	400 K	Neumann BC
u_y	0	-
u_x	0	-
p	101325	Neumann BC
Y_f	$\frac{1}{1+\frac{\varphi}{E}(1+3.76\frac{W_{N_2}}{W_{O_2}})}$	slip

Bunsen-type flame		
field	inlet	left and right wall
T	800 K	Neumann BC
u_y	κ	no-slip
u_x	0	slip
p	101325	Neumann BC
Y_f	$\frac{1}{1+\frac{\varphi}{E}(1+3.76\frac{W_{N_2}}{W_{O_2}})}$	slip

Table 7.1 provides details of the convolutional blocks, layers of the network, and activation function. The *UNet32* architecture with 2 layers is used for 32 × 32 cases and the *UNet100* architecture with 3 layers is used for 100 × 100 case discussed in section 7.3. Table 7.2 shows the comparison between MAPE achieved by ResNet and UNet architectures for the purely data-driven and the hybrid NN-PDE approach for different cases considered in this study. For the hybrid NN-PDE approach with 32 × 32 resolution cases, the ResNet and UNet achieve comparable performance with the UNet yielding the lowest error. The main difference arises in the 100 × 100 resolution case. The ResNet fails to converge during training, leading to high test error. On the contrary, the UNet achieves low training and testing errors. The main advantage of UNet comes from transforming high resolution input to low resolution representation and reconstructing it back to the high resolution output. Therefore, we choose the UNet architecture for the hybrid NN-PDE approach. For the purely data-driven approach the ResNet was found to be better for 2 out of 3 cases. Specifically for the nonUniform-Bunsen32 case, UNet performs very poorly for the PDD approach. Therefore, we report ResNet errors for the PDD approach throughout the study.

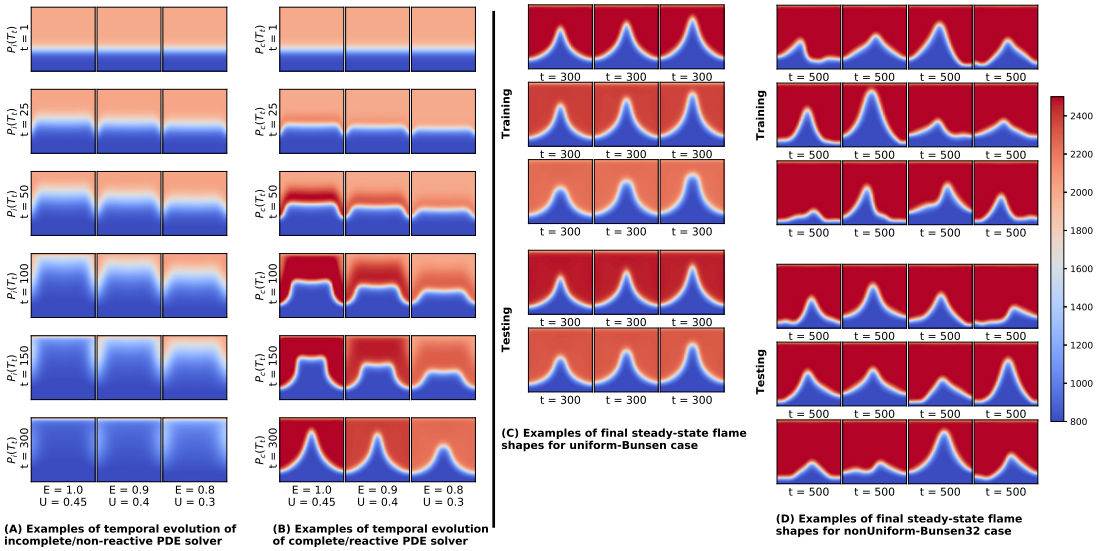


Figure 7.6: Instances of (A): incomplete PDE solver; (B): complete PDE solver for different operating conditions. (C): Snapshot of complete PDE solver at $t = 300$ for all training and testing datasets of uniform-Bunsen case. (D): Snapshot of complete PDE solver at $t = 500$ for all training and testing datasets of nonUniform-Bunsen32 case

7.3 Numerical Experiments

We consider a case of planar 2D premixed methane-air flame propagating in a quiescent mixture (**Planar-v0**) and two cases of the transient evolution of an initially planar laminar premixed flame into a Bunsen-type flame under different inlet velocity conditions (**uniform-Bunsen, nonUniform-Bunsen**). We obtain the target data by considering the source terms as defined by equation (7.6) with the parameters mentioned in section 7.2.1.

Planar-v0 For the most basic case, the planar 2D flame model setup, we consider the reacting Navier-Stokes equations described in section 7.2 with zero inlet velocity i.e. $u_x = 0$, $u_y = 0$. We consider a square domain of size 0.05 m $\times$ 0.05 m with 32×32 resolution and closed boundary conditions, as shown in figure 7.5. The simulation is initialized using a steep transition between a premixed methane-air mixture and burnt gases. Our training data consists of 6 simulations of 300 steps created by varying the equivalence ratio E. It represents the stoichiometric mixture (φ) of fuel Y_f and oxidizer Y_o mass fractions, i.e., $E = \varphi \frac{Y_f}{Y_o}$ and thus fundamentally influences the dynamics of the chemical reaction. For the training data we use $E_{\text{train}} = \{1.0, 0.9, 0.8, 0.7, 0.6\}$, while the test dataset contains $E_{\text{test}} = \{0.95, 0.85, 0.75, 0.65\}$.

uniform-Bunsen In contrast to the planar case, the premixed methane-air mixture is now fed with a constant inlet velocity. The boundary conditions upstream, at $y = 0$, are $(u_x, u_y)_{x,y=0} = (0, \kappa)$ where κ is a n-dimensional vector with constant ampli-

tude. The target simulation contains a heat release rate term, and in this case, the initial temperature field evolves into different Λ-shaped flames at the end of the 300^{th} time step. Training and testing datasets are created by varying the equivalence ratio and inlet velocity amplitude U: $E_{\text{train}} = \{1.0, 0.9, 0.8\}$ and $U_{\text{train}} = \{0.45, 0.4, 0.3\}$. The test dataset uses $E_{\text{test}} = \{0.95, 0.85\}$ $U_{\text{test}} = \{0.43, 0.375, 0.325\}$. The length and temperature of the flame significantly vary depending on the inlet velocity and equivalence ratio provided. Figure 7.6 (A) shows the temperature field evolution of the incomplete PDE solver at different time instances for 3 different operating conditions for the uniform-Bunsen case. Figure 7.6 (B) shows the corresponding target training data for the uniform-Bunsen case. All the simulations are run for 300 steps. It shows the difference between non-reactive and reactive flow solver simulations for different operating conditions over 300 time-steps. Figure 7.6 (C) shows the last snapshot ($t = 300$) of 9 training simulations and 6 testing datasets with operating parameters interpolated between the training data parameters for uniform-Bunsen case. These datasets are obtained by varying the equivalence ratio and the magnitude of the uniform inlet velocity. The flame temperature depends on the equivalence ratio used and the flame height depends on the inlet velocity amplitude and equivalence ratio.

nonUniform-Bunsen As a third case we consider the transient evolution of a premixed methane-air flame with non-uniform inlet velocity. The boundary conditions upstream, at y = 0, are $(u_x, u_y)_{x,y=0} = (0, \kappa)$ where κ is a n-dimensional vector whose elements are each sampled from a uniform distribution from [0.2, 0.65]. We experiment with two different domain sizes, 32 × 32 (**nonUniform-Bunsen32**) and 100 × 100 (**nonUniform-Bunsen100**) . The larger domain size used is closer to the practical reactive flow domain utilized in CFD applications [4] with a highly resolved flame. These inlet velocity conditions generate complex flame shapes, which increase the difficulty of the prediction problem. We consider simulation sequences with 500 steps, as it takes longer time for the flame to reach the steady-state solution. Figure 7.6 (D) showcases the snapshots of training and testing dataset at $t = 500$ for nonUniform-Bunsen32 case. Non-uniform variations in inlet velocity profile leads to different complex flame shapes. We use 12 datasets with 500 simulation steps to train the models and test it on 12 test cases shown in figure 7.6 (D). For the nonUniform-Bunsen32 case with 32 unrolling steps ($m = 32$), training requires approximately 60 hours with approximately 1 GB of GPU memory.

The training and test datasets are split using different initial conditions. This ensures that varied (training) and unseen (test) data is provided. For example, in the case of Planar-v0 scenario, the training data consists of 5 simulations of planar flame with different equivalence ratios, evolving over 300 simulation steps. During test time, only the initial state of the flow field (ϕ_0) and unseen boundary conditions (i.e. other equivalence ratios than those in the training datasets) are specified and the trained hybrid NN-PDE model is used to make predictions over 300 time steps. A similar setup is used for the baseline approaches. Multi-step training framework shown in figure 7.2 demonstrates the training setup for the baseline and hybrid NN-PDE approaches. Continuous time-slices are used during training. As shown in figure 7.2 (A), during training, the *Predictor Block* is unrolled for m steps i.e. the network is trained using multiple "se-

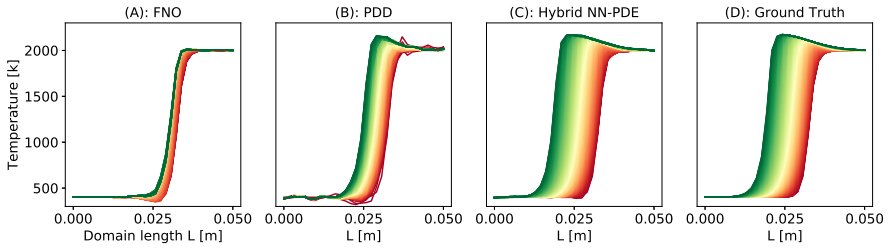


Figure 7.7: 1D cut of the planar flame simulation over 300 steps. The initial state is plotted in red, target state in green. Hybrid NN-PDE approach predicts physically accurate results over longer rollouts.

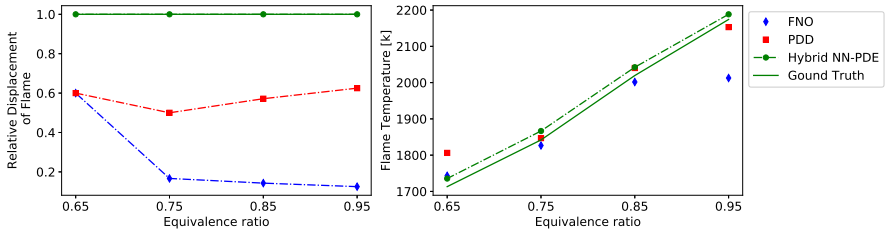


Figure 7.8: Hybrid NN-PDE approach predicts physically accurate results with correct flame temperature and relative displacement of flame front across different equivalence ratios

quences" of m consecutive snapshots (selected from the cases in the training datasets). During testing, only ϕ_0 (the initial flow state) along with unseen boundary conditions are provided and the trained predictor block is evaluated 300 times to obtain predictions from $t = 1, 2 \ldots, 300$.

7.4 Results

We demonstrate the capabilities of the proposed learning approach to represent the complete PDE description with the aforementioned cases of increasing difficulty. We also study its ability to generalize to unseen operating conditions such as equivalence ratios, simultaneous variations in constant inlet velocity and equivalence ratio, and non-uniform inlet velocity profiles. As baselines, we compare against a purely data-driven approach; a neural network model with exactly the same look-ahead steps, and the Fourier neural operator of [78] that likewise includes m look-ahead steps as discussed in section 7.2.3. All the qualitative and quantitative evaluations shown in the paper are performed on test datasets with unseen initial conditions.

7.4.1 Planar-v0

Table 7.4 compares the mean absolute percentage errors (MAPE) and mean squared errors of the temperature field for all the cases discussed in section 7.3. For Planar-v0,

Table 7.4: Mean and standard deviation of errors over all time steps of all testsets. The Hybrid NN-PDE approach outperforms all other baselines considered.

		Baseline FNO	Baseline NN	Hybrid NN-PDE	Hybrid NN-PDE-dt
MAPE	Planar-v0	8.27 ± 5.51%	6.33 ± 3.05%	1.40 ± 0.65%	**1.21** ± 0.39%
	uniform-Bunsen	15.57 ± 8.72%	7.58 ± 3.73%	**0.72** ± 0.37%	1.11 ± 0.47%
	nonUniform-Bunsen32	12.30 ± 7.98%	12.48 ± 11.31%	**2.04** ± 1.39%	2.46 ± 1.61%
	nonUniform-Bunsen100	-	-	**3.23** ± 3.76%	4.14 ± 4.99%
MSE	Planar-v0	88316 ± 124394	34491 ± 43966	1122 ± 1300	**292** ± 296
	uniform-Bunsen	220817 ± 167024	72563 ± 56919	**721** ± 1007	1267 ± 1622
	nonUniform-Bunsen32	184860 ± 185694	130219 ± 142094	**9647** ± 13903	17569 ± 24392
	nonUniform-Bunsen100	-	-	**23293** ± 37451	26862 ± 47639

the FNO and PDD approaches yield large errors with a MAPE of 8.27% and 6.33%, respectively. On the other hand, the hybrid NN-PDE model trained with 32 look-ahead steps reduces the error to 1.4%, and thus performs significantly better than the two baselines. This behaviour is visualized in figure 7.7 with a 1D transverse cut of the simulation domain over 300 time-steps. The hybrid NN-PDE approach successfully captures the propagation of the flame.

In figure 7.8, we also compare two important physical quantities: the flame temperature and relative displacement of the flame front, across different equivalence ratios. It can be seen that the hybrid NN-PDE model (green circles) accurately predicts the flame temperature for different test cases (solid green line). The relative displacement of the flame front is computed as $|\tilde{x}_t - \tilde{x}_0|/|x_t - x_0|$, where x_t is the position along the flame normal at time t on the $1200K$ isotherm of the ground truth simulation, and $\tilde{x}$ denotes the predicted position. For all test cases, the hybrid approach accurately predicts the flame displacement over $t = 300$ steps, while the other approaches yield significant errors. Figure 7.9(A) shows the 2D visualization of the temperature field predictions for Planar-v0 case. As seen from the ground truth images, the methane-air mixture (black colour) converts into the burned products (yellow colour) due the chemical reaction at the flat flame surface (red colour). The dotted, horizontal red line helps to compare the transition of the flame interface, i.e. the displacement of the flame front. Due to the chemical reaction, the fuel-air mixture is consumed and turns into the burned product as the simulation progresses. The FNO approach completely fails to predict the propagation of the flame for the given test case. Its output does not show any evolution from the initial temperature profile for the given operating condition. The purely data-driven approach fails to capture the flame front displacement correctly, thus leading to an inaccurate prediction with large errors. The hybrid NN-PDE model accurately captures this evolution of planar methane-air flame in a quiescent mixture. Figure 7.9(B) shows the instantaneous MAPE w.r.t. ground truth data for predictions shown in figure 7.9(A). The absolute error shown in figure 7.9(B) exceeds 1100 K for the FNO and

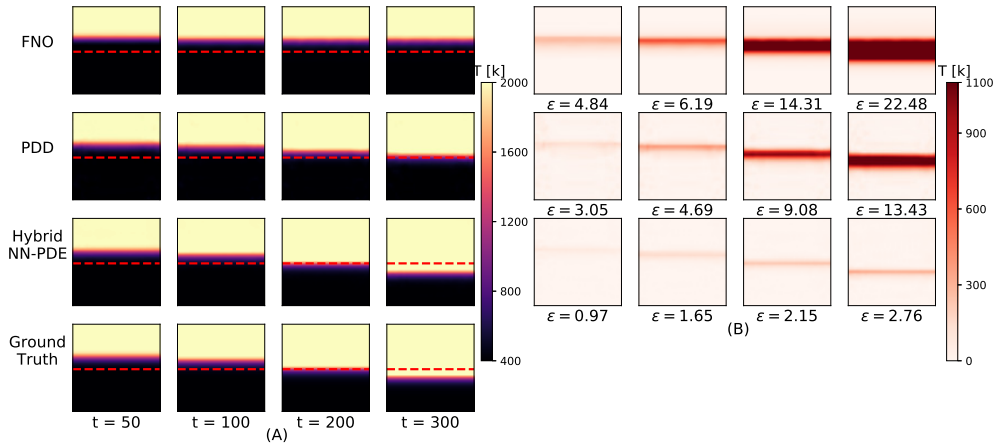


Figure 7.9: Planar-v0 flame case with $E = 0.95$. (A) Temperature field predictions; (B) Absolute error between ground truth ($\mathcal{P}_c(T_t)$) and the output predicted - from top to bottom top - by: FNO, purely data-driven approach and hybrid NN-PDE. The numbers represent instantaneous MAPE

purely data-driven approaches as these do not predict the flame temperature and flame front displacement correctly. We use the upper limit of 1100 K for colourbar to highlight the errors in the hybrid NN-PDE approach more clearly. Large errors in FNO and PDD results stem from their inability to reliably predict the flame temperature and flame front displacement.

7.4.2 Bunsen-type flame

For the uniform-Bunsen case, the PDD baseline with an error of 7.58% performs better than the FNO model which yields an error of 15.57%. However, a neural network model combined with an incomplete PDE solver for 32 look-ahead steps yields a significantly lower error of 0.72% as shown in table 7.4. This means that the hybrid approach reduces the errors by a factor of 10 over the baselines considered. Figure 7.10(A) shows the temporal evolution of the hybrid NN-PDE approach predicted over 300 time steps for the uniform-Bunsen case. It shows the results for a test case with $U = 0.4375$ and $E = 0.95$. It predicts a symmetric flame with accurate flame height and achieves very low instantaneous MAPE of 0.73% at t=300, when compared with the ground truth data.

Next, we study a complex scenario of nonUniform-Bunsen flame with 32×32 resolution. The hybrid NN-PDE approach outperforms the PDD approach and FNO with an improvement of $\sim 80\%$. We also include a variant of the PDD approach (PDD-5) which is trained to predict all quantities ($\phi^S = [T, Y_f, Y_o, u, p]$) of the flow state. It employs a neural network model to learn the complete flow states $\mathcal{P}_c(\phi)$ given an input flow state ϕ^S where $\phi^S = [T, Y_f, Y_o, u, p]$. Keeping all hyperparameters the same, the ResNet is trained to predict all quantities of flow state for nonUniform-Bunsen32 case. As the PDD approach does not enforce any physical laws, the inclusion of additional quantities

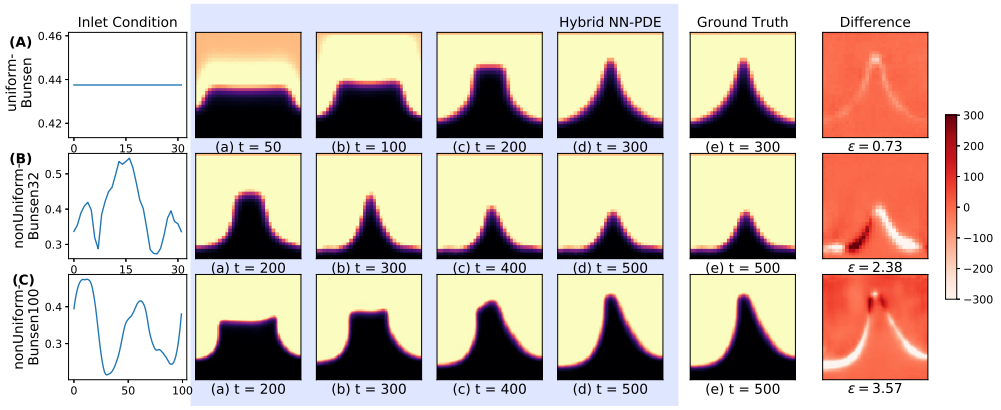


Figure 7.10: Left : Inlet velocity profile used. (a-d) Temperature field prediction by hybrid NN-PDE approach over different steps given the inlet velocity profile. (e) Ground truth data. Right : Difference between ground truth data and hybrid NN-PDE output at last snapshot. Top to bottom: 32×32 resolution cases of (A) uniform-Bunsen, (B) nonUniform-Bunsen32, and (C) 100×100 testcase nonUniform-Bunsen100. ϵ represents the instantaneous MAPE

results in a more difficult learning task, which given the same number of learnable parameters naturally results in a deteriorated quality of the inferred results. Figure 7.11 shows the MAPE of the temperature field predicted by this new variant (referred to as PDD-5) with the PDD approach described in section 7.2.3 and hybrid NN-PDE approach for the nonUniform-Bunsen32 case. Across all test cases, PDD-5 performs worse as problem complexity has increased.

The hybrid NN-PDE model achieves a low MAPE of 2.04% compared to the higher errors of 23.45% and 12.48% predicted by the PDD-5 and the PDD approach, respectively. The MSE values show this trend even more clearly. The large standard deviations of the MSE numbers indicate that the predictions made by the baseline approaches contain substantial deviations from the target values. It is important to highlight that despite using the same training data and a similar neural network architecture with the same number of look-ahead steps, the hybrid approach outperforms the PDD approach due to its learned collaboration with the incomplete PDE solver. It accurately reproduces the complete PDE behaviour. Figure 7.10(B) highlights this with a visualization of the hybrid NN-PDE model predictions for nonUniform-Bunsen32 case. The trained model accurately predicts the flame simulation over long roll-outs of 500 steps and achieves the complex flame shape with a low, instantaneous MAPE of 2.38%.

Finally, we showcase the ability of the hybrid approach to predict the temporal evolution of highly resolved flames with the uniform-Bunsen100 scenario. Despite the increased complexity of the larger resolution, it achieves a very good overall MAPE of 3.23% over 12 test cases of 500 simulation steps. Figure 7.10(C) shows an example of physically accurate predictions made by the hybrid NN-PDE model. We omit the evaluation of baselines for high resolution cases as they do not succeed to model the flame dynamics for low resolution cases.

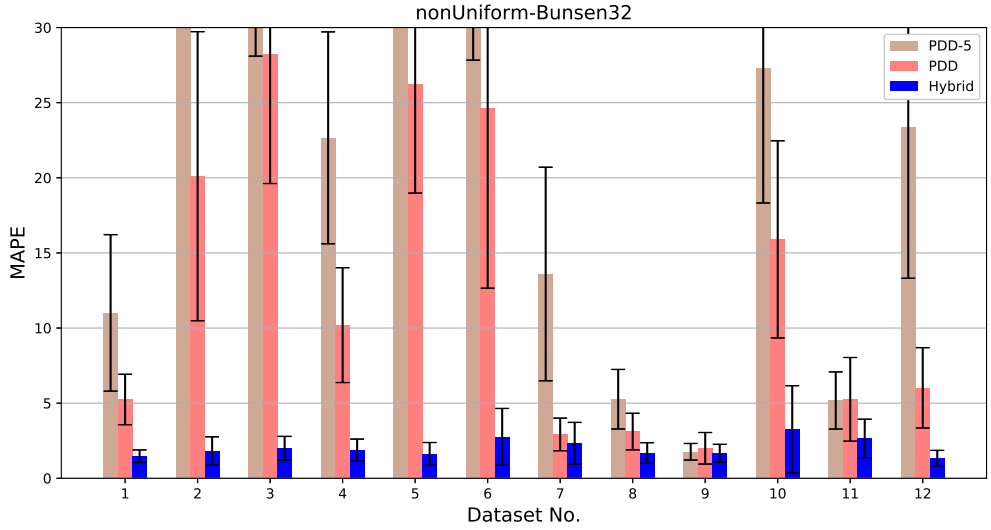


Figure 7.11: Bar plot of MAPE of temperature field predictions by a purely data-driven model trained to predict 5 quantities of flow state (PDD-5), PDD model trained to predict 3 quantities of flow state (PDD), and hybrid NN-PDE model (Hybrid)

In the following subsections, we present additional visualizations and a detailed comparison of the hybrid NN-PDE model against baseline approaches for different scenarios of Bunsen-type flames.

7.4.2.1 uniform-Bunsen

Figures 7.12 show an example of the uniform-Bunsen case: a test case with $u_y = 0.375$ and $E = 0.95$. The constant inlet velocity results in a symmetric, Λ-shaped flame. The vertical, dotted, red line in figure 7.12(A) helps to assess the symmetric nature of the flame. The purely data-driven model predicts a thicker flame ($t = 50$) or a flame with a spurious tip ($t = 172$) or an asymmetric flame ($t = 225$). Snapshots of the hybrid NN-PDE model predictions in figure 7.12(A) show that it adapts to this scenario very well and succeeds in obtaining the correct results for long term forecasts of the temperature field. Furthermore, the flame shape and height are also better predicted, as shown in figure 7.12(A). Very low error levels in figure 7.12(B), such as $\epsilon = 1.26$ at $t = 300$, indicate that the hybrid model recovers the temperature field well.

7.4.2.2 nonUniform-Bunsen

Figure 7.13 compares the temporal predictions made by FNO, PDD and the hybrid NN-PDE model with ground truth data for nonUniform-Bunsen32 case. Compared to Planar-v0 and uniform-Bunsen case, FNO predicts qualitatively better results for this case. Although still highly inaccurate, it predicts shapes that come closer to the target flame shapes for the later part of the simulation (for $t > 300$). This improvement might be due to the larger training dataset used for this scenario compared to the previous

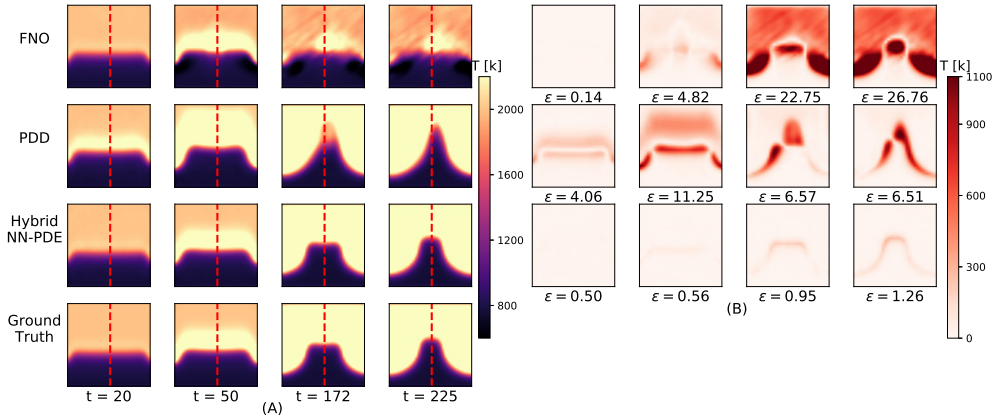


Figure 7.12: uniform-Bunsen case with constant inlet velocity $U = 0.375$ and $E = 0.95$. (A) Temperature field predictions; (B) Absolute error between ground truth ($\mathcal{P}_c(T_t)$) and the output predicted - from top to bottom top - by: FNO, purely data-driven approach and hybrid NN-PDE. Hybrid NN-PDE model predicts physically accurate evolution of the flame cases under study

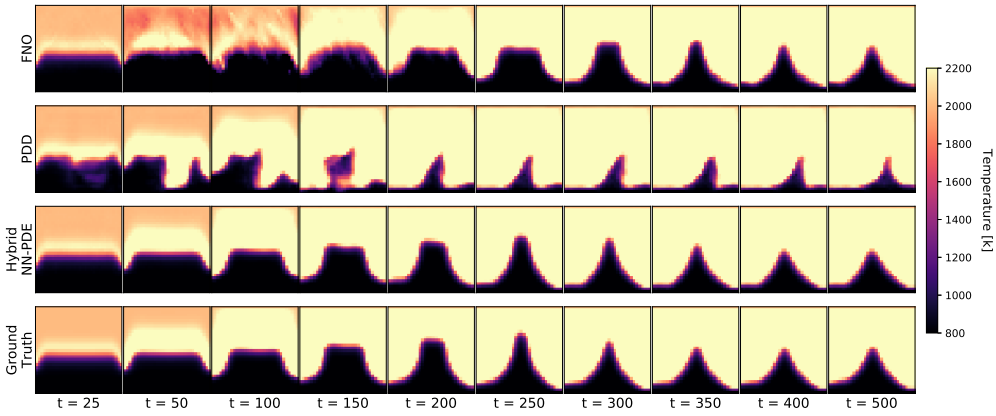


Figure 7.13: nonUniform-Bunsen32 : Comparison between different approaches - from top to bottom - FNO, PDD, hybrid NN-PDE for a test case. Hybrid approach predictions accurately match with the ground truth data over long-rollouts of 500 time steps

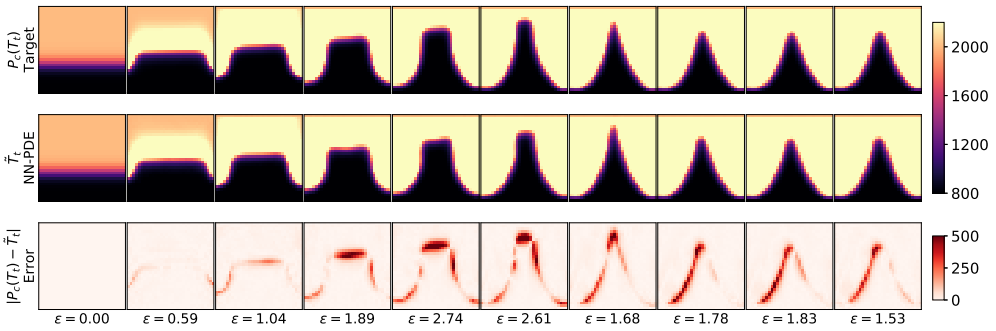


Figure 7.14: nonUniform-Bunsen32 : Visualization of reactive flow trajectories predicted by the hybrid approach for different test case

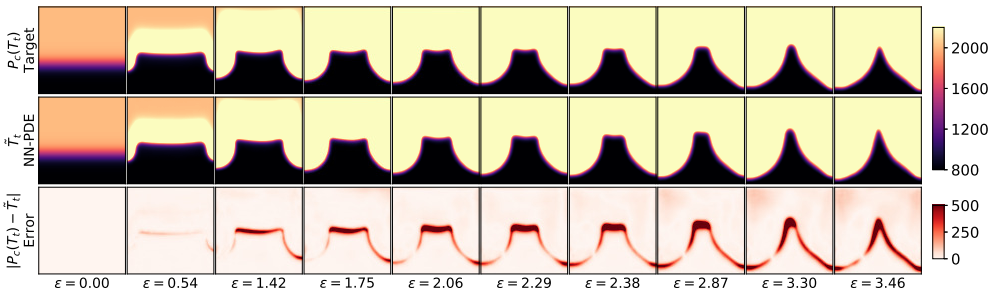


Figure 7.15: nonUniform-Bunsen100 : Comparison between ground truth data ($\mathcal{P}_c(T_t)$) and hybrid NN-PDE approach predictions ($\tilde{T}_t$) for a different test case of complex, high resolution scenario

two scenarios. However, it fails to predict accurate temporal predictions over the entire simulation. The hybrid NN-PDE approach predicts the accurate flame evolution well. Additionally, we showcase the performance of the hybrid NN-PDE approach on two different test cases with complex flame shapes in figure 7.14. The neural network model along with the incomplete PDE solver reconstructs these complex flame shapes in an accurate manner.

For the nonUniform-Bunsen100 case, figure 7.15 compares the predictions made by hybrid NN-PDE over $\{0, 100, 200, 250, 300, 325, 350, 400, 450, 500\}$ simulation steps with the ground truth data. It also shows the absolute difference between them. As the simulation progresses, higher errors are observed around the flame front. However, the hybrid NN-PDE approach captures the flame shape very accurately for longer roll-outs of 500 simulation steps. Currently, this approach uses a training dataset similar to nonUniform-Bunsen32. Further improvements in accuracy can be achieved by increasing the training dataset size or training the hybrid model with longer look-ahead steps.

7.5 Discussion

In this section, we study the effect of temporal coarsening, incorrect PDE parameters and longer look-ahead steps on predictions made by the hybrid NN-PDE solver.

7.5.1 Relaxing temporal stiffness in the PDE solver

Traditionally, the source terms and reaction rate terms involved in modelling the fast chemistry of reacting flows require the use of very small time-steps in simulations due to the stiffness of the chemical mechanisms. The incomplete PDE solver used in the hybrid NN-PDE approach, does not contain the source terms and reaction rate terms. Therefore, the time scales associated with the chemical reactions play a less important role in maintaining numerical stability, and it becomes possible for the solver to employ larger time steps. To illustrate this advantage, we train the neural network model with a time-step that is twice as large as the largest time-step Δt_c required for the complete PDE solver to yield a stable result.

We mimic the setup described in section 7.3, but now the incomplete PDE solver uses a time-step of $2\Delta t_c$, which is too large for the complete PDE solver by itself to converge to a solution. Figure 7.16 shows the results obtained from the models trained with a larger time-step are in good agreement with the target data for 4 different scenarios considered. The flame dynamics are predicted accurately while using less simulation steps. For the uniform-Bunsen case, the hybrid approach takes $8.23 \pm 0.011s$ to infer one simulation run, whereas the complete PDE solver requires $15.21 \pm 0.003s$. Similar performance gains are observed across the other cases studied. Note that for the previous cases with a timestep size of one Δt_c, the trained model incurs a negligible runtime overhead. Even though incomplete PDE solvers are cheaper to run, as mentioned in the introduction section, increasing the spatial or temporal resolution of the incomplete PDE solver would not converge to the solutions of the complete description. Using the incomplete solver,

these solutions cannot be obtained irrespective of how much computational resources are invested.

The last column of table 7.4 (*Hybrid NN-PDE-dt*) summarizes the errors of the large time-step approach for all four reactive cases considered. Despite an effectively doubled computational performance, this model achieves similar errors to the hybrid NN-PDE approach. This highlights the capabilities of learned, hybrid PDE solvers, which can produce these solutions without the stability problems exhibited by the complete PDE solver, while at the same time being more accurate than pure data-driven predictions.

7.5.2 Robustness of the hybrid solver

Generalization to incorrect PDE parameters The proposed hybrid NN-PDE model is capable of completing the PDE description even if the underlying incomplete PDE solver has incorrect parameters. We refer to the incomplete solver with incorrect parameters as 'incomplete, incorrect PDE'. We experiment with a modified hybrid NN-PDE approach wherein we combine an incomplete, incorrect PDE solver with a neural network model. We assume that the known values of the incomplete PDE parameters in equation (7.5), strain rate tensor (τ), the diffusion coefficient of species k (D_k) and mixture thermal conductivity (λ), are incorrect. Figure 7.17 (A) shows the difference between the temperature field evolution of the incomplete PDE solver with correct parameters ($[\tau, D_k, \lambda] = [0.1, 0.1, 0.1]$) and incorrect parameters ($[\tau, D_k, \lambda] = [0.05, 0.05, 0.05]$) at different time-steps. The hybrid NN-PDE combines this incomplete, incorrect PDE solver with the neural network model to obtain the solutions of the complete PDE solver with the correct parameters. Figure 7.17 (B) compares the flame dynamics predicted by this hybrid NN-PDE model with that of the complete, correct PDE solver. A good match is observed over various test cases. The hybrid model with an incomplete, incorrect solver achieves an overall MAPE of 2.48 $\pm$ 1.20 %, compared to the MAPE of 2.04 $\pm$ 1.39 % for the hybrid model with incomplete, correct PDE.

Effect of longer look-ahead steps We evaluate the effect of varying look-ahead steps on the performance of the hybrid model. We show the comparison of models trained with $m = \{2, 4, 8, 16, 32\}$ for different cases in figure 7.18. Models with smaller m (2) do not learn to accurately correct the fields over a long time and quickly diverge from the target simulation. Using larger m improves the quality of prediction drastically as the model learns the correction via the gradients over longer simulation steps. For Planar-v0 and uniform-Bunsen cases, iterating the NN and PDE solver for 32 time-steps improves the accuracy by 81.7% and 94% respectively compared to the $m = 2$ model. For nonUniform-Bunsen32 and nonUniform-Bunsen100, the model performance improves by 84.2% and 70.9% respectively, by using $m = 32$ instead of $m = 2$ model.

Effect of training dataset size We study the effect of training dataset size on the accuracy of PDD predictions. We train PDD models with different numbers of simulations in the training dataset. Each simulation contains 500 time-steps. Figure 7.19 (A) shows the MAPE of PDD models trained with $\{4, 8, 12, 16, 24, 32\}$ training datasets, over a fixed testset. We compare the performance of these PDD models with an equivalent (trained with the same look-ahead steps $m = 2$) hybrid NN-PDE model,

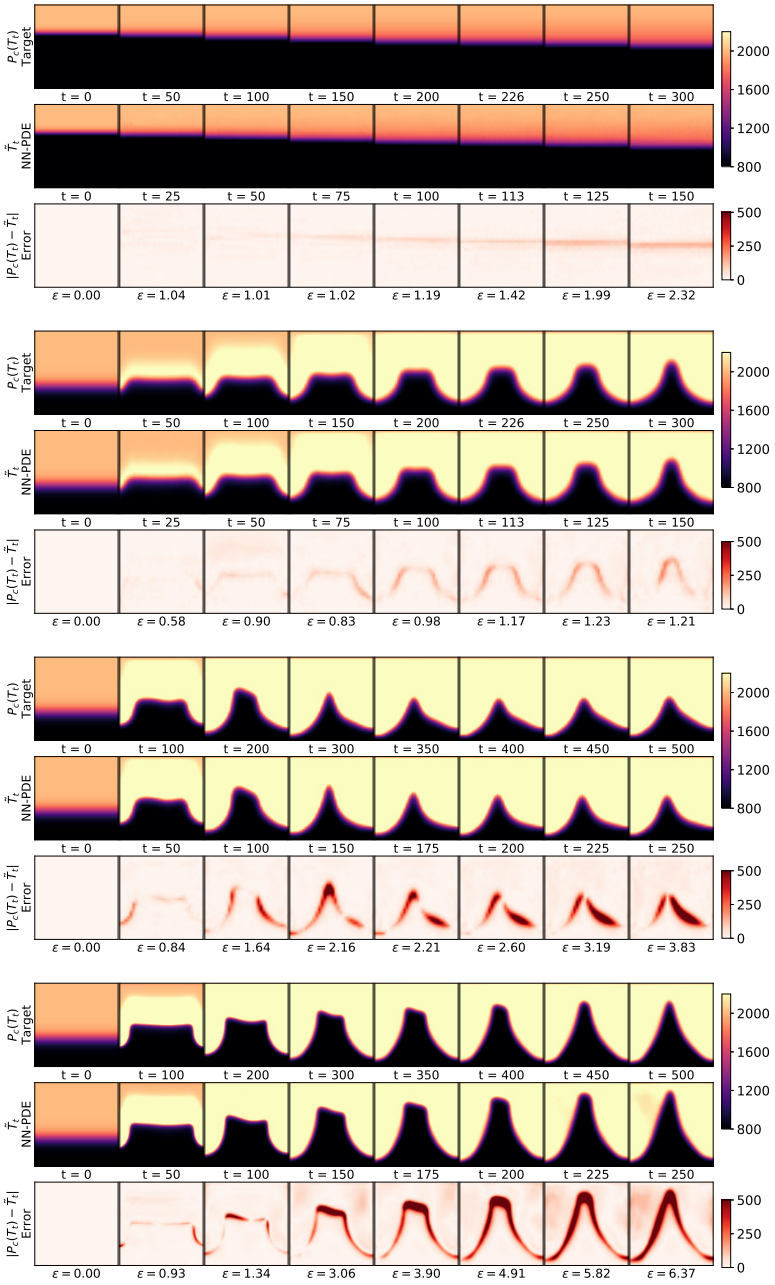


Figure 7.16: Predictions made by hybrid NN-PDE model ($\tilde{T}_t$) with an incomplete PDE solver at twice the time-step of $2\Delta t_c$, are compared with the ground truth solutions coming from the complete PDE solver at time-step of Δt_c. We showcase the effectiveness of the hybrid approach in relaxing temporal stiffness of the complete PDE solver on reactive flow cases of - from top to bottom - Planar-v0, uniform-Bunsen, nonUniform-Bunsen32 and nonUniform-Bunsen100 for different test cases

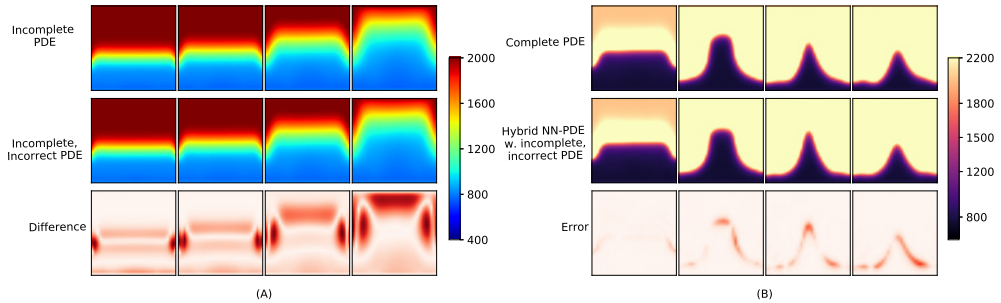


Figure 7.17: Generalization to incorrect PDE parameters (A) visualization of differences in temperature fields due to incorrect parameters in incomplete PDE description. (B) Modified hybrid NN-PDE model, which combines the incomplete, incorrect PDE solver with neural network model is able to recover solutions of complete, correct PDE

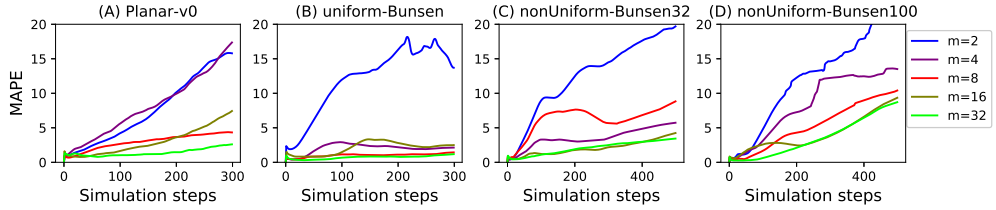


Figure 7.18: Effect of longer look-ahead steps. MAPE of temperature field predictions by hybrid NN-PDE model, over all testsets. Models trained with higher look-ahead steps m accurately predict the temporal evolution of dynamics for a longer duration across all cases considered.

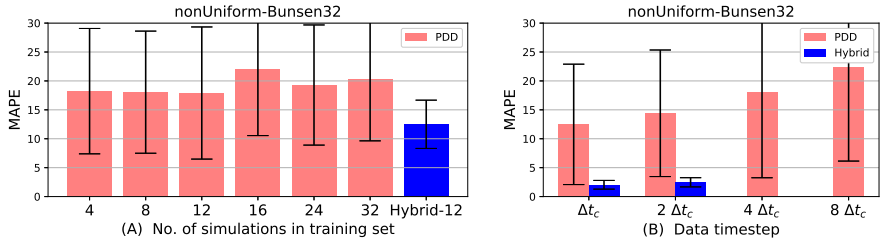


Figure 7.19: (A) Effect of increasing training dataset size for PDD approach, over fixed testsets, is compared with equivalent (trained with same look-ahead steps $m = 2$) hybrid NN-PDE model. (B) Effect of temporal coarsening on PDD models trained with $m = 32$ look-ahead steps

trained with 12 simulation sets. Figure 7.19 (A) shows increasing the number of training sets has little or no effect on the prediction capabilities of the PDD models for the nonUniform-Bunsen32 case. The hybrid model with 12 training sets achieves a MAPE of $12.49 \pm 4.17\%$, an improvement of 38% over the PDD model with 32 training sets. The purely data-driven models cannot achieve the same level of accuracy even in the presence

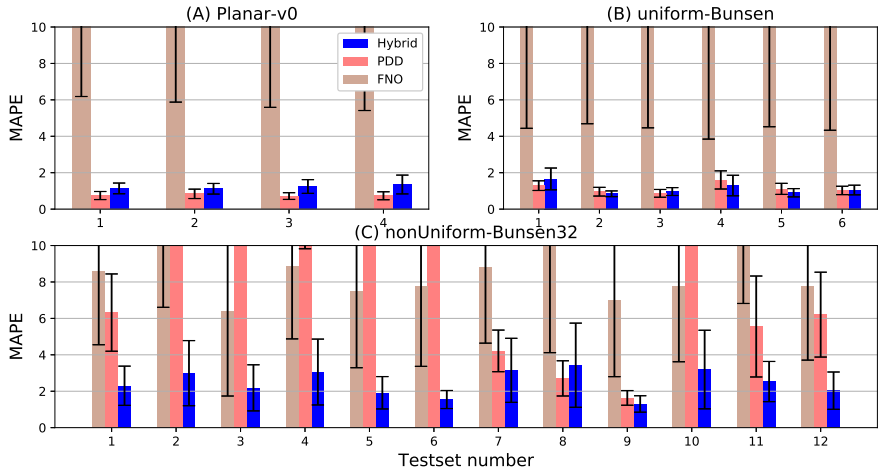


Figure 7.20: Bar plot of MAPE of temperature field predictions by FNO, PDD and hybrid NN-PDE model at time-step $2\Delta t_c$, for different testcases

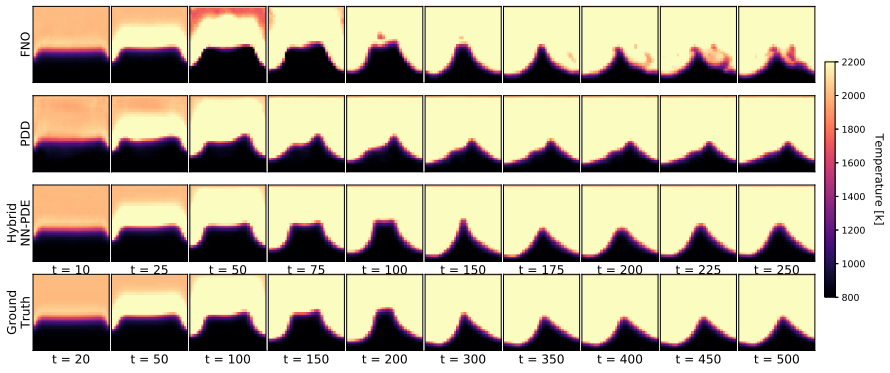


Figure 7.21: Comparison between different approaches at time-step $2\Delta t_c$ - from top to bottom - FNO, PDD, hybrid NN-PDE for the nonUniform-Bunsen32 test case. Hybrid approach predictions accurately match with the ground truth data over long roll-outs

of large amounts of data. This result strengthens the hypothesis that integrating the incomplete PDE solver into the neural network training yields a learning signal that fundamentally differs from that produced by training with pre-computed data. The purely data-driven models cannot achieve the same level of accuracy even in the presence of large amounts of data.

Effect of temporal coarsening Table 7.4 compare the MAPE of temperature field predictions at time-step Δt_c, for all three scenarios. The hybrid NN-PDE approach consistently performs better than both the baselines for all three scenarios considered. Higher standard deviations in table 7.4 indicate the large differences in the prediction of flame temperatures for baseline approaches. The multi-physics systems we study provide a substantially more difficult environment than regular fluids: the chemical reactions

are numerically very stiff, and the resulting dynamic interactions are difficult to capture by numerical solvers. To further illustrate the temporal behaviour of the simulations, we perform additional experiments with the baseline approaches using twice the time-step size of the complete solver ($2\Delta t_c$). As shown in figure 7.20, for the Planar-v0 and uniform-Bunsen case, PDD catches up with the hybrid approach whereas the PDD approach completely fails to predict the dynamics of the nonUniform-Bunsen32 case. We further investigate the performance of the PDD approach on the nonUniform-Bunsen32 case with larger time-steps (2 times, 4 times and 8 times) as shown in figure 7.19 (B). PDD achieves a higher MAPE of 22.44 % for $8\Delta t_c$ setup as compared to the MAPE of 12.48 % for Δt_c setup. The PDD approach does not predict the dynamics accurately for any of the larger time-steps considered. We can not showcase the performance of the hybrid NN-PDE solver for time steps greater than $2\Delta t_c$ as it is restricted by the underlying incomplete PDE solver. Figure 7.21 visualize these results over different time-steps for one of the test cases. The predictions made by FNO, PDD and hybrid NN-PDE model for the nonUniform-Bunsen32 case at twice the time-step of $2\Delta t_c$, are compared with the ground truth solutions coming from the complete PDE solver at time-step of Δt_c. FNO and PDD (upper two rows) fail to recover the correct flame shapes over 250 simulation steps. The hybrid NN-PDE model (second row from below) predicts the flame shape accurately, thus relaxing the temporal stiffness of the complete PDE solver.

7.6 Controlling Flame Dynamics

To demonstrate the flexibility of the hybrid NN-PDE solver developed in this , we consider a multi-physics system control problem. We test the joint applicability of a neural network with a pre-trained hybrid reactive flow solver to obtain the desired flame shape. We integrate the hybrid solver as a layer into the neural network model to enable the training of a flame shape controller. As a result, physics knowledge from the hybrid solver can be embedded into the optimization problem by matching observations coming from the solver. The hybrid solver provides the agent with feedback on how interactions at any point in time affect the flame shape. We consider a physical system of reactive flows where the agent can interact with the system by controlling the inlet velocity of fuel-air flow.

We use a predictor-corrector approach [26] with a supervised loss function and control sequence refinement for the nonUniform-Bunsen case with 32×32 resolution. A pre-trained hybrid solver from section 7.4.2.2 is used as a reactive flow solver. This hybrid model consists of an incomplete PDE solver along with an UNet model detailed in section 7.2.3, section 7.2.4 and figures 7.2, 7.3. It is first trained using training instances of the uniform-Bunsen32 case described in section 7.3. Additionally, it is further trained on the training data of flame shape transitions from an arbitrary flame shape to any other arbitrary flame shape. Once trained, the weights of the hybrid solver are frozen and used as a layer into the predictor-corrector approach. It is differentiable by construction, when combined with the deep neural networks is shown to learn a policy to control flame

shapes. The learning objective is to arrive at a target temperature field (T_*) with the desired flame shape from an arbitrary, given initial configuration (T_0) by controlling the velocity field (u_i^c). We consider a real-world setting where only few states of the reactive flow system are observable and controllable: the temperature and mass-fraction fields are observable, and only the inlet of the velocity field is controllable. The predictor-corrector approach consists of two neural network models. First, an observation predictor network (OP), predicts the intermediate states of the observable quantity i.e. temperature in this case (T_i^p). The corrector (CFE) network predicts the control force (u_i^c) required to follow the trajectory predicted by OP network. The inlet velocity field predicted by the CFE network is used by the pre-trained hybrid solver to compute the corrected temperature field (T_i^c). Figure 7.22 shows the sequence of OP-CFE network used to control the flame shape T_* given the initial flame shape T_0, over the trajectory of 4 time-steps. The OP network is modelled as a temporal hierarchical process to incorporate knowledge about longer time spans. Therefore, OP networks are trained to predict the optimal center point between two time steps instead of predicting the state of the next time step. The recursive call to the OP network enables the prediction of observable quantities at every time step (T_i^p). Using T_{i-1} and OP prediction at step i, i.e. T_i^p, CFE network predicts the control force at step i, u_i^c. Once u_i^c is predicted, the learned hybrid solver can be used to advance the simulation to next time step and obtain the actual state T_i^c. Using this sequence of OP-CFE network, we can obtain T_n^c with n CFE evaluations, which matches target flame shape T_* very closely. This sequence can be generalized to any arbitrarily long length of sequences. OP-CFE networks are trained using the supervised loss function on the target temperature field achieved and inlet velocity predicted, respectively. The loss function formulations for OP and CFE network are,

$$L_{op} = |OP[T_i, T_j] - T^{GT}(\frac{i+j}{j})|^2 \qquad L_{CFE} = \sum_{i=1}^{n} |u_i^c - u_i^*|^2, \tag{7.9}$$

where T^{GT} and u^* indicate the ground truth temperature field and inlet velocity, respectively. The simplest technique to obtain the optimal trajectory, given an initial state is to use the chain of CFE networks followed by the solver. CFE, modelled using deep neural network, predicts the control force required to predict the transition to the next observable state. Therefore, as a baseline model, we consider the CFE only approach. All neural networks used in this work are modified UNet architectures [148].

Figure 7.23 (b-e) shows examples of the transition achieved and compares the target flame shape obtained using the neural network model with the target flame shape (T_*). All three cases are test cases, i.e. were not seen at training time, and have a sequence length of 64 simulation steps. It is visible that the neural network model succeeds at controlling the inlet velocity of fuel-air mixture flow to obtain the desired flame shapes. To quantify these results, the learned model achieves MAPE of 1.34 $\pm$ 0.41% over 50 test cases considered. It achieves an improvement of 65.7% compared to a baseline model of the CFE-only approach, without a predictor network for the long term prediction of temperature fields. This baseline model has a MAPE of 3.91 $\pm$ 2.19%. This

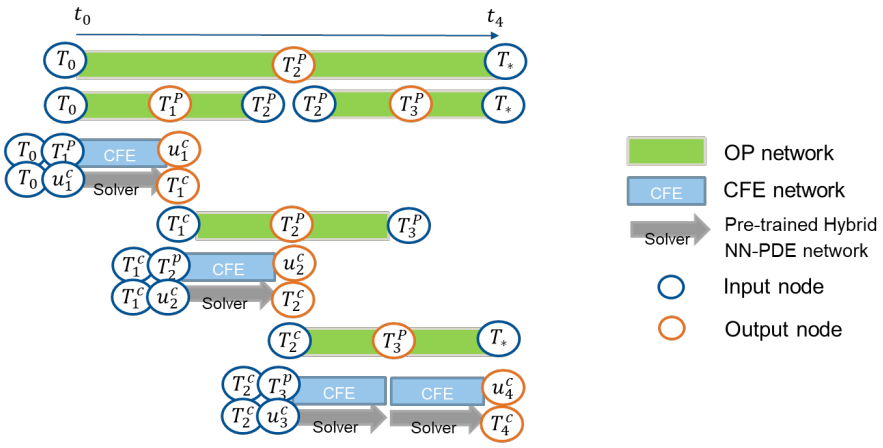


Figure 7.22: Details of the OP-CFE prediction sequence

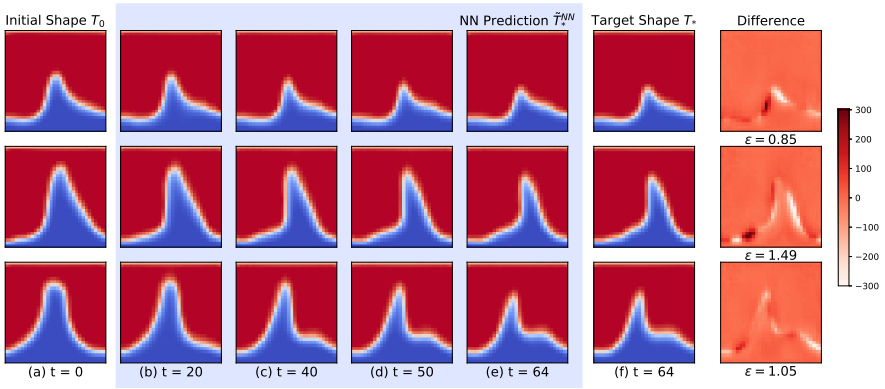


Figure 7.23: Temperature field predictions achieved by the neural network model $\tilde{T}_*^{NN}$ from given initial flame shape T_0 to achieve target flame shape $\tilde{T}_*$ by controlling flow velocity

case highlights the capabilities of differentiable hybrid NN-PDE multi-physics solvers. Specifically, the hybrid solver helps the OP-CFE networks to yield controllers that steer the complex physics of the reactive flow over long time spans.

8 Conclusion & Outlook

In this dissertation, we focus on modelling flame dynamics using deep learning-based algorithms. In the following, we will summarize all methods discussed in previous chapters. Afterwards, limitations and future directions are presented.

8.1 Summary

In the previous chapters, we have presented the ability of neural networks to model nonlinear flame dynamics in various applications, such as nonlinear flame response, the temporal evolution of reactive flow simulations and control of reactive flow simulations. We demonstrated the ability of the neural networks to accurately and efficiently model the nonlinear flame response compared to the conventional methods. It requires data from only one CFD simulation, which is excited with a broadband signal characterized by a uniform distribution of all amplitudes of interest. Our neural network modelling approach offers a significant advantage in capturing the flame response not only at the input frequency but also at higher harmonics. This is reflected in further evaluations; when coupled with an acoustic solver, the trained neural network accurately predicts self-excited thermo-acoustic oscillations. We tackle the challenge of the reconstruction of unmeasured quantities in reacting flows. The convolutional neural network-based model can be used to reconstruct the heat release rate from velocity measurements in a methane/air premixed flame under harmonic excitation, thus reducing the complexity of measuring the heat release rate for unseen operating points. In addition, another work shows how we can combine a neural network model with a non-reactive PDE solver to obtain the solutions of reactive flow simulation. The problem deals with the modelling of the complex reactive flow dynamics when complete information on the physics of the system is missing. We propose a hybrid NN-PDE solver which learns to complete an incomplete PDE solver (i.e. non-reactive flow solver) when the underlying chemistry is unknown. The incomplete PDE solver helps the neural network model recover the target simulation with a significantly improved accuracy. We showcase the applicability of this framework to the problem of controlling an arbitrary flame shape to achieve a desired

target flame shape. All methods developed in the present thesis are formulated in a very general way and can easily be applied to a variety of other problems.

8.2 Outlook

We have demonstrated the effectiveness of deep learning methods in modelling flame dynamics. We believe that combining the advantage of physical simulations and neural network models is a promising direction. While achieving superior performance over existing methods, we are aware of the limitations, including loss of interpretability (black box nature of the model), the large training time required, sub-optimal results for extrapolated operating points. These limitations can be an inspiration for future work.

Uncertainty Quantification Thermo-acoustic stability analysis is essential to the gas turbine design process. We showcase the ability of neural network models to capture the flame response for variations in SISO systems. To extend this approach to MISO or MIMO systems, multiple CFD simulations could potentially be used to capture the variance in multiple input systems. The NN inputs need to be modified to account for multiple lags associated with such systems. Additionally, a lack of exact knowledge of operating conditions (such as inlet flow velocity, equivalence ratio, wall temperature) yields uncertainties in the flame response model. Therefore, neural network models need to be adapted to quantify these uncertainties. The NN model for flame describing function has aleatoric uncertainty arising from the data with combustion noise and epistemic uncertainty caused by a lack of knowledge of operating or boundary conditions. The Bayesian framework is capable of providing uncertainty analysis. Bayesian neural network [149, 150] can be used to encode the aleatoric and the epistemic uncertainty [151]. Monte Carlo dropout [152, 153] is a simple to implement method to model uncertainties, as no changes to the standard deep learning training procedure are required. Other methods, such as mixture density networks [154], deep ensembles [155], deep Gaussian mixture ensembles [156], can be used to estimate both types of uncertainties. These models enable ways to incorporate prior domain information and may lead to better interpretability.

Differential reactive flow solvers We have demonstrated the applicability of the hybrid NN-PDE approach on laminar, Bunsen-type flames. However, turbulent flames are more common in typical combustion systems. Therefore, applying the proposed methods to the turbulent flame systems would be an interesting future direction. Combining a neural network model with a differentiable PDE solver has exciting applications [27, 28]. A coarse mesh reactive flow solver can be integrated with a neural network model to obtain the fine mesh solutions. These methods have shown to be faster than the fine mesh solver for turbulent flow simulations [28, 95]. ReacTorch [157] and JAX-reactor [158] are packages for simulating chemically reacting flows in PyTorch and Jax respectively. These packages are aimed at providing differentiating capabilities for chemically reactive flow simulations. It would be beneficial to drive the development of these packages and existing differentiable solvers, such as PhiFlow [84], SU2 [85], SciML [86], to include reactive flow simulations. The differential PDE solver for reactive flows

can be integrated with neural network model for multiple applications. The neural network can be effectively trained to correct for sub-grid scale reaction rates when combined with a low-resolution reactive flow solver, thus improving approximations inside CFD. As an extension to the flow control application demonstrated in Section 7.6, the differentiable reactive flow solver can be integrated with NN model to control inlet conditions to avoid thermoacoustic instabilities or flashbacks.

Handling of real-world, sparse, noisy data This study has developed methods to predict thermoacoustic instabilities in the design phase of the combustor. This work focuses on enhancing solutions of numerical simulations using neural network models. Going forward, it would be beneficial to extend these methods for experimental evaluations. Applying the proposed approach to cases, where a limited amount of experimental data is available would be a very interesting step. Especially if the data is very sparse and contains noise, learning meaningful dynamics could become excessively challenging. Therefore, combining structured knowledge and learning-based methods has the potential to create meaningful representations.

Bibliography

[1] C. J. Goy, S. R. James, and S. Rea. Monitoring combustion instabilities: E. ON UK's experience. *Progress in Astronautics and Aeronautics*, 210:163, 2005.

[2] T. Emmert, S. Jaensch, C. Sovardi, and W. Polifke. taX - A Flexible Tool for Low-Order Duct Acoustic Simulation in Time and Frequency Domain. In *Forum Acusticum, Krakow, Poland, Sept*, pages 7–12, 2014.

[3] F. Selimefendigil and W. Polifke. A nonlinear frequency domain model for limit cycles in thermoacoustic systems with modal coupling. *International Journal of Spray and Combustion Dynamics*, 3(4):303–330, 2011.

[4] S. Jaensch, M. Merk, E. Gopalakrishnan, S. Bomberg, T. Emmert, R. Sujith, and W. Polifke. Hybrid CFD/low-order modeling of nonlinear thermoacoustic oscillations. *Proceedings of the Combustion Institute*, 36(3):3827–3834, 2017.

[5] M. Merk, W. Polifke, R. Gaudron, M. Gatti, C. Mirat, and T. Schuller. Measurement and simulation of combustion noise and dynamics of a confined swirl flame. *AIAA Journal*, 56(5):1930–1942, 2018.

[6] A. J. Eder, B. Dharmaputra, M. Désor, C. F. Silva, A. M. Garcia, B. Schuermans, N. Noiray, and W. Polifke. Generation of entropy waves by fully premixed flames in a non-adiabatic combustor with hydrogen enrichment. In *Proceedings of ASME Turbo Expo 2023*, Boston, MA, USA, 2023.

[7] B. List, L.-W. Chen, and N. Thuerey. Learned turbulence modelling with differentiable fluid solvers. *arXiv preprint arXiv:2202.06988*, 2022.

[8] M. Zastempowski. Analysis and modeling of innovation factors to replace fossil fuels with renewable energy sources-evidence from european union enterprises. *Renewable and Sustainable Energy Reviews*, 178:113262, 2023.

[9] Eurostat. statistical data. 2022. doi:https://doi.org/10.2908/NRG_IND_FFGAE.

[10] M. Stefanizzi, T. Capurso, G. Filomeno, M. Torresi, and G. Pascazio. Recent combustion strategies in gas turbines for propulsion and power generation toward a zero-emissions future: Fuels, burners, and combustion techniques. *Energies*, 14(20):6694, 2021.

[11] Y. Tian, S. Yang, and J. Le. Study on flame stabilization of a hydrogen and kerosene fueled combustor. *Aerospace Science and Technology*, 59:183–188, 2016.

[12] T. Tan, W. Fan, and R. Zhang. Study on hydrogen staged combustion for gas turbine. *Fuel*, 349:128700, 2023.

[13] T. C. Lieuwen and V. Yang. *Combustion instabilities in gas turbine engines: operational experience, fundamental mechanisms, and modeling*. American Institute of Aeronautics and Astronautics, 2005.

[14] J. Beita, M. Talibi, S. Sadasivuni, and R. Balachandran. Thermoacoustic instability considerations for high hydrogen combustion in lean premixed gas turbine combustors: a review. *Hydrogen*, 2(1):33–57, 2021.

[15] A. A. Khateeb, T. F. Guiberti, X. Zhu, M. Younes, A. Jamal, and W. L. Roberts. Stability limits and no emissions of technically-premixed ammonia-hydrogen-nitrogen-air swirl flames. *International journal of hydrogen energy*, 45(41):22008–22018, 2020.

[16] M. Foust, D. Thomsen, R. Stickles, C. Cooper, and W. Dodds. Development of the GE aviation low emissions taps combustor for next generation aircraft engines. In *50th AIAA aerospace sciences meeting including the new horizons forum and aerospace exposition*, page 936, 2012.

[17] T. Poinsot and D. Veynante. *Theoretical and numerical combustion*. RT Edwards, Inc., 2005.

[18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[19] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[20] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[23] A. Radford and K. Narasimhan. Improving language understanding by generative pre-training. 2018. URL: `https://api.semanticscholar.org/CorpusID:49313245`.

[24] N. Thuerey, K. Weißenow, L. Prantl, and X. Hu. Deep learning methods for Reynolds-averaged navier-stokes simulations of airfoil flows. *AIAA Journal*, 58(1):25–36, 2020.

[25] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[26] P. Holl, V. Koltun, and N. Thuerey. Learning to control PDEs with differentiable physics. In *ICLR*, 2019.

[27] K. Um, R. Brand, F. Yun, P. Holl, and N. Thuerey. Solver-in-the-loop: Learning from differentiable physics to interact with iterative PDE-solvers. In *Neural Information Processing Systems (NeurIPS)*, volume 33, pages 6111–6122, 2020.

[28] D. Kochkov, J. A. Smith, A. Alieva, Q. Wang, M. P. Brenner, and S. Hoyer. Machine learning accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21), 2021.

[29] M. A. Vaudrey and W. R. Saunders. Control of combustor instabilities using an artificial neural network. In *Turbo Expo: Power for Land, Sea, and Air*, volume 78545, page V001T03A093. American Society of Mechanical Engineers, 2000.

[30] R. Blonbou, A. Laverdant, S. Zaleski, and P. Kuentzmann. Active control of combustion instabilities on a rijke tube using neural networks. volume 28, pages 747–755. Elsevier, 2000.

[31] W. Polifke. Modeling and analysis of premixed flame dynamics by means of distributed time delays. *Progress in Energy and Combustion Science*, 79:100845, 2020.

[32] V. Kornilov, R. Rook, J. ten Thije Boonkkamp, and L. De Goey. Experimental and numerical investigation of the acoustic response of multi-slit Bunsen burners. *Combustion and Flame*, 156(10):1957–1970, 2009.

[33] S. Jaensch and W. Polifke. Uncertainty encountered when modelling self-excited thermoacoustic oscillations with artificial neural networks. *International Journal of Spray and Combustion Dynamics*, 9(4):367–379, 2017.

[34] S. Kulkarni, S. Guo, C. F. Silva, and W. Polifke. Confidence in flame impulse response estimation from large eddy simulation with uncertain thermal boundary conditions. *Journal of Engineering for Gas Turbines and Power*, 143(12), 2021.

[35] B. Kim, V. C. Azevedo, N. Thuerey, T. Kim, M. Gross, and B. Solenthaler. Deep fluids: A generative network for parameterized fluid simulations. In *Computer Graphics Forum*, pages 59–70. Wiley Online Library, 2019.

[36] C. J. Lapeyre, A. Misdariis, N. Cazard, D. Veynante, and T. Poinsot. Training convolutional neural networks to estimate turbulent sub-grid scale reaction rates. *Combustion and Flame*, 203:255–264, 2019.

[37] J. Kukkonen, T. Olsson, D. M. Schultz, A. Baklanov, T. Klein, A. Miranda, A. Monteiro, M. Hirtl, V. Tarvainen, M. Boy, et al. A review of operational, regional-scale, chemical weather forecasting models in europe. *Atmospheric Chemistry and Physics*, 12(1):1–87, 2012.

[38] F. Williams. *Combustion Theory*. CRC Press, 2018.

[39] J. O. Hirschfelder, C. F. Curtiss, and R. B. Bird. The molecular theory of gases and liquids. John Wiley & Sons, 1964.

[40] K. Kuo. *Principles of Combustion*. Wiley, 2005.

[41] N. Noiray, D. Durox, T. Schuller, and S. Candel. A unified framework for nonlinear combustion instability analysis based on the flame describing function. *Journal of Fluid Mechanics*, 615:139–167, 2008.

[42] C. F. Silva, F. Nicoud, T. Schuller, D. Durox, and S. Candel. Combining a Helmholtz solver with the flame describing function to assess combustion instability in a premixed swirled combustor. *Combustion and Flame*, 160(9):1743–1754, 2013.

[43] N. Noiray, D. Durox, T. Schuller, and S. Candel. Self-induced instabilities of premixed flames in a multiple injection configuration. *Combustion and flame*, 145(3):435–446, 2006.

[44] N. Noiray, D. Durox, T. Schuller, and S. Candel. Passive control of combustion instabilities involving premixed flames anchored on perforated plates. *Proceedings of the Combustion Institute*, 31(1):1283–1290, 2007.

[45] B. D. Bellows, M. K. Bobba, A. Forte, J. M. Seitzman, and T. Lieuwen. Flame transfer function saturation mechanisms in a swirl-stabilized combustor. *Proceedings of the combustion institute*, 31(2):3181–3188, 2007.

[46] B. D. Bellows, M. K. Bobba, J. M. Seitzman, and T. Lieuwen. Nonlinear flame transfer function characteristics in a swirl-stabilized combustor. In *Turbo Expo: Power for Land, Sea, and Air*, volume 42363, pages 823–833, 2006.

[47] P. Palies, D. Durox, T. Schuller, and S. Candel. The combined dynamics of swirler and turbulent premixed swirling flames. *Combustion and flame*, 157(9):1698–1717, 2010.

[48] P. Palies, D. Durox, T. Schuller, and S. Candel. Nonlinear combustion instability analysis based on the flame describing function applied to turbulent premixed swirling flames. *Combustion and Flame*, 158(10):1980–1991, 2011.

[49] H. Krediet, C. Beck, W. Krebs, S. Schimek, C. Paschereit, and J. Kok. Identification of the flame describing function of a premixed swirl flame from LES. *Combustion Science and Technology*, 184(7-8):888–900, 2012.

[50] X. Han and A. S. Morgans. Simulation of the flame describing function of a turbulent premixed flame using an open-source LES solver. *Combustion and Flame*, 162(5):1778–1792, 2015.

[51] N. Stiennon, L. Ouyang, J. Wu, D. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. F. Christiano. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021, 2020.

[52] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[53] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016.

[54] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

[55] S. Schlimpert, M. Meinke, and W. Schröder. Nonlinear analysis of an acoustically excited laminar premixed flame. *Combustion and Flame*, 163:337–357, 2016.

[56] K. T. Kim. Nonlinear interactions between the fundamental and higher harmonics of self-excited combustion instabilities. *Combustion Science and Technology*, 189(7):1091–1106, 2017.

[57] M. Haeringer, M. Merk, and W. Polifke. Inclusion of higher harmonics in the flame describing function for predicting limit cycles of self-excited combustion instabilities. *Proceedings of the Combustion Institute*, 37(4):5255–5262, 2019.

[58] A. Orchini and M. P. Juniper. Flame double input describing function analysis. *Combustion and Flame*, 171:87–102, 2016.

[59] W. Polifke, A. Poncet, C. Paschereit, and K. Döbbeling. Reconstruction of acoustic transfer matrices by instationary computational fluid dynamics. *Journal of Sound and Vibration*, 245(3):483–510, 2001.

[60] A. P. Dowling. Nonlinear self-excited oscillations of a ducted flame. *Journal of fluid mechanics*, 346:271–290, 1997.

[61] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.

[62] F. Selimefendigil, S. Föller, and W. Polifke. Nonlinear identification of unsteady heat transfer of a cylinder in pulsating cross flow. *Computers & Fluids*, 53:1–14, 2012.

[63] K. Förner and W. Polifke. Nonlinear aeroacoustic characterization of Helmholtz resonators with a local-linear neuro-fuzzy network model. *Journal of Sound and Vibration*, 407:170–190, 2017.

[64] M. Rywik, D. da Cruz, and W. Polifke. Investigating inconsistent uncertainty quantification encountered in neural network modelling of nonlinear response of a laminar flame. In *INTER-NOISE and NOISE-CON Congress and Conference Proceedings*, volume 265, pages 6199–6207. Institute of Noise Control Engineering, 2023.

[65] N. Tathawadekar, N. A. K. Doan, C. F. Silva, and N. Thuerey. Modeling of the nonlinear flame response of a Bunsen-type flame via multi-layer perceptron. *Proceedings of the Combustion Institute*, 38(4):6261–6269, 2021.

[66] V. Yadav, M. Casel, and A. Ghani. Physics-informed recurrent neural networks for linear and nonlinear flame dynamics. *Proceedings of the Combustion Institute*, 2022.

[67] M. Rywik, A. Zimmermann, A. J. Eder, E. Scoletta, and W. Polifke. Spatially resolved modeling of the nonlinear dynamics of a laminar premixed flame with a multilayer perceptron-convolution autoencoder network. In *Turbo Expo: Power for Land, Sea, and Air*, volume 86953, page V03AT04A069. American Society of Mechanical Engineers, 2023.

[68] I. E. Lagaris, A. Likas, and D. I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE transactions on neural networks*, 9(5):987–1000, 1998.

[69] Z. Long, Y. Lu, X. Ma, and B. Dong. PDE-net: Learning PDEs from data. In *International Conference on Machine Learning*, pages 3208–3216. PMLR, 2018.

[70] J. Han, A. Jentzen, and E. Weinan. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, 115(34):8505–8510, 2018.

[71] Y. Bar-Sinai, S. Hoyer, J. Hickey, and M. P. Brenner. Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31):15344–15349, 2019.

[72] K. Fukami, K. Fukagata, and K. Taira. Super-resolution reconstruction of turbulent flows with machine learning. *Journal of Fluid Mechanics*, 870:106–120, 2019.

[73] K. Stachenfeld, D. B. Fielding, D. Kochkov, M. Cranmer, T. Pfaff, J. Godwin, C. Cui, S. Ho, P. Battaglia, and A. Sanchez-Gonzalez. Learned simulators for turbulence. In *International Conference on Learning Representations*, 2022. URL: `https://openreview.net/forum?id=msRBojTz-Nh`.

[74] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

[75] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*, 2020.

[76] K. Bhattacharya, B. Hosseini, N. B. Kovachki, and A. M. Stuart. Model reduction and neural networks for parametric PDEs. *The SMAI journal of computational mathematics*, 7:121–157, 2021.

[77] R. G. Patel, N. A. Trask, M. A. Wood, and E. C. Cyr. A physics-informed operator regression framework for extracting data-driven continuum models. *Computer Methods in Applied Mechanics and Engineering*, 373:113500, 2021.

[78] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

[79] S. Greydanus, M. Dzamba, and J. Yosinski. Hamiltonian neural networks. *Advances in Neural Information Processing Systems*, 32:15379–15389, 2019.

[80] V. Dwivedi and B. Srinivasan. Physics informed extreme learning machine (pielm)–a rapid method for the numerical solution of partial differential equations. *Neurocomputing*, 391:96–118, 2020.

[81] O. Fuks and H. A. Tchelepi. Limitations of physics informed machine learning for nonlinear two-phase transport in porous media. *Journal of Machine Learning for Modeling and Computing*, 1(1), 2020.

[82] S. Wang, H. Wang, and P. Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed deeponets. *Science advances*, 7(40):eabi8605, 2021.

[83] Z. Li, H. Zheng, N. Kovachki, D. Jin, H. Chen, B. Liu, K. Azizzadenesheli, and A. Anandkumar. Physics-informed neural operator for learning partial differential equations. *arXiv preprint arXiv:2111.03794*, 2021.

[84] N. Thuerey, P. Holl, M. Mueller, P. Schnell, F. Trost, and K. Um. Physics-based deep learning. *arXiv preprint arXiv:2109.05237*, 2021.

[85] T. D. Economon, F. Palacios, S. R. Copeland, T. W. Lukaczyk, and J. J. Alonso. SU2: An open-source suite for multiphysics simulation and design. *AIAA Journal*, 54(3):828–846, 2016.

[86] C. Rackauckas, Y. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, A. Ramadhan, and A. Edelman. Universal differential equations for scientific machine learning. *arXiv preprint arXiv:2001.04385*, 2020.

[87] R. Wang, K. Kashinath, M. Mustafa, A. Albert, and R. Yu. Towards physics-informed deep learning for turbulent flow prediction. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1457–1466, 2020.

[88] E. A. Illarramendi, M. Bauerheim, and B. Cuenot. Performance and accuracy assessments of an incompressible fluid solver coupled with a deep convolutional neural network. *Data-Centric Engineering*, 3:e2, 2022.

[89] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud. Neural ordinary differential equations. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 6572–6583, 2018.

[90] J. Tompson, K. Schlachter, P. Sprechmann, and K. Perlin. Accelerating eulerian fluid simulation with convolutional networks. In *International Conference on Machine Learning*, pages 3424–3433. PMLR, 2017.

[91] A. G. Özbay, A. Hamzehloo, S. Laizet, P. Tzirakis, G. Rizos, and B. Schuller. Poisson cnn: Convolutional neural networks for the solution of the poisson equation on a cartesian mesh. *Data-Centric Engineering*, 2:e6, 2021. doi:10.1017/dce.2021.7.

[92] E. Ajuria Illarramendi, A. Alguacil, M. Bauerheim, A. Misdariis, B. Cuenot, and E. Benazera. Towards an hybrid computational strategy based on deep learning for incompressible flows. In *AIAA AVIATION 2020 FORUM*, page 3058, 2020.

[93] J. Pathak, M. Mustafa, K. Kashinath, E. Motheau, T. Kurth, and M. Day. Using machine learning to augment coarse-grid computational fluid dynamics simulations. *arXiv preprint arXiv:2010.00072*, 2020.

[94] G. Dresdner, D. Kochkov, P. Norgaard, L. Zepeda-Núñez, J. A. Smith, M. P. Brenner, and S. Hoyer. Learning to correct spectral methods for simulating turbulent flows. *arXiv preprint arXiv:2207.00556*, 2022.

[95] F. d. A. Belbute-Peres, T. Economon, and Z. Kolter. Combining differentiable PDE solvers and graph neural networks for fluid flow prediction. In *International Conference on Machine Learning*, pages 2402–2411. PMLR, 2020.

[96] P. Holl, V. Koltun, K. Um, T. P. LTCI, I. Paris, and N. Thuerey. phiflow: A differentiable PDE solving framework for deep learning via physical simulations. In *NeurIPS Workshop*, 2020.

[97] J. Sirignano, J. F. MacArt, and J. B. Freund. Dpm: A deep learning PDE augmentation method with application to large-eddy simulation. *Journal of Computational Physics*, 423:109811, 2020.

[98] Y. Yin, V. Le Guen, J. Dona, E. de Bézenac, I. Ayed, N. Thome, and P. Gallinari. Augmenting physical models with deep networks for complex dynamics forecasting. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124012, 2021.

[99] N. Takeishi and A. Kalousis. Physics-integrated variational autoencoders for robust and interpretable generative modeling. *Advances in Neural Information Processing Systems*, 34:14809–14821, 2021.

[100] L. Selle, G. Lartigue, T. Poinsot, R. Koch, K.-U. Schildmacher, W. Krebs, B. Prade, P. Kaufmann, and D. Veynante. Compressible large eddy simulation of turbulent combustion in complex geometry on unstructured meshes. *Combustion and Flame*, 137(4):489–505, 2004.

[101] S. Föller and W. Polifke. Advances in identification techniques for aero-acoustic scattering coefficients from large eddy simulation. In *18th International congress on sound and vibration (ICSV18)*, 2011.

[102] F. Masters and K. R. Gurley. Non-gaussian simulation: cumulative distribution function map-based spectral correction. *Journal of engineering mechanics*, 129(12):1418–1428, 2003.

[103] V. Kornilov, K. Schreel, and L. De Goey. Experimental assessment of the acoustic response of laminar premixed Bunsen flames. *Proceedings of the Combustion Institute*, 31(1):1239–1246, 2007.

[104] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

[105] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning*, pages 807–814, 2010.

[106] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[107] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

[108] J. P. Moeck, C. Scharfenberg, O. Paschereit, and R. Klein. A zero-Mach solver and reduced order acoustic representations for modeling and control of combustion instabilities. In *Active Flow Control II*, pages 291–306. Springer, 2010.

[109] N. Tathawadekar, A. Ösün, A. J. Eder, C. F. Silva, and N. Thuerey. Linear and nonlinear flame response prediction of turbulent flames using neural network models. *International Journal of Spray and Combustion Dynamics*, 2024. doi:10.1177/17568277241262641.

[110] F. Nicoud and F. Ducros. Subgrid-Scale Stress Modelling Based on the Square of the Velocity Gradient Tensor. *Flow Turbulence and Combustion*, 62(3):183–200, 1999. doi:10.1023/A:1009995426001.

[111] O. Colin, F. Ducros, D. Veynante, and T. Poinsot. A Thickenend Flame Model for Large Eddy Simulation of Turbulent Premixed Combustion. *Physics of Fluids*, 12(7):1843–1863, 2000. doi:10.1063/1.870436.

[112] B. Franzelli, E. Riber, L. Y. Gicquel, and T. Poinsot. Large Eddy Simulation of combustion instabilities in a lean partially premixed swirled flame. *Combustion and Flame*, 159(2):621–637, February 2012. doi:10.1016/j.combustflame.2011.08.004.

[113] H. Weller, G. Tabor, H. Jasak, and C. Fureby. A Tensorial Approach to Computational Continuum Mechanics Using Object-Oriented Techniques. *Computers in physics*, 12:620–631, 1998. doi:10.1063/1.168744.

[114] A. J. Eder, C. F. Silva, M. Haeringer, J. Kuhlmann, and W. Polifke. Incompressible versus compressible large eddy simulation for the identification of premixed flame dynamics. *International Journal of Spray and Combustion Dynamics*, 15(1):16–32, 2023. doi:doi.org/10.1177/17568277231154204.

[115] G. E. Hinton. A practical guide to training restricted boltzmann machines. *Neural Networks: Tricks of the Trade: Second Edition*, pages 599–619, 2012.

[116] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

[117] W. Polifke. Black-box system identification for reduced order model construction. *Annals of Nuclear Energy*, 67:109–128, 2014.

[118] A. K. Tangirala. *Principles of system identification: theory and practice*. Crc Press, 2018.

[119] R. S. Blumenthal, P. Subramanian, R. Sujith, and W. Polifke. Novel perspectives on the dynamics of premixed flames. *Combustion and flame*, 160(7):1215–1224, 2013.

[120] B. Schuermans, H. Luebcke, D. Bajusz, and P. Flohr. Thermoacoustic analysis of gas turbine combustion systems using unsteady CFD. In *ASME Turbo Expo 2005: Power for Land, Sea, and Air*, pages 287–297. American Society of Mechanical Engineers, 2005.

[121] A. Blondé. *Thermoacoustic Dynamics of Gas Turbine Combustors: Effects of Hydrogen Blending and Tunable Acoustic Boundaries*. PhD thesis, ETH Zurich, Zürich, 2024.

[122] N. Tathawadekar, C. Silva, P. Sitte, and N. A. K. Doan. Physical quantities reconstruction in reacting flows with deep learning. In *INTER-NOISE and NOISE-CON Congress and Conference Proceedings*, volume 265, pages 1645–1656. Institute of Noise Control Engineering, 2023.

[123] The OpenFOAM Foundation. OpenFOAM-7. `https://openfoam.org/`, 2019.

[124] T. Poinsot and D. Veynante. *Theoretical and Numerical Combustion*. R.T. Edwards, Inc., Philadelphia, 2nd edition, 2005.

[125] E. Shelhamer, J. Long, and T. Darrell. Fully Convolutional Networks for Semantic Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(4):640–651, 2017. `arXiv:1605.06211`, `doi:10.1109/TPAMI.2016.2572683`.

[126] N. N. Tathawadekar, N. A. K. Doan, C. F. Silva, and N. Thuerey. Incomplete to complete multiphysics forecasting: a hybrid approach for learning unknown phenomena. *Data-Centric Engineering*, 4:e27, 2023.

[127] E. Kalnay. *Atmospheric modeling, data assimilation and predictability*. Cambridge university press, 2003.

[128] C. M. Rhie and W.-L. Chow. Numerical study of the turbulent flow past an airfoil with trailing edge separation. *AIAA journal*, 21(11):1525–1532, 1983.

[129] M. I. Zafar, M. M. Choudhari, P. Paredes, and H. Xiao. Recurrent neural network for end-to-end modeling of laminar-turbulent transition. *Data-Centric Engineering*, 2:e17, 2021.

[130] A. A. Wheeler, W. J. Boettinger, and G. B. McFadden. Phase-field model for isothermal phase transitions in binary alloys. *Physical Review A*, 45(10):7424, 1992.

[131] S. B. Pope. *Turbulent flows*. Cambridge university press, 2000.

[132] C. J. Chen. *Fundamentals of turbulence modelling*. Crc Press, 1997.

[133] F.-K. Benra, H. J. Dohmen, J. Pei, S. Schuster, B. Wan, et al. A comparison of one-way and two-way coupling methods for numerical analysis of fluid-structure interactions. *Journal of applied mathematics*, 2011, 2011.

[134] Z. Liu and Z. Liu. *Multiphysics in porous materials*. Springer, 2018.

[135] T. C. Lieuwen. *Unsteady Combustor Physics*. Cambridge University Press, 2012.

[136] A. Gruber, E. S. Richardson, K. Aditya, and J. H. Chen. Direct numerical simulations of premixed and stratified flame propagation in turbulent channel flow. *Physical Review Fluids*, 3(11):110507, 2018.

[137] M. Z. Jacobson. *Fundamentals of atmospheric modeling*. Cambridge university press, 1999.

[138] D. Rolnick, P. L. Donti, L. H. Kaack, K. Kochanski, A. Lacoste, K. Sankaran, A. S. Ross, N. Milojevic-Dupont, N. Jaques, A. Waldman-Brown, et al. Tackling climate change with machine learning. *ACM Computing Surveys (CSUR)*, 55(2):1–96, 2022.

[139] V. N. Gamezo, A. M. Khokhlov, E. S. Oran, A. Y. Chtchelkanova, and R. O. Rosenberg. Thermonuclear supernovae: Simulations of the deflagration stage and their implications. *Science*, 299(5603):77–81, 2003.

[140] G. Wanner and E. Hairer. *Solving ordinary differential equations II*, volume 375. Springer Berlin Heidelberg New York, 1996.

[141] H. N. Najm, P. S. Wyckoff, and O. M. Knio. A semi-implicit numerical scheme for reacting flow: I. stiff chemistry. *Journal of Computational Physics*, 143(2):381–402, 1998.

[142] O. M. Knio, H. N. Najm, and P. S. Wyckoff. A semi-implicit numerical scheme for reacting flow: Ii. stiff, operator-split formulation. *Journal of Computational Physics*, 154(2):428–467, 1999.

[143] T. Mitani. Propagation velocities of two-reactant flames. *Combustion Science and Technology*, 21(3-4):175–177, 1980.

[144] L. S. Pontryagin. *Mathematical theory of optimal processes*. CRC press, 1987.

[145] F. H. Harlow and J. E. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *The physics of fluids*, 8(12):2182–2189, 1965.

[146] P. J. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[147] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[148] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[149] J. Lampinen and A. Vehtari. Bayesian approach for neural networks—review and case studies. *Neural networks*, 14(3):257–274, 2001.

[150] E. Goan and C. Fookes. Bayesian neural networks: An introduction and survey. *Case Studies in Applied Bayesian Data Science: CIRM Jean-Morlet Chair, Fall 2018*, pages 45–87, 2020.

[151] L. V. Jospin, H. Laga, F. Boussaid, W. Buntine, and M. Bennamoun. Hands-on bayesian neural networks—a tutorial for deep learning users. *IEEE Computational Intelligence Magazine*, 17(2):29–48, 2022.

[152] J. Postels, F. Ferroni, H. Coskun, N. Navab, and F. Tombari. Sampling-free epistemic uncertainty estimation using approximated variance propagation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2931–2940, 2019.

[153] P. Goel and L. Chen. On the robustness of monte carlo dropout trained with noisy labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2219–2228, 2021.

[154] C. M. Bishop. Mixture density networks. 1994.

[155] T. Pearce, F. Leibfried, and A. Brintrup. Uncertainty in neural networks: Approximately bayesian ensembling. In *International conference on artificial intelligence and statistics*, pages 234–244. PMLR, 2020.

[156] Y. El-Laham, N. Dalmasso, E. Fons, and S. Vyetrenko. Deep gaussian mixture ensembles. *arXiv preprint arXiv:2306.07235*, 2023.

[157] W. Ji and S. Deng. ReacTorch: A differentiable reacting flow simulation package in PyTorch. 2020. URL: `https://github.com/DENG-MIT/reactorch`.

[158] JAX-reactor. URL: `https://github.com/comocheng/jax-reactor`.