Journal of Computational Physics 524 (2025) 113698

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

RF-PINNs: Reactive flow physics-informed neural networks for field reconstruction of laminar and turbulent flames using sparse data

Vikas Yadav *, Mario Casel, Abdulla Ghani

Data Analysis and Modeling of Turbulent Flows, Institute of Fluid Dynamics and Technical Acoustics, Technische Universität Berlin, Müller-Breslau-Straße 15, 10623 Berlin, Germany

ARTICLE INFO

Keywords:
Physics-informed neural networks
Reactive flow
Field reconstruction

ABSTRACT

Physics-Informed Neural Networks (PINNs) have emerged as a promising tool to model flow fields by embedding physical laws into neural networks and thereby reducing the dependency on data. While PINNs have shown substantial success in modeling non-reactive flows, their application for chemically reacting flows is less well understood. The overarching objective of this work is to propose reactive flow physics-informed neural networks (RF-PINNs) that reconstruct all flow quantities of interest solely based on sparse velocity profiles. We present RF-PINNs to reconstruct steady-state mean fields of premixed laminar and turbulent flames with sparse data. First, we extensively study a prototype flame to determine the best combination of governing equations to find a balance between prediction accuracy and optimization speed. We then demonstrate the RF-PINN prediction capabilities by utilizing measured velocity profiles to reconstruct the entire velocity, temperature, and density mean fields of laminar and turbulent flames. To highlight the bidirectional reconstruction capability, we utilize in the second step measured temperature profiles for both flames and, again, successfully reconstruct the flow fields of interest. For both scenarios, the subsequent comparison of predicted field quantities based on only 0.2% of the entire data set is sufficient for accurate field reconstruction. This study underscores the potential of RF-PINNs to complement comprehensive field data of laminar and turbulent reacting flows using both sparse and noisy experimental data.

Contents

1. Introduction . . . 2
2. Governing equations of reactive flows . . . 4
   2.1. Navier-Stokes equations for reacting flows . . . 4
   2.2. Non-dimensionalized Navier-Stokes equations . . . 5
3. Physics-informed neural networks . . . 6
4. Methodology of reactive flow PINNs . . . 7
   4.1. Training routine . . . 7
      4.1.1. Creation of lookup table . . . 7

* Corresponding author.
E-mail address: v.yadav@tu-berlin.de (V. Yadav).

https://doi.org/10.1016/j.jcp.2024.113698
Received 13 August 2024; Received in revised form 17 December 2024; Accepted 20 December 2024

Available online 27 December 2024

4.1.2. Reconstruction of field quantities . . . 8
4.2. Optimizer . . . 8
4.3. Hyperparameter tuning . . . 9
4.4. Choice of equations for RF-PINNs: a study on a 1D adiabatic flame . . . 9
4.4.1. Configuration . . . 9
4.4.2. RF-PINN model . . . 10
4.4.3. Prediction/analysis . . . 11
5. Application of RF-PINNs to a 2D laminar slit flame . . . 14
5.1. Multi-slit flame configuration . . . 14
5.2. RF-PINN model . . . 14
5.2.1. 2D equations and BCs . . . 14
5.2.2. Model architecture . . . 15
5.3. Prediction/analysis . . . 15
5.3.1. Case scenarios . . . 15
5.3.2. Model performance in case 1 . . . 15
5.3.3. Model performance in case 2 . . . 15
5.3.4. Model performance in case 3 . . . 15
6. Application of RF-PINNs to a turbulent flame . . . 17
6.1. Volvo flame configuration . . . 19
6.2. RF-PINN model . . . 19
6.3. Prediction/analysis . . . 20
6.3.1. Case scenarios . . . 20
6.3.2. Model performance in case 1 . . . 21
6.3.3. Model performance in case 2 . . . 21
7. Conclusion . . . 22
CRediT authorship contribution statement . . . 23
Declaration of competing interest . . . 23
Acknowledgement . . . 23
Data availability . . . 23
References . . . 24

1. Introduction

In the last ten years, data-driven modeling techniques have become increasingly valuable due to the rise of big data and the development of graphical processing units (GPUs). Machine learning (ML) approaches have proven successful in various fluid dynamics applications, as documented in [1–3].

Deep neural networks, a subset of ML, have shown promising results in assisting classical approaches like numerical simulations and experiments because of their capability to learn complex nonlinear dynamics [4]. This is demonstrated in, e.g., the work of Wang et al. [5], where they coupled a neural network with OpenFOAM to optimize the decision-making in high-dimensional fluid flow problems, whereas Pfaff et al. [6] introduced adaptive mesh discretization using graphical neural networks. On the other hand, Fukami et al. [7] used sparse sensor data and convolutional neural networks to recover spatial fields of unsteady wake flow and turbulence channel flow. Neural networks have also been combined with the data assimilation approach on a nonlinear two-layer quasi-geostrophic turbulent flow [8]. In the case of reacting flow simulations, where chemical kinetics play a major role, considerable progress has been made in employing machine learning techniques. For example, generative adversarial networks (GANs) are utilized to reconstruct the Reynolds stress subfilter scales from the large-scale large eddy simulation (LES) data in premixed turbulent reacting flows [9], as well as to generate planar methylidyne radical distributions in premixed turbulent flames [10]. Rywik et al. [11] proposed autoencoders to capture nonlinear spatial flame response to acoustic perturbations and produced laminar premixed flame shapes. Ihme et al. [12] offer comprehensive insights into the various machine-learning techniques, which are purely data-driven, employed in the field of combustion.

Despite the widespread application of NNs in various engineering fields, they often fail to generalize the underlying physics because of data sparsity: as in most technical systems, the amount of data is small and does not fulfill the requirement for NNs of large amounts of labeled data to train and optimize the models effectively. A new branch of artificial neural networks called physics-informed neural networks (PINNs) has recently emerged to address this issue. PINNs integrate physical laws into their optimization process to improve their generalization capability, while at the same time less data is required for model training. The origin of PINNs can be traced back to the work of Dissanayake and Phan-Thien [13], which introduced simple networks for solving the linear Poisson equation and thermal conduction problem with a nonlinear heat generation term. With the advancement of GPU technology, PINNs are now widely utilized to solve various engineering problems, many of which are summarized in [14–16] and application-specific to fluid mechanics are detailed in [17,18]. Following the advent of PINNs, they have been widely used in two ways for non-reactive flow problems: to solve forward problems without using any data [19–21] and inverse problems for parameter estimation [22] or missing field reconstruction [17,23–28].

Nomenclature

θ	Progress variable	Ψ	Neural network weights
c_p	Isobaric specific heat capacity	ϵ	Turbulent dissipation rate
$\mathbf{I}$	Identity tensor	ρ	Density
$\mathcal{D}$	Diffusivity	Φ	Equivalence ratio
Da	Damköhler number	$\mathcal{B}$	Boundary residual
b	Single neuron bias	$\mathcal{F}$	Governing equation residual
L	Length scale	$\mathcal{I}$	Initial residual
S	Flame speed	$\mathcal{L}_{\mathcal{B}}$	Boundary condition loss
u''	Turbulence intensity	$\mathcal{L}_{\mathcal{D}}$	Data loss
w	Single neuron weight	$\mathcal{L}_{\mathcal{F}}$	Governing equation loss
x, y	Spatial coordinates	$\mathcal{L}$	Initial condition loss
p	Pressure	$\mathcal{L}$	Total loss
Pr	Prandtl number	ADAM	Adaptive momentum estimation
T	Temperature	L-BFGS	Limited-memory Broyden–Fletcher–Goldfarb–Shanno
Re	Reynolds number		
Le	Lewis number	$(\cdot)_{\mathrm{k}}$	k^{th} species
t	Time	$(\cdot)_{\mathrm{t}}$	Turbulent
$\mathbf{u}$	Velocity	$(\cdot)_{\mathrm{F}}$	Fuel
$\mathbf{V}$	Diffusion velocity	$g(\cdot)$	Non-linear transformation
k	Turbulent kinetic energy	$(\cdot)^*$	Non-dimensional
U	Bulk velocity	$(\cdot)^{\mathrm{l}}$	l^{th} layer
Y	Mass fraction	$\widetilde{(\cdot)}$	Favre-filtered
λ	Heat conductivity	$(\cdot)^{\mathrm{T}}$	Transpose
$\dot{\omega}$	Reaction rate	$\widehat{(\cdot)}$	Prediction
μ	Dynamic viscosity	$\underline{(\cdot)}$	Temporal mean

The application of PINNs has expanded to include reactive flow problems in the last couple of years. Bode et al. [29] used the continuity equation to enforce physically plausible solutions of the reconstructed field in GANs, whereas Chi et al. [30] used an approximation equation of the heat release rate (HRR) as a constraint to reconstruct its spatial and temporal evolutions in premixed flames. Yadav et al. [31] introduced physics-informed long short-term memory (PI-LSTM) by incorporating the low-frequency limit for perfectly-premixed flames in LSTMs to reduce the direct numerical simulation (DNS) data requirement for dynamical systems analysis. PINNs have also been used to identify governing parameters of noisy thermoacoustic systems [32].

However, limited work can be found on the application of PINNs to solve forward problems in reactive flows, with the exception of [33], where a simple one-step global mechanism was used involving five chemical species. This may be attributed to the increase in the number of governing equations with the increase in species for detailed chemistry to capture the essential physics, which makes the optimization very challenging. For instance, if a reduced chemical mechanism for methane/air combustion with 13 species [34] would be employed, then 13 species equations would be required as constraints for the neural network. This significantly increases the problem's dimensionality, making the optimization difficult [35]. Moreover, as shown in the recent study of Grossmann et al. [36], PINNs are far from replacing well-established numerical methods in solving forward problems with the current machine learning infrastructure. Nevertheless, their effectiveness in solving inverse problems for tasks such as field reconstruction [37–39] is more relevant in engineering problems than solving forward problems, particularly in the case of chemically reacting flows. In this work, we address inverse PINNs for reactive flows based on sparse data that come from experiments, with the aim of providing a robust and efficient framework to reconstruct missing fields.

Reliable experiments and the recorded data are essential to understand the underlying physics of a system. However, experiments (in particular for reacting flows) require costly measuring devices and often provide partial insights due to limited optical accessibility or limitations in measuring a physical quantity. Hence, the determination of all relevant flow field data in the system can become difficult. In this context, the ability of PINNs to solve inverse problems is beneficial in such scenarios to complement experimental data because of their effectiveness to solve ill-posed problems [24,40]. Therefore, we set the following scenario as the motivation for this work: in experiments on reactive flows, particle image velocimetry (PIV) data can be measured in the domain of interest. These sparse data can then be used in PINN models to reconstruct missing data in unmeasured regions or reconstruct completely unknown fields, e.g. density and temperature field data in the entire domain.

Data sparsity is rarely addressed in the context of PINNs for reactive flows. Liu et al. [38] developed a PINN model using the momentum and energy conservation as governing equations. They relied on PIV and temperature sparse data, consisting of approximately 6400 to 16000 data points for each field, to reconstruct missing velocity and temperature data in the inaccessible region. Liu et al. [39] extended their previous model and combined it with a DeepONet [41] to predict soot. Utilizing the reconstructed temperature field and measurements of soot volume and hydroxyl (OH) radical, they predicted the soot-volume fraction distributions. However, in both studies, they did not address the PINN's ability to reconstruct fields based on *sparse* data. Here, we address the

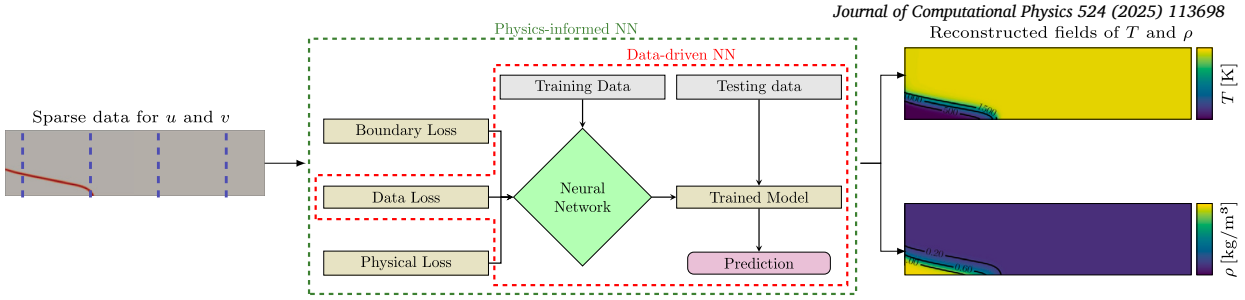

Fig. 1. Illustration of reconstruction via RF-PINNs with sparse data. One way to use RF-PINNs is to reconstruct the T and ρ fields based on sparse data, e.g. from PIV.

problem of sparse data and rely only on velocities profiles (given the difficulty in experimentally measuring temperature within the gas field) to reconstruct the remaining fields whose data are unavailable.

In this work, we propose reactive flow PINNs (RF-PINNs) to reconstruct steady-state mean fields of missing physical quantities using only sparse PIV data. This study does not focus on the prediction of the chemical species. Fig. 1 depicts our idea of using sparse data from the computational domain to reconstruct accurate mean flow fields. We propose a novel approach to reduce the dependencies on the species equations and instead use the progress variable (θ) equation encompassing the reaction rate term. This makes the neural network optimization faster because of fewer governing equations as physical constraints. The reaction rate term in the progress variable equations can either be closed using a surrogate NN model, as proposed in the case of the laminar flame (Sec. 5) or by using any available reaction rate model of choice, as tested in the case of the turbulent flame (Sec. 6). In the laminar case, we use a computationally cheap 1D chemical kinetics solver to build surrogate models to predict the reaction rate. This gives us the advantage of incorporating the detailed chemical kinetics of our choice into the progress variable equation. We further improve the efficiency of the model's convergence using a mapping approach (reaction rate as a function of progress variable) instead of explicitly using highly non-linear reaction rate equations in RF-PINNs. The idea resembles the flamelet modeling approach [42–47], where detailed chemistry is precomputed and stored in tables. With negligible research studies available on PINNs for reactive flows, the number of equations to be considered as physical constraints remains an open question. We also discuss this in our study with the help of a prototype 1D freely propagating flame and find a balance between accuracy and optimization speed.

The paper is structured as follows: Section 2 describes the Navier-Stokes (NS) equations, including the derivation of the non-dimensional form and the assumptions made. Section 3 briefly describes the fundamentals of PINNs and the general objective function used in RF-PINNs. Section 4 presents the methodology and guidelines for solving inverse problems with RF-PINNs, including a discussion on the choice of equations for PDE loss. The application of RF-PINNs to laminar and turbulent cases is elaborated in Secs. 5 and 6, respectively.

2. Governing equations of reactive flows

The goal of this study is to reconstruct flow quantities in reacting flows using sparse data by solving an inverse PINN problem. For this, we require physical constraints in the form of partial differential equations (PDEs) that govern reactive flows, which will be derived in this section following Poinsot and Veynante [48]. While the derivation of the governing equations is as general as possible, we will introduce further configuration-specific simplifications in the associated application sections of this study.

2.1. Navier-Stokes equations for reacting flows

We consider deflagration flames in this study. These are considered in the low-mach number limit and, therefore, modeled as incompressible, while viscous heating is negligible. Thus, we describe the multi-component, chemically reacting flow by the following conservation laws for mass, species, momentum, and energy:

$$\partial_t \rho + \nabla \cdot \rho \mathbf{u} = 0, \tag{1.1}$$

$$\rho \left(\partial Y_k + (\mathbf{u} \cdot \nabla) Y_k \right) + \nabla \cdot \rho \mathbf{V}_k + Y_k = \dot{\omega}_k, \tag{1.2}$$

$$\rho \left(\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla) \mathbf{u} \right) = -\nabla p + \nabla \cdot \mu \left(\nabla \mathbf{u} + (\nabla \mathbf{u})^{\mathrm{T}} - \frac{2}{3} (\nabla \cdot \mathbf{u}) \mathbf{I} \right), \tag{1.3}$$

$$\rho c_p \left(\partial_t T + (\mathbf{u} \cdot \nabla) T \right) = \dot{\omega}_T + \nabla \cdot \lambda \nabla T - \rho \nabla T \sum_{k=1}^{N} c_{p,k} Y_k \mathbf{V}_k, \tag{1.4}$$

where ρ, $\mathbf{u}$, p, T, μ, c_p, $\dot{\omega}_T$, λ, and $\mathbf{I}$ are the density, velocity, pressure, temperature, dynamic viscosity, specific heat capacity, heat release rate, heat conductivity, and identity matrix, respectively. Y_k, $\mathbf{V}_k$, $\dot{\omega}_k$ are the mass fraction, diffusion velocity, and the reaction rate of k^{th} species, respectively.

While the system of equations in Eq. (1) exactly describes deflagration flames in the low-Mach limit, it features a large dimensionality since, in addition to mass, momentum and energy conservation, N species transport equations need to be solved. This possibly poses a challenge for PINNs because a large number of PDEs makes the loss landscape very hard to optimize [35]. Therefore,

we derive a more convenient formulation that is based on a variable that globally describes the combustion progress (the so-called progress variable θ) and replace the N species transport equations. For deriving the progress variable equation, we make the following simplifying assumptions:

1. All species have the same molecular weight ($W = W_k$)
2. All species have the same constant heat capacity ($c_p = c_{p,k}$)
3. All species have the same molecular diffusion coefficient ($D = D_k$)
4. Assumptions 1-3 result in the same Lewis number of all species ($\mathrm{Le} = \mathrm{Le}_k$)
5. Species and heat diffusion become equal, therefore ($\mathrm{Le}_k = \mathrm{Le} = 1$)

Given these assumptions, we introduce a transport equation for the reduced fuel mass fraction Y

$$\rho\left(\partial_t Y + (\mathbf{u}\cdot\nabla)Y\right) - \nabla\cdot\rho D\nabla Y = \frac{\dot{\omega}_\mathrm{F}}{Y_\mathrm{F}^1}, \tag{2}$$

where Y_F^1 ($Y = Y_\mathrm{F}/Y_\mathrm{F}^1$) is the fuel mass fraction in the unburnt, fresh gas region and $\dot{\omega}_\mathrm{F}$ is its associated reaction rate.

In addition, we introduce the definition of the progress variable

$$\theta = \frac{T - T_1}{T_2 - T_1}, \tag{3}$$

which is transported by

$$\rho\left(\partial_t\theta + (\mathbf{u}\cdot\nabla)\theta\right) = -\frac{\dot{\omega}_\mathrm{F}}{Y_\mathrm{F}^1} + \nabla\cdot\frac{\lambda}{c_p}\nabla\theta, \tag{4}$$

where T_1 and T_2 are the minimum and maximum temperatures in the system, respectively, and thus $\theta \in [0, 1]$.

Adding Eqs. (2) and (4), while assuming unity Lewis number ($Le = 1 \rightarrow \lambda/c_p = \rho D$), yields:

$$\rho\left(\partial_t(\theta + Y) + (\mathbf{u}\cdot\nabla)(\theta + Y)\right) = \nabla\cdot\frac{\lambda}{c_p}\nabla(\theta + Y). \tag{5}$$

While Eq. (5) is free of any sources, it additionally can be shown that $(\theta + Y) = 1$, meaning that θ and Y are dependent quantities that are inversely related, e.g., when the progress variable increases, the fuel mass fraction decreases. Therefore, it is sufficient to keep only one equation such that the resulting set of describing equations reads:

$$\partial_t\rho + \nabla\cdot\rho\mathbf{u} = 0, \tag{6.1}$$

$$\rho\left(\partial_t\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{u}\right) = -\nabla p + \nabla\cdot\mu\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^\mathrm{T} - \frac{2}{3}(\nabla\cdot\mathbf{u})\mathbf{I}\right), \tag{6.2}$$

$$\rho\left(\partial_t\theta + (\mathbf{u}\cdot\nabla)\theta\right) = -\frac{\dot{\omega}_\mathrm{F}}{Y_\mathrm{F}^1} + \nabla\cdot\frac{\lambda}{c_p}\nabla\theta. \tag{6.3}$$

This system, again, describes a flow in the low-Mach limit where the combustion process is modeled by the progress variable θ and the corresponding reaction rate $\dot{\omega}_\mathrm{F}/Y_\mathrm{F}^1$. While the former is equivalent to the temperature, the latter requires an appropriate model to close the equations. This model is, however, fuel-specific and will be introduced in Sec. 4.1.

2.2. Non-dimensionalized Navier-Stokes equations

Data normalization is a crucial preprocessing step in any data-driven modeling framework. It modifies the loss landscape and enables faster model convergence, especially in supervised learning, where input and output features must be scaled to the same order. The use of PINNs to build inverse models presents a challenge to directly scale the values as the range of the unknown physical quantities is unavailable. To address this issue, one common approach is to non-dimensionalize the variables in the equations using their characteristic quantities. Many studies, such as those by Wang et al. [16], Jin et al. [49], Jagtap et al. [24], and Liu et al. [33], have shown this approach to be effective. In our case, we normalize all field variables in Eqs. (6) as follows.

- $\rho = \rho^*\rho_\mathrm{in}$, $\mathbf{u} = \mathbf{u}^* u_\mathrm{in}$, $p = p^*\rho u_\mathrm{in}^2$, where the subscript 'in' refers to the variable's state in the fresh gases, i.e., at the domain inlet
- the spatial coordinates are normalized by a domain characteristic length resulting in $\nabla = \nabla^*/L$
- the time is normalized by $t = t^* L/u_\mathrm{in}$

After substitution of these relations into Eqs. (6) and omitting the asterisk superscript, the non-dimensionalized equations read:

$$\partial_t\rho + \nabla\cdot\rho\mathbf{u} = 0, \tag{7.1}$$

$$\rho\left(\partial_t\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{u}\right) = -\nabla p + \nabla\cdot\mathrm{Re}^{-1}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^\mathrm{T} - \frac{2}{3}(\nabla\cdot\mathbf{u})\mathbf{I}\right), \tag{7.2}$$

$$\rho\left(\partial_t\theta + (\mathbf{u}\cdot\nabla)\theta\right) = -\frac{L}{\rho_\mathrm{in}u_\mathrm{in}}\frac{\dot{\omega}_\mathrm{F}}{Y_\mathrm{F}^1} + \nabla\cdot(\mathrm{Re}\,\mathrm{Pr})^{-1}\nabla\theta, \tag{7.3}$$

where $\mathrm{Re} = \rho_{\mathrm{in}} u_{\mathrm{in}} L/\mu$ and $Pr = c_p \mu/\lambda$ are the Reynolds and Prandtl number, respectively.

3. Physics-informed neural networks

Artificial neural networks have the ability to learn the underlying dynamics of a large class of non-linear systems [50]. Neurons are the basic computational units that build neural networks. Each neuron performs a single non-linear transformation of the linearly mapped inputs:

$$a = g\left(\sum_{i=1}^{n} w_i x_i + b\right), \tag{8}$$

where x_i are the inputs linearly mapped using weights w_i and bias b. The non-linear transformation $g(.)$ can take the form of sigmoid and hyperbolic tangent, among others, and are widely described [51,52]. Neurons can be stacked together in layers called multi-layer perceptron (MLP) to address complex non-linear behaviors. This is represented generally using $Z^{[l]} = g(W^{[l]} A^{[l-1]} + B^{[l]})$, where $W^{[l]} \in \mathbb{R}^{n_l \times n_{l-1}}$ represents the weight matrix of the l^{th} layer, $Z^{[l-1]} \in \mathbb{R}^{n_{l-1} \times n_{l-2}}$ represents the output from the $(l-1)^{\text{th}}$ layer and $B^{[l]}$ is the bias vector of the l^{th} layer. Each layer of neurons is connected to all the neurons of the next layer via weights, making the network fully connected, also called a feed-forward neural network (FFNN). In a purely data-driven supervised learning approach, the model is trained as a black box model, and the weights and biases are optimized until the objective function reaches a predefined stopping tolerance.

Although pure data-driven modeling is a widely used deductive learning technique, its performance depends on the accuracy, reliability, and relevance of the data. One common issue is that purely data-driven models may perform poorly when limited data is available. This leads to overfitting [53], where the model accurately fits the data but cannot generalize the underlying dynamics of the system well. To overcome this limitation, physical knowledge of the underlying system can be incorporated into the data-driven model, resulting in a new variant of the neural network called a physics-informed neural network (PINN). There are multiple ways to embed physics into the PINNs, such as using differentiable numerical simulations of physical systems to combine the approximation capabilities of deep learning with the accuracy of standard solvers. Tompson et al. [54] proposed this approach of integrating the solver in the training loop, which later found its application to 3D flow problems as shown in the work of Um et al. [55]. E and Yu [56] incorporated physics by minimizing the weak formulation of PDEs and approximating the test function used in the weak formulation through deep neural networks. Physics can also be integrated into the neural network by enforcing hard constraints before learning begins [57,58]. Another approach involves adjusting the loss function to include differentiable and algebraic equations as soft constraints, where violations are penalized during training. This method of providing soft constraints first presented in an early work of Dissanayake and Phan-Thien [13], and later popularized by Raissi et al. [19], is now widely used in many studies [59,14,24,31]. In this work, we employ the latter variant of incorporating physics as soft constraints, where the governing PDEs are incorporated in the loss term. The network's automatic differentiation ability computes the required derivatives of dependent variables at any point in the domain. Additionally, we design the network architecture such that it maps the necessary dependent variables (ρ in our study) to positive values. This restricts ρ to a positive solution space, thereby providing a hard constraint. In a forward Computational Fluid Dynamics (CFD) analysis, the problem is well-posed only when appropriate boundaries and initial conditions are specified. This information is also valuable to PINNs, so we increase the "physical content" of the loss function to incorporate the boundaries and initial conditions.

In order to understand how we incorporate the physical constraints in the model, we define a physical system over a domain $\Omega \subset \mathbb{R}^d$ having a solution $\boldsymbol{z} \in \mathcal{Z}$ constrained by:

$$\mathcal{F}[\boldsymbol{x}, t; \boldsymbol{z}(\boldsymbol{x}, t)] = 0, \quad \boldsymbol{x} \in \Omega, t \in [0, T], \tag{9}$$

$$\mathcal{B}[\boldsymbol{x}, t; \boldsymbol{z}(\boldsymbol{x}, t)] = 0, \quad \boldsymbol{x} \in \partial\Omega, t \in [0, T], \tag{10}$$

$$\mathcal{I}[\boldsymbol{x}, t; \boldsymbol{z}(\boldsymbol{x}, t)] = 0, \quad \boldsymbol{x} \in \Omega, t = 0. \tag{11}$$

Here, $\mathcal{F}$ is the residual of the governing equation consisting of differential operators of $\boldsymbol{z}$ as a function of spatial variables ($\boldsymbol{x}$) and temporal variable (t). $\mathcal{B}$ and $\mathcal{I}$ are the residual of the boundary conditions on the boundary ($\partial\Omega$) and the residual of the initial conditions, respectively. We employ an FFNN to approximate the solution vector $\hat{\boldsymbol{z}}(\boldsymbol{x}, t; \boldsymbol{\Psi})$, where $\boldsymbol{\Psi}$ are the trainable parameters (weights and biases) of the NN. Owing to the automatic differentiation ability of the NN, the residual functions ($\mathcal{F}$, $\mathcal{B}$ and $\mathcal{I}$) are approximated as well. The NN is optimized with the weights ($\boldsymbol{\Psi}$) such that the total loss ($\mathcal{L}$) is minimized:

$$\arg\min_{\boldsymbol{\Psi}} \mathcal{L}(\boldsymbol{\Psi}) = \lambda_f \mathcal{L}_f(\boldsymbol{\Psi} + \lambda_d \mathcal{L}_d(\boldsymbol{\Psi})) + \lambda_b \mathcal{L}_b(\boldsymbol{\Psi}) + \lambda_i \mathcal{L}_i(\boldsymbol{\Psi}), \tag{12}$$

where $\mathcal{L}_f$ is the residual loss for the governing PDE. $\mathcal{L}_d$, $\mathcal{L}_b$, and $\mathcal{L}_i$ are supervised loss terms for the sample data inside the domain, sample data from the boundary, and initial conditions, respectively. $\lambda_f, \lambda_d, \lambda_b$, and λ_i are their corresponding weights. Equations (13) - (16) represent the individual loss terms evaluated using the mean squared error (MSE):

$$\mathcal{L}_f(\boldsymbol{\Psi}) = \frac{1}{N_f} \sum_{j=1}^{N_f} |\mathcal{F}[\boldsymbol{x}_j^f, t_j^f; \hat{\boldsymbol{z}}(\boldsymbol{x}_j^f, t_j^f)]|^2, \tag{13}$$

$$\mathcal{L}_d(\mathbf{\Psi}) = \frac{1}{N_d}\sum_{j=1}^{N_d} |\hat{\boldsymbol{z}}(\boldsymbol{x}_j^{\boldsymbol{d}}, t_j^d) - \boldsymbol{z}(\boldsymbol{x}_j^{\boldsymbol{d}}, t_j^d)|^2, \tag{14}$$

$$\mathcal{L}_b(\mathbf{\Psi}) = \frac{1}{N_b}\sum_{j=1}^{N_b} |\mathcal{B}[\boldsymbol{x}_j^{\boldsymbol{b}}, t_j^b; \hat{\boldsymbol{z}}(\boldsymbol{x}_j^{\boldsymbol{b}}, t_j^b)]|^2, \tag{15}$$

$$\mathcal{L}_i(\mathbf{\Psi}) = \frac{1}{N_i}\sum_{j=1}^{N_i} |\mathcal{F}[\boldsymbol{x}_j^{\boldsymbol{i}}, t_j^i; \hat{\boldsymbol{z}}(\boldsymbol{x}_j^{\boldsymbol{i}}, t_j^i)]|^2, \tag{16}$$

where $\{x^f, t^f\}$, $\{x^d, t^d\}$, $\{x^b, t^b\}$, and $\{x^i, t^i\}$ are the training data to enforce the PDE loss, data loss, boundary loss, and initial condition loss, respectively. N_f, N_d, N_b, and N_i are the collocation points for evaluating the PDEs, number of available ground truths inside the domain, number of points on the boundaries, and for evaluating the initial conditions, respectively.

Optimizing only the supervised terms ($\mathcal{L}_d$, $\mathcal{L}_b$, $\mathcal{L}_i$) may lead to bad generalization in regions without data. In contrast, solely minimizing the residual from PDE ($\mathcal{L}_f$) might lead to convergence to a trivial solution. The tunable scaling weights, therefore, play a vital role in balancing the loss terms to accurately approximate the specific solutions of the training data while simultaneously capturing knowledge about the underlying PDE. The scalable weights, along with the number of layers, neurons, and activation functions of FFNN, are the hyperparameters that must be tuned. Failing to tune these hyperparameters correctly could prevent the model from converging to a global minimum, even in a simple non-reactive PINN problem. This can become more challenging when solving highly non-linear, multi-scale PDEs, as in the present study. There are several modifications proposed to the non-reactive PINN to address this issue. Nabian et al. [60], Wu et al. [61] suggest resampling of collocation points, while Jagtap et al. [62,63] propose adaptive activation functions to accelerate the convergence of the optimizer. McClenny and Braga-Neto [64] suggest a self-adaptive weighing scheme of the loss terms. These strategies work with varying degrees of success, and no strategy guarantees global convergence. Instead of experimenting with the above modifications, our work centers around developing training strategies to simplify combustion chemistry (Sec. 4.1) and manually tuning the important hyperparameters as described in Sec. 4.3.

4. Methodology of reactive flow PINNs

The primary purpose of RF-PINNs is to reconstruct missing fields using sparse data of reactive flows, as illustrated in Fig. 1. In this work, we constrain the neural networks with Eqs. (7) described in Sec. 2.2. Since we focus on steady-state reactive problems, we modify Eqs. (7) to neglect the unsteady terms and include steady-state conservation equations as physical constraints ($\mathcal{L}_f$) in PINNs. The governing equations are case-dependent (laminar and turbulent flame) and thus will be revisited in the later sections (Secs. 5 and 6). The RF-PINN network takes spatial coordinates as inputs and generates physical quantities such as ρ, $\mathbf{u}$, and T as outputs. RF-PINNs use a classical Feed-Forward Neural Network (FFNN) architecture with tunable hyperparameters, such as the number of layers of neurons, activation functions, type of optimizers, and their learning rate. This section will, therefore, shed light on the adopted training routine, optimization strategy, and the details of the imposed PDEs. The first part of the section covers details of the training algorithm used in the study, the preprocessing, and the model development steps with a focus on both laminar and turbulent flames. A discussion on the choice of optimizers and the hyperparameter tuning approach follows this. Finally, we delve into the requirements of the equations for RF-PINNs, introduce a custom loss function to include physics, address the best compromise in the choice of PDEs, and discuss the limit of the required data points.

4.1. Training routine

Applying any neural network to an engineering problem requires appropriate preprocessing steps like normalization (discussed in Sec. 2.2) and a structured training routine to ensure accurate model performance. This becomes more critical when only sparse data is available, as the network has a higher tendency to find a local minimum. In our work, we confine our solution space by incorporating soft and hard constraints. Nevertheless, this still does not guarantee global convergence. In this section, we discuss the RF-PINNs training routine to maximize the probability of reaching global convergence via a simple yet effective novel strategy.

In Sec 2.1, we already highlighted the issue of failure of PINNs for high-dimensional non-linear complex problems involving multiple PDEs. To overcome this, we opted to use the simplified progress variable equation (Eq. (7.3)). As described in Sec. 2, incorporating the progress variable equation introduces the reaction rate term, which needs to be closed to obtain a unique solution. Various available reaction rate models are highly non-linear (like the Arrhenius law [48] for the laminar case). Explicitly using these models leads to slower training and decreases the probability of reaching a global minimum. To overcome these issues, we utilize an approach of tabulating the reaction rate term as a function of θ. With regard to this, we divide our training process into two stages. In the first stage, we prepare a lookup table to learn the reaction rate mapping, and the second stage focuses on training of RF-PINNs.

4.1.1. Creation of lookup table

The idea to create a precomputed lookup table for the reaction rate resembles the flamelet modeling approach [42–44], where detailed chemistry is precomputed and stored in tables. This allows us to estimate accurate chemical kinetics at a low computational cost. As enumerated in Poinsot and Veynante [48], there are various modeling approaches available to create a lookup table by varying the progress variable θ and tabulating the corresponding reaction rate $\dot{\omega}$. In RF-PINNs, we adopt slightly different strategies depending on the flame configuration (laminar and turbulent), as discussed below.

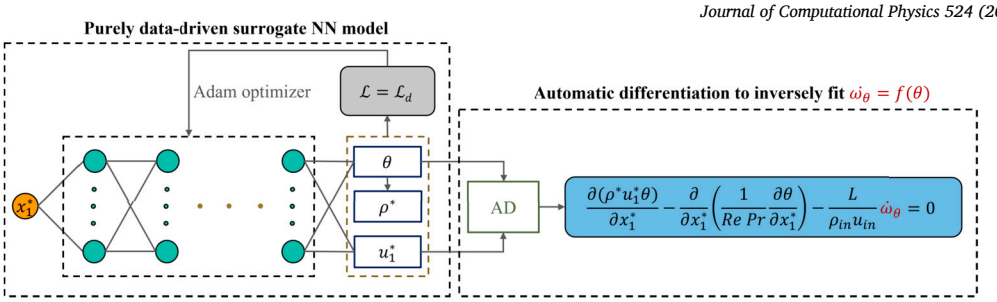

Fig. 2. Surrogate NN model for the laminar case to obtain accurate reaction rate as a function of progress variable based on 1D Cantera simulations.

Laminar case:
As introduced at the beginning of this section, we can use the existing models, like the Arrhenius model, to create a lookup table for the laminar case. This is done by varying the progress variable and tabulating the corresponding reaction rate. Although this may seem simple, the reaction rate models have parameters that require significant effort to fine-tune to accurately reflect the correct chemical kinetics of the problem under investigation. To overcome this issue, we use a computationally efficient 1D chemical kinetics solver. We perform 1D freely propagating adiabatic flame simulations with a chosen detailed chemistry in an open-source solver Cantera [65], which computes the density (ρ), velocity (u), and temperature (T) fields. While the solver provides information on individual species' production and destruction reaction rates, it can be challenging to use the solver to compute the global reaction rate needed in the progress variable equation. The net production and destruction reaction rate of different species have to be combined in the ratio of species mass fraction at different spatial locations. To solve this problem, we use a data-driven neural network to learn the predicted quantities of the 1D simulation and obtain the reaction rate as a function of θ. We use automatic differentiation to compute the partial derivatives of the predicted quantities required in the progress conservation equation (Eq. (7.3)). Fig. 2 illustrates this process to inversely fit $\dot{\omega}$ for the given reference values of u, ρ and θ. After inversing Eq. (7.3), we estimate the reaction rate and tabulate one reaction rate for every θ. This approach serves two purposes: first, we do not rely on a complex reaction rate equation, and second, we can use any chemical kinetics scheme and estimate the reaction rate for any fuel based on the values of u, ρ, and θ. While solving Eq. (7.3) inversely, relying purely on data could lead to overfitting and unphysical results (negative $\dot{\omega}$). In order to rectify this, we can additionally impose mass conservation in the loss function of the surrogate model ($\rho \cdot u =$ constant), leading to a smooth fit of $\dot{\omega}$.

Turbulent case:
We do not employ the same strategy of solving Eq. (7.3) inversely for turbulent flames because the Cantera simulations apply only to laminar cases. Therefore, we compute the reaction rate directly using the appropriate model as detailed in Poinsot and Veynante [48]. In our work, we compute the reaction rate using the Eddy Break-Up (EBU) model (Eq. (17)):

$$\overline{\dot{\omega}}_\theta = C\frac{\epsilon}{k}\overline{\rho}\widetilde{\theta}(1-\widetilde{\theta}). \tag{17}$$

k and ϵ are the turbulent kinetic energy and its dissipation rate, respectively, while C is a model constant, typically in the order of unity. $\bar{\rho}$ and $\widetilde{\theta}$ are the temporal average of density and Favre-averaged progress variable, respectively (detailed in Sec. 6).

4.1.2. Reconstruction of field quantities

In the second stage, we set up the data in the input and output format for the supervised model training. The dataset is non-dimensionalized, and the governing equations are formulated in their non-dimensional form (Eqs. (7)). We create a point cloud of independent variables (collocation points N_f) to evaluate Eqs. (7). Like grid-based numerical simulation, we increase the number of collocation points in regions with, e.g., strong gradients. This helps the model to generalize well and generate smooth solutions. In our study, these regions are the areas where the fuel-air mixture burns.

The total loss is evaluated using Eq. (12). Since we only use the steady-state equations, λ_i in Eq. (12) is set to 0. The scalar weights (λ_d, λ_f, and λ_b), in the objective function are used to scale each loss term to improve the training convergence. It has been reported numerous times that training may stall if these individual weights are not set properly [66]. While some papers propose dynamic approaches to adjust these weights, they are not generally applicable and still require tuning [16]. This work does not focus on addressing the weight distribution issue directly. We leave this part out of the scope of our current work and set constant values for these weights. The general approach sets a high weight on known boundary values and field data, and our study follows the same weighting strategy ($\lambda_d, \lambda_b > \lambda_f$). The low weights on the PDE losses are also justified in our case since the reaction rate model in the progress variable equation is a simplified representation compared to (i) the detailed kinetics of the laminar case (Sec. 5) and (ii) the experimental data of the turbulent case (Sec. 6).

4.2. Optimizer

In a purely data-driven model, the Adam's (Adaptive momentum) algorithm has proven to be very effective in quickly converging to an accurate solution. For PINNs, Liu et al. [67] demonstrated that the combination of the L-BFGS (Limited-memory Broyden–Fletcher–Goldfarb–Shanno) algorithm and the classical Adams algorithm is highly effective in improving the convergence rate of the

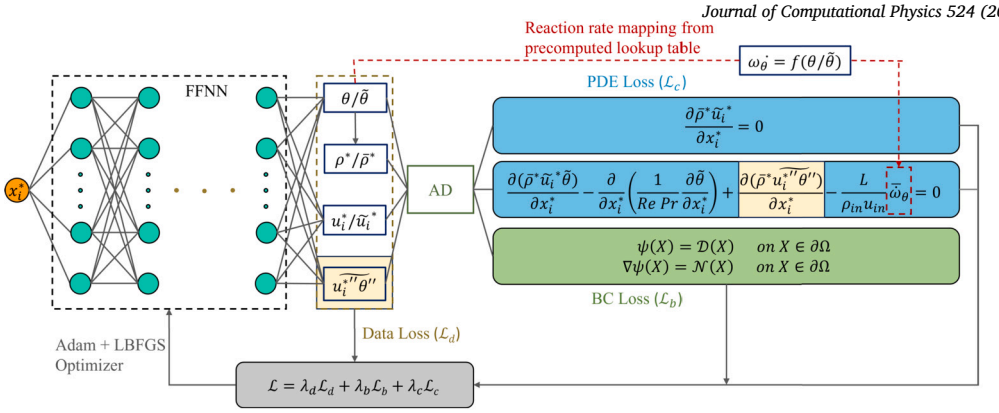

Fig. 3. RF-PINNs framework: Schematic of the neural network architecture, the governing equations, and the combined objective function used model training. In the laminar case, three quantities are predicted, and in the turbulent case, scalar flux terms are additionally predicted (highlighted in yellow). (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

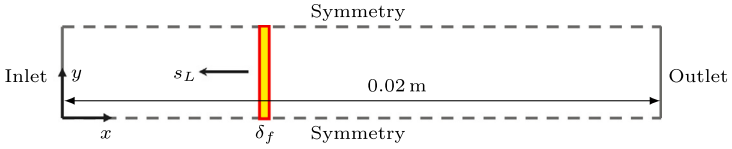

Fig. 4. Sketch of 1D freely propagating flame computational domain.

total training error. To achieve similar results, we have adopted a comparable approach, training the models for more iterations of the L-BFGS than the Adams optimizer. The model is trained for m Adams iterations using a decreasing learning rate, followed by n L-BFGS iterations. The values of m and n vary depending on the complexity of the problem. The Adam optimizer is employed with an adaptive learning rate until it reaches a plateau, after which the L-BFGS optimizer takes over. In the rest of the paper, we will refer to the Adam optimizer as optimizer A and the L-BFGS optimizer as optimizer B.

4.3. Hyperparameter tuning

For our study, we employed the Feed-Forward Neural Network (FFNN), which consists of layers of neurons that propagate information in a single direction. The key hyperparameters are the number of layers, the number of neurons, and the activation functions. Tuning these parameters is an iterative process, and it is important to have a strategy rather than optimize all hyperparameters simultaneously. Jagtap and Karniadakis [68] summarize the importance and selection of activation functions for model convergence. Based on their findings, we chose the hyperbolic tangent function as the activation function, which is suitable for our problem because of the jump in the values of physical quantities in the domain due to the combustion process. The choice of the number of layers and neurons is an iterative process, where more complex problems require a higher number of trainable parameters. This is achieved by increasing the number of neurons and reducing the number of layers or vice versa. However, deeper layers are generally more effective in learning complex behavior, as demonstrated in [69]. In our study, we test model convergence for a range of layers, each with 30 neurons.

The general workflow of our RF-PINN framework is visualized using Fig. 3 and depicts predictions and equations for both the laminar and the turbulent cases, each of which will be tackled in Secs. 5 and 6.

The methodology of RF-PINN described in this section for laminar and turbulent cases is also summarized in Algorithm 1.

4.4. Choice of equations for RF-PINNs: a study on a 1D adiabatic flame

Section 2.1 motivates the use of the progress variable equation instead of multiple species and energy equations. This is further supported by the recent work of Liu et al. [39], where, although they utilized species, momentum, and temperature equations to constrain the model, they resorted to a simple chemistry model to reduce the computational cost. In Sec. 2, we introduced the non-dimensional form of the governing equations (Eqs. (7)) for RF-PINNs. However, it raises the question: Do we need all three PDEs for the steady-state applications or can we further reduce the dependencies on equations? This section tests the influence of these equations on a prototype case of a 1D freely-propagating adiabatic flame configuration. Since the computation cost of this setup is low, we extensively study training and testing RF-PINN models employing different combinations of the governing equations.

4.4.1. Configuration

We consider a premixed methane-air mixture with an equivalence ratio of 0.8 operated at atmospheric pressure ($p = 1$ bar). The inlet flow velocity is 0.26 m s^{-1} at a temperature of $T = 293$ K. The length of the domain is $L = 0.02$ m. Fig. 4 depicts the configuration

Algorithm 1 Training pipeline of RF-PINN.

Require: Inlet ρ, Inlet $\boldsymbol{u}$, Inlet T, $\mathcal{B}$

Creation of lookup table

switch flowtype **do**
  case laminar
    Simulate 1D flame using Cantera simulation for the given parameters
    Define purely data-driven MLP to learn the simulated values
    for $k \leftarrow 1$ to $Epoch$ **do**
      Predict θ, u
      Calculate ρ based on Gas law
      Compute data loss: ($\mathcal{L}_d(\boldsymbol{\Psi})$)
      Update weights ($\boldsymbol{\Psi}$)
    end for
    Estimate $\dot{\omega}_\theta$ using automatic differentiation to fit $\rho\mathbf{u}\cdot\nabla\theta = -\frac{L}{\rho_{\text{in}}u_{\text{in}}}\dot{\omega}_\theta + \nabla\cdot(\text{Re Pr})^{-1}\nabla\theta$
  case turbulent
    Choose a model (for e.g. EBU: Eq. (17))
    Compute $\dot{\omega}_\theta$ for $\theta \in [0, 1]$
Tabulate the reaction rate as a function of progress variable ($\dot{\omega}_\theta = f(\theta)$)

Reconstruction of fields

Create input-output data using boundary data and given sparse domain data
Define collocation points (N_f) in the computational domain
Define #$epochs$ for ADAM ($Epoch_{Adam}$) and L-BFGS ($Epoch_{L-BFGS}$) optimizers
Define scaling factors for each loss (λ_d, λ_f, λ_b)
Initialize the PINN model with the random weights.
for $k \leftarrow 1$ to $Epoch_{Adam}$ **do**
  Predict θ, u, v
  Calculate ρ based on gas law
  Determine $\dot{\omega}_\theta$ from lookup table
  Compute total loss $\mathcal{L}(\boldsymbol{\Psi}) = \lambda_d\mathcal{L}_d(\boldsymbol{\Psi}) + \lambda_f\mathcal{L}_f(\boldsymbol{\Psi}) + \lambda_b\mathcal{L}_b(\boldsymbol{\Psi})$;
  Update weights ($\boldsymbol{\Psi}$)
end for
Load weights ($\boldsymbol{\Psi}$) for L-BFGS optimizer
for $k \leftarrow 1$ to $Epoch_{L-BFGS}$ **do**
  Repeat Step 10 until convergence
end for

sketch of the 1D freely propagating flame, where δ_f denotes the flame thickness and s_L is the laminar flame speed. The simulations are carried out in Cantera using GRI-mech (Frenklach et al. [70]), which is a detailed kinetic mechanism for methane/air combustion with 53 species and 325 reactions. The laminar flame speed corresponds to $0.26\,\text{m}\,\text{s}^{-1}$, which generates a steady-state flame. This aligns with the idea to reconstruct mean flow and flame fields.

4.4.2. RF-PINN model

1D equations and BCs:

As the considered configuration is one-dimensional, the velocity only features one component ($\mathbf{u} = [u, 0, 0]^{\text{T}}$) in the streamwise direction (x). In addition to this, all field variables $\phi \in \{\rho, \mathbf{u}, p, \theta, \dot{\omega}_\theta\}$ are assumed to be homogeneous in the remaining directions (y, z), such that the associated derivatives in these directions are zero ($\partial_y\phi = \partial_z\phi = 0$). The system of PDEs (Eqs. (7)) is simplified to a steady-state one-dimensional formulation:

$$\frac{\partial \rho u}{\partial x} = 0, \tag{18.1}$$

$$\rho u\frac{\partial u}{\partial x} = -\frac{\partial p}{\partial x} + \frac{\partial}{\partial x}\left(\text{Re}^{-1}\frac{4}{3}\frac{\partial u}{\partial x}\right), \tag{18.2}$$

$$\rho u\frac{\partial \theta}{\partial x} = -\frac{L}{\rho_{\text{in}}u_{\text{in}}}\frac{\dot{\omega}_{\text{F}}}{Y_{\text{F}}^1} + \frac{\partial}{\partial x}\left((\text{Re Pr})^{-1}\frac{\partial \theta}{\partial x}\right). \tag{18.3}$$

We introduce boundary conditions to define the operating conditions (Table 1). In this case, they only apply to two points: the locations of $x = 0$ and $x = L$. While the fresh gas conditions are set via the density and the inlet velocity, the associated conditions on the progress variable at the inlet and the outlet ($\theta = 0$ and $\theta = 1$ respectively) constrain the fresh and burnt gases.

Model architecture:

Before the model training, it is essential to create a training dataset to incorporate the known data information and the boundary conditions. Since we are working with a 1D steady-state problem, the input to RF-PINN consists of one feature (spatial coordinate x at 522 locations). The output consists of dependent field variables (i.e., 522 data points) for θ, u, and ρ. As described in Sec. 4.1.2, we vary the number of layers in our FFNN architecture, and the other hyperparameters stay the same for all the applications. After some tuning, six layers of 30 neurons each proves sufficiently complex for a low MSE of the order 10^{-3}. We train the model for 100 epochs of optimizer A and 10000 epochs of optimizer B and use 1000 collocation points to evaluate the PDE loss ($\mathcal{L}_f$). We use relatively

Table 1
Boundary conditions for the 1D flame.

Variable	Location	Value
ρ	Inlet	$1.1435\,\mathrm{kg\,m^{-3}}$
u	Inlet	$0.26\,\mathrm{m\,s^{-1}}$
θ	Inlet	0.0
θ	Outlet	1.0

Table 2
Definition of test cases for various combinations of PDEs to evaluate the best choice in terms of accuracy and training cost for RF-PINNs (avg. runtime = 25 min).

Case	Field for $\mathcal{L}_D$	No. of data points	PDEs used
1	u	522	Mass, momentum and progress
2	u	1	Mass, momentum and progress
3	u	1	Mass and momentum
4	u	1	Momentum and progress
5	u	1	Mass and progress
6	u	1	Mass
7	u	522	Mass
8	θ	1	Mass and progress

high collocation points (500) in the region of the flame ($0.006 < x < 0.008$) to capture the high gradient information accurately and avoid overfitting. Since we have only two boundary points (Table 1), scaling the boundary and data loss terms is necessary. We used a factor of 1000 to scale the boundary and data loss terms to match the order of the number of collocation points used. This is justified because the data have higher fidelity than the simplified equations we use for RF-PINNs. Owing to the simplified progress variable equation, the PDE loss will always have an error by definition. Consequently, in our training, we do not expect the PDE loss to decrease continuously with increasing epochs but rather to reach a plateau.

4.4.3. Prediction/analysis

Case scenarios:

We test the RF-PINN framework with known field data for the data loss term during training while reconstructing fields of unknown physical quantities. These known fields are analogous to the scenarios where the experimental data are available, but in this 1D configuration (Table 1), we instead use the Cantera simulation data. The Cantera simulation generates field data at 522 spatial points, which are unevenly distributed. The points are denser in the flame front region and sparser in other areas. Table 2 enumerates these scenarios that we test for both dense and sparse data with different combinations of conservation equations. For the first 7 cases, we provide only velocity as available data while training the network and provide the temperature field data for case 8. The tests are conducted on a single GPU from a cluster of GPUs with varying computing capabilities. An average[1] of 25 minutes is needed to run the cases tabulated in Table 2.

Model performance in cases 1 and 2:

Fig. 5 summarizes the performance of scenarios where all three conservation equations are provided to the model, showing the evolution of the loss curves and prediction of missing density and temperature fields. Case 1 (Figs. 5a,c) shows the dense case providing the entire velocity field data (522 data points) to RF-PINN. Here, the mass, momentum, and progress conservation equations are evaluated for $\mathcal{L}_f$ as physical constraints. The network is relatively easy to learn and reaches the MSE of 10^{-3}. Correspondingly, the predictions exactly match those from the reference solution (Fig. 5c). However, it is not always possible to have dense data and as stated in the introduction, we explore the lower limit bound of provided data. Therefore, in case 2, we challenge the model by providing sparse velocity data (only one data point) but constrain the network with all three equations. We observe that the model still performs well in predicting the reconstructed fields accurately (Fig. 5d). The MSE of the total loss is in the same order (10^{-3}) as that in case 1. The data loss curve in Fig. 5b shows many fluctuations, which is expected as the network tries to fit a single data point.

Model performance in cases 3, 4 and 5:

In the following scenarios (cases 3 to 5), we consider two equations at a time and give sparse data to the model. In case 3, we neglect the progress variable conservation equation and train the RF-PINN network on the other two equations. The trained model for case 3 is slightly inaccurate in the region of the flame front as seen in Fig. 6d although the loss converges to a low value of 10^{-3} (Fig. 6a). This could be because of the missing progress variable equation, which provides information on the flame thickness. In case 4, we provide momentum and progress conservation equations. The trained model performs worse than case 3. Even though the total loss falls within the range of 10^{-3} (Fig. 6b) and the model accurately predicts the given data point (Fig. 6e), it fails to generalize in other

[1] We state the average run time since the computations were not performed on a specific GPU type.

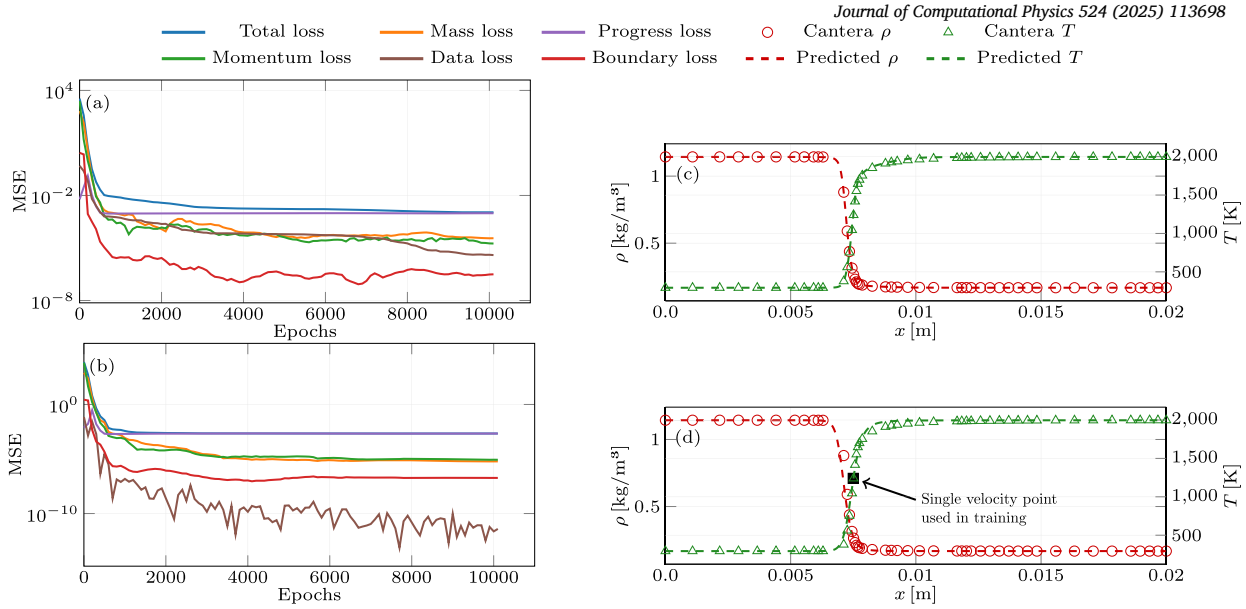

Fig. 5. Evolution of loss curves (left), temperature and density predictions (right) for cases 1 to 2 (top to bottom).

regions, implying the presence of a non-unique solution. This also indicates a violation of the mass conservation equation in the region $0.005 < x < 0.013$. For case 5, we enforce mass and progress conservation during the training process. The total loss reaches the order of 10^{-3} after 10000 epochs (Fig. 6c). The predictions align with those of the reference solution (Fig. 6f), showcasing that the combination of mass and progress conservation with sparse data leads to accurate physical solutions. This also demonstrates that model 5, with fewer physical constraints, performed as accurately as models 1 and 2.

Model performance in cases 6 and 7:

In the next couple of tests (6 and 7), we impose only mass conservation during the training process but vary the amount of training data. For case 6, with a single velocity data point, the total loss is as low as 10^{-6} after 10000 epochs as seen in Fig. 7a. However, the predictions do not match those from the reference solution (Fig. 7c). Imposing just mass conservation with sparse data leads to many solutions (non-unique), and the optimizer may not always reach the desired solution. On the other hand, by providing dense (here, 522) data points, we can make accurate predictions of the missing fields (Fig. 7d). Nevertheless, it can be observed in Fig. 7b that the convergence process is slow, and the model reaches a total loss of 10^{-2} after 10000 epochs.

Model performance in case 8:

In the previous test cases of combining various conservation equations, we concluded that mass and progress conservation equations perform for sparse data. We perform a final test where we provide sparse temperature information to the model instead of using velocity data and use again only mass and progress conservation equations for the evaluation of the PDE losses. This strategy aligns with the common scenario of using experimental data (availability of either sparse velocity fields or temperature fields). The evolution of the loss curves is depicted in Fig. 8a. The plot shows that all the losses converge to low values and reach a plateau after 9000 epochs. The total loss reduces to 10^{-2} along with the mass and progress conservation loss. The data loss is in the order of 10^{-5}. Since there are just two boundary points, the network learns them quickly, leading to an MSE of 10^{-8}. Correspondingly, very accurate predictions can be seen of the reconstruction of the unknown ρ and u fields in Fig. 8b.

Note that the value of the total loss for a specific case can not be considered for the quality of the prediction: (i) in the context of sparse data, multiple solutions may be found (see, e.g. case 6), which indicate a low error on the data loss but fails to reconstruct the missing fields. Hence, these solutions need to be ruled out by providing accurate constraints such as the loss on the progress variable; (ii) the total loss sums the individual loss contributions (Eq. (12)) coming, e.g. from PDEs or BCs and, hence, will be nominally higher than the total loss of a case, which only provides data and no other constraints.

In this section, we performed tests on a 1D prototype flame, exploring different combinations of equations with RF-PINN to find the best compromise of computational speed and accuracy. Our findings indicate that when dealing with limited data, using only a mass conservation equation or a combination of momentum and progress variable, mass and momentum conservation equations lead to non-unique solutions. However, we can achieve a unique solution by incorporating all three conservation equations or using a combination of mass and progress variable equations. The latter is relatively cost-effective due to the absence of the momentum equation. Therefore, we will exclusively utilize these two equations for all subsequent analyses in this study.

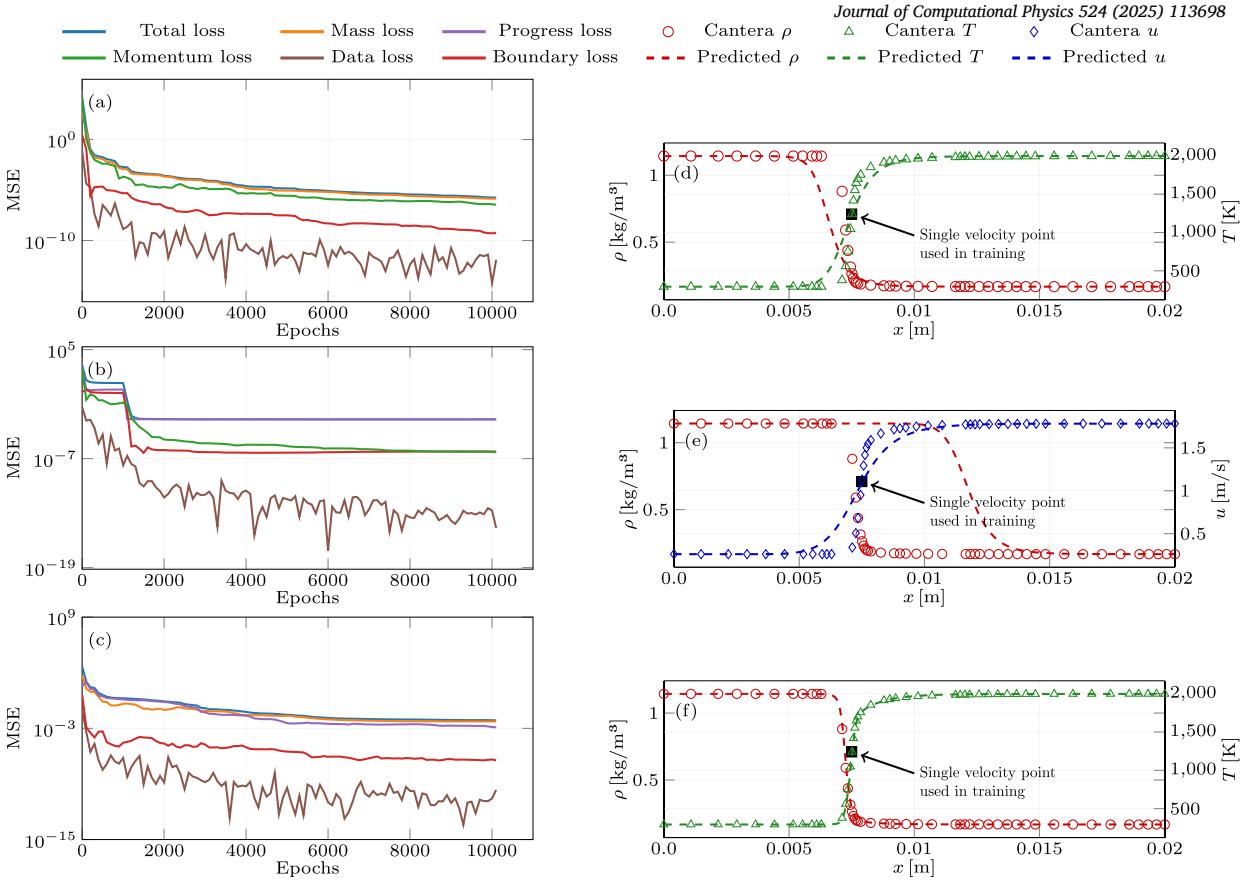

Fig. 6. Evolution of loss curves (left), temperature and density predictions (right) for cases 3 to 5 (top to bottom).

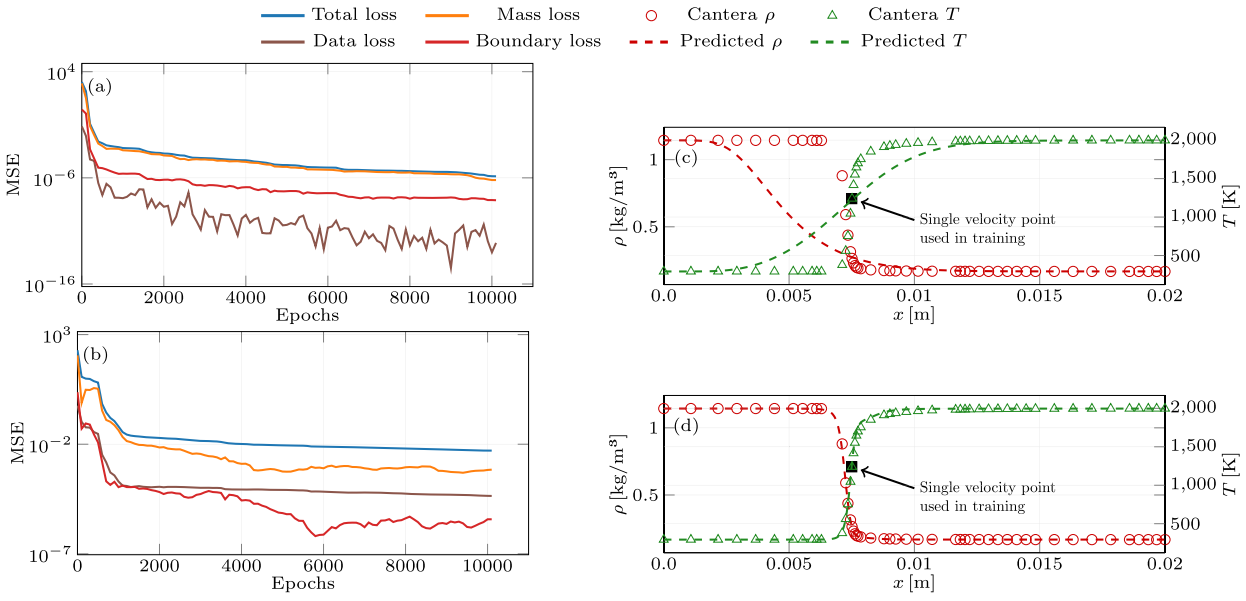

Fig. 7. Evolution of loss curves (left), temperature and density predictions (right) for cases 6 to 7 (top to bottom).

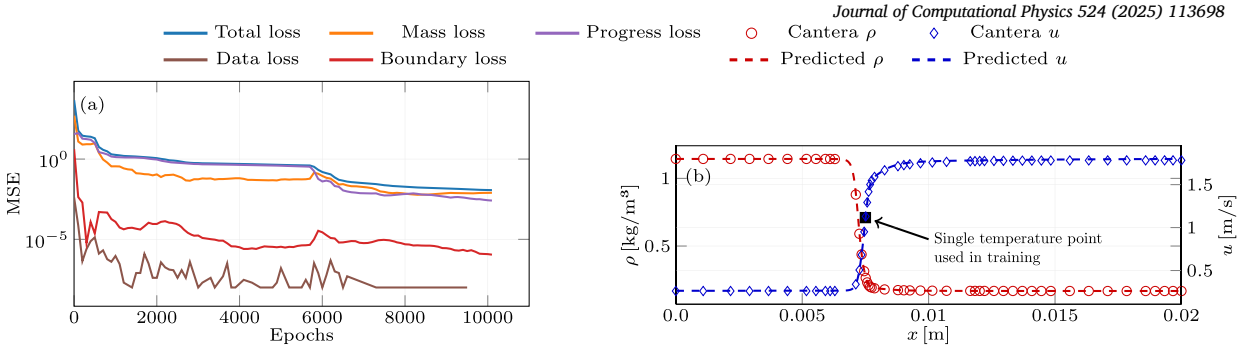

Fig. 8. Evolution of loss curves (left), velocity and density predictions (right) for case 8.

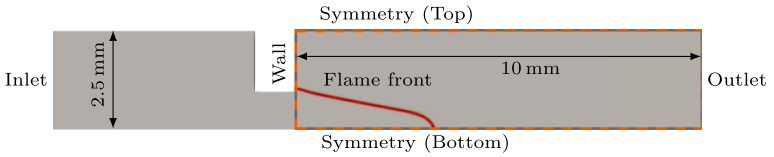

Fig. 9. Sketch of the 2D multi-slit premixed methane-air configuration. The red frame shows the computational domain for RF-PINNs.

Table 3
Boundary conditions for the 2D multi-slit flame.

Variable	Location	Value
u	Wall	$0.0\,\mathrm{m\,s^{-1}}$
u	Inlet	Parabolic velocity profile
v	Inlet + Top + Bottom + Wall	$0.0\,\mathrm{m\,s^{-1}}$
θ	Inlet	0.0
$\partial\theta/\partial n$	Top + Wall	0.0

5. Application of RF-PINNs to a 2D laminar slit flame

After extensively testing RF-PINNs on 1D flame configurations, we test the framework on a 2D laminar flame and analyze a premixed multi-slit configuration. The configuration, the modified equations, and the changes to the architecture are detailed below.

5.1. Multi-slit flame configuration

The experimental setup corresponds to the multi-slit burner investigated by Kornilov et al. [71] and has been widely used in other numerical studies (Haeringer et al. [72], Ghani and Polifke [73], Casel and Ghani [74]). The setup is shown in Fig. 9. A premixed methane-air of equivalence ratio 0.8 enters the domain from the left with a uniform velocity of $0.4\,\mathrm{m\,s^{-1}}$. The entire domain of length 0.016 m consists of a plenum, an area jump, and the combustion chamber where the flame stabilizes downstream. The combustion chamber has a length of 0.01 m. The inlet temperature is set to 293 K, and the adiabatic wall condition is maintained at the plate on which the flame stabilizes. Two-dimensional direct numerical simulations (DNS[2]) were performed using the fully compressible solver *AVBP* developed by Cerfacs/IFPen [75–79]. Chemical kinetics were modeled by a two-step chemical scheme of Franzelli et al. [80]. Since the reaction occurs in the combustion chamber, we analyze this region using the RF-PINN methodology (red frame in Fig. 9).

5.2. RF-PINN model

5.2.1. 2D equations and BCs

The governing equations for this configuration are the steady-state equations of Eqs. (7). Therefore, the implemented PDEs in the loss function of RF-PINNs, in this case, are very similar to those used in the 1D case (Sec. 4.4), with one exception: The equations here govern a two-dimensional flow, where only the third direction features a zero velocity and homogeneous scalar conservative variables. The modified equations are:

$$\frac{\partial \rho u}{\partial x} + \frac{\partial \rho v}{\partial y} = 0, \tag{19.1}$$

[2] DNS in our context resembles flame resolved simulation.

Table 4
Cases for testing RF-PINN on 2D multi-slit flame (avg. runtime = 9 h).

Case	Field for $\mathcal{L}_D$	No. of data points
1	u and v	Full (75000)
2	u and v	Sparse (140)
3	θ	Sparse (140)

$$\rho u\frac{\partial\theta}{\partial x}+\rho v\frac{\partial\theta}{\partial y}=-\frac{L}{\rho_{\text{in}}u_{\text{in}}}\frac{\dot{\omega}_{\text{F}}}{Y_{\text{F}}^{1}}+\frac{\partial}{\partial x}\left(\text{Re Pr}^{-1}\frac{\partial\theta}{\partial x}\right)+\frac{\partial}{\partial y}\left(\text{Re Pr}^{-1}\frac{\partial\theta}{\partial y}\right). \tag{19.2}$$

The boundary conditions imposed while performing DNS and correspondingly used in training the RF-PINN model are shown in Table 3. Unlike the 1D case, where the boundary conditions included only Dirichlet conditions, we also impose Neumann boundary conditions to restrict the solution space and avoid local minimum.

5.2.2. Model architecture

Since the problem features a higher complexity than the 1D case, we cannot use the same architecture. This is a typical high-bias characteristic of neural networks, where the model underperforms since the provided layers of neurons are insufficient to learn the flow fields. We tuned the neural network layers and increased them from 6 to 10. The other hyperparameters, such as the number of neurons, activation functions, normalization range, and optimization parameters, were kept constant. Since this flame is a 2D problem, the input to the network consists of x and y coordinates encompassing the boundaries and the available field data. The output constitutes velocity components (u and v), θ, and ρ. We use 10000 collocation points to evaluate the PDEs' loss in the computational domain. The models run for 10000 epochs with optimizer A and 50000 epochs with optimizer B until the total loss converges to a sufficiently low value. Similar to the 1D case, we use the factor of 1000 to prioritize the boundary and data losses.

5.3. Prediction/analysis

Similar to the 1D case, we devise multiple cases ($\mathbf{u}$ as data and θ as data) and also investigate the lower limit of provided data before the RF-PINN model fails. To validate the RF-PINN predictions, we use the DNS simulation data [73,74].

5.3.1. Case scenarios

A high number of data points (≈ 75000) is available in the entire domain, which can be used for supervised learning as data loss while training the model. However, such huge datasets are not available in experiments. Thus, we test the ability of RF-PINN to reconstruct missing fields when either dense or sparse data is provided as ground truth during training. Table 4 describes the considered scenarios based on the sparsity of the data to solve the inverse problem. In the first scenario, we provide dense (≈ 75000) data to the model. For the second and third cases, we use only 0.2% of the available velocity and temperature data, respectively. These cases of sparse datasets may be associated with data measured experimentally with probes. The computation time is comparable across all cases, as the same PDEs (mass and progress conservation equations) are applied, with an average duration of 9 h.

5.3.2. Model performance in case 1

As an initial step, we use dense simulation data of velocity in the x and y directions as data points in the RF-PINN framework. This corresponds to approximately 75000 points in the entire domain. The mass and progress conservation losses reach a minimum of 10^{-3} (Fig. 10a), whereas the boundary and data losses are in the order of 10^{-5}. The line predictions of the velocity components at varying locations are shown in Figs. 10(j-o), and they match the simulation data exactly. However, this is expected as these data were given to the model while training. Figs. 10(d-i), on the other hand, show the line prediction of ρ and T fields along with their contour plots (Figs. 10b-c). We observe that the RF-PINN framework accurately predicts the missing fields T and ρ.

5.3.3. Model performance in case 2

In order to validate the robustness of the framework, we reduce the amount of data provided to the network to $\approx 0.2\%$. Similar to experiments where PIV may not be available due to restricted optical access, but measurements by probes may be provided, we provide the network with sparse data at various sections in the domain. This sums up to only 140 points, as shown in Fig. 11, a massive reduction of data from the previous case. Reducing the data any further increases the prediction error. Keeping the other parameters fixed, we train the network for 10000 iterations with optimizer A and 50000 iterations with optimizer B. Figs. 12(e-j) depict the predicted line plots of the velocities at sections that were not provided as data to the network. The trained model does an excellent job to accurately predict the fields at unknown locations. There is a slight discrepancy in the prediction of u at $x = 4.5$ mm. The contour plots (Figs. 12a-d) are very similar to the DNS fields. Figs. 12(k-p) further show the prediction of the quantities ρ and T, which were not provided to the network, and the predictions are very accurate. This showcases the framework's excellent field reconstruction capacity using sparse data.

5.3.4. Model performance in case 3

To further test the limit of the RF-PINN framework, along with reducing the sparsity of the data, we now also limit the number of physical quantities provided to the network. In this last scenario, we provide sparse line data of the temperature field to the network

Journal of Computational Physics 524 (2025) 113698

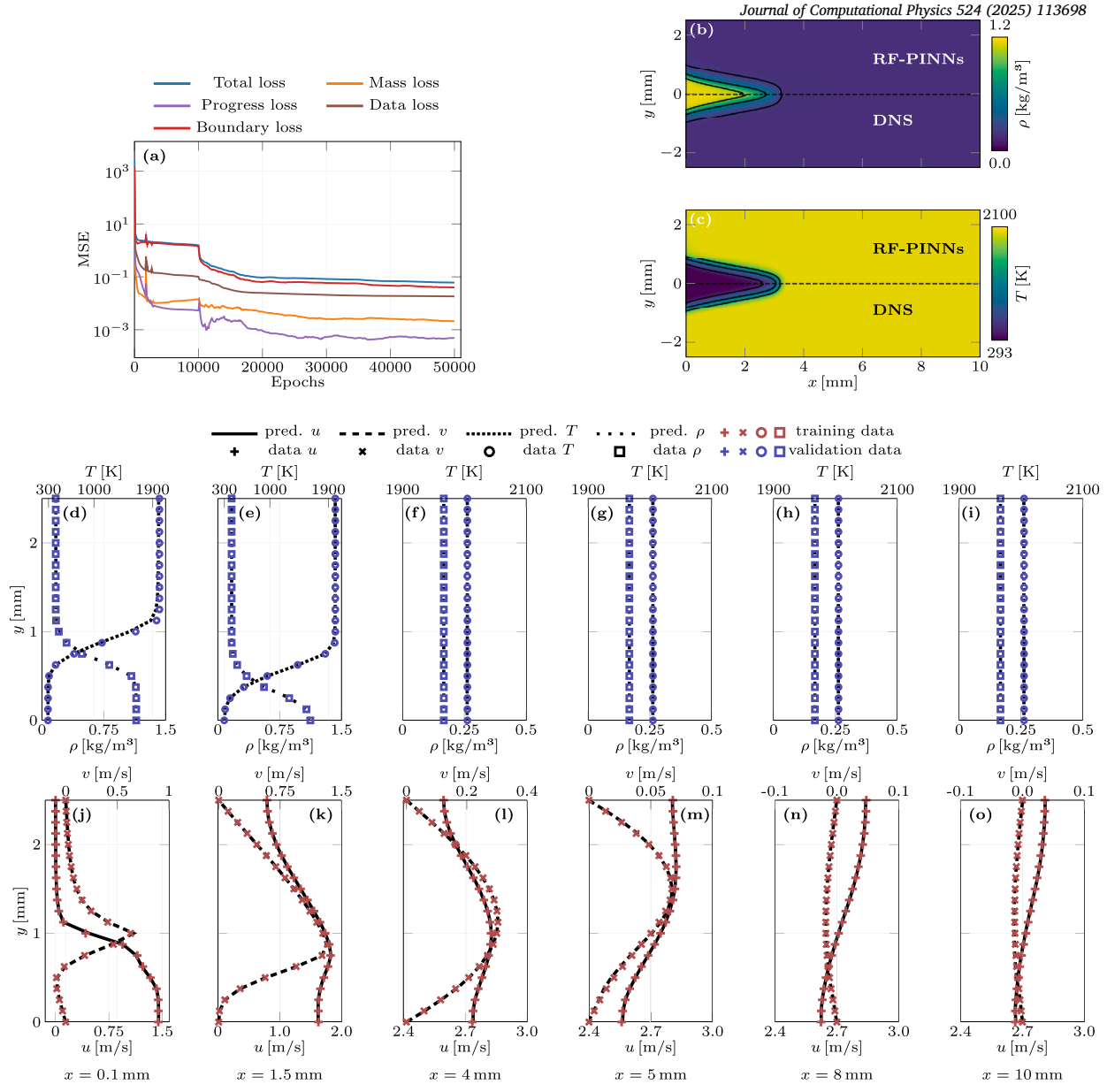

Fig. 10. RF-PINNs prediction and comparison to DNS results for case 1: (a) evolution of loss curves. (b-c) contour plots of density and temperature. (d-i) density and temperature profiles. (j-o) streamwise and transverse velocity components.

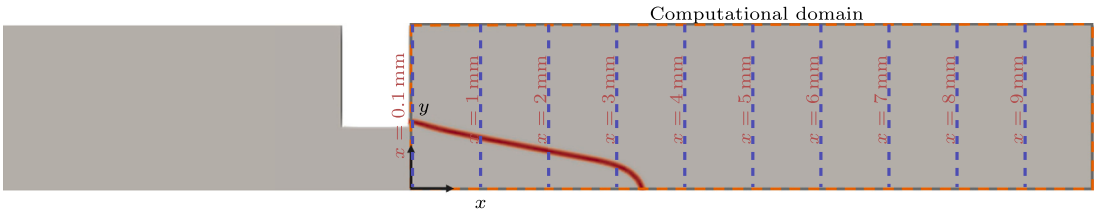

Fig. 11. Kornilov configuration computational domain showing location of sparse data provided as data loss to the model for training.

and train it in the same way as in case 2. The performance of the trained model is visualized in Fig. 13. The model predicts slight discrepancies for the velocity fields at $x = 3.5$ and 7.5 mm compared to the previous cases. This is expected as we have provided very little data. However, the model overall accurately predicts the velocities for all locations. On the other hand, the density and temperature field predictions (Fig. 13 k-p) are very accurate and close to the ground truth.

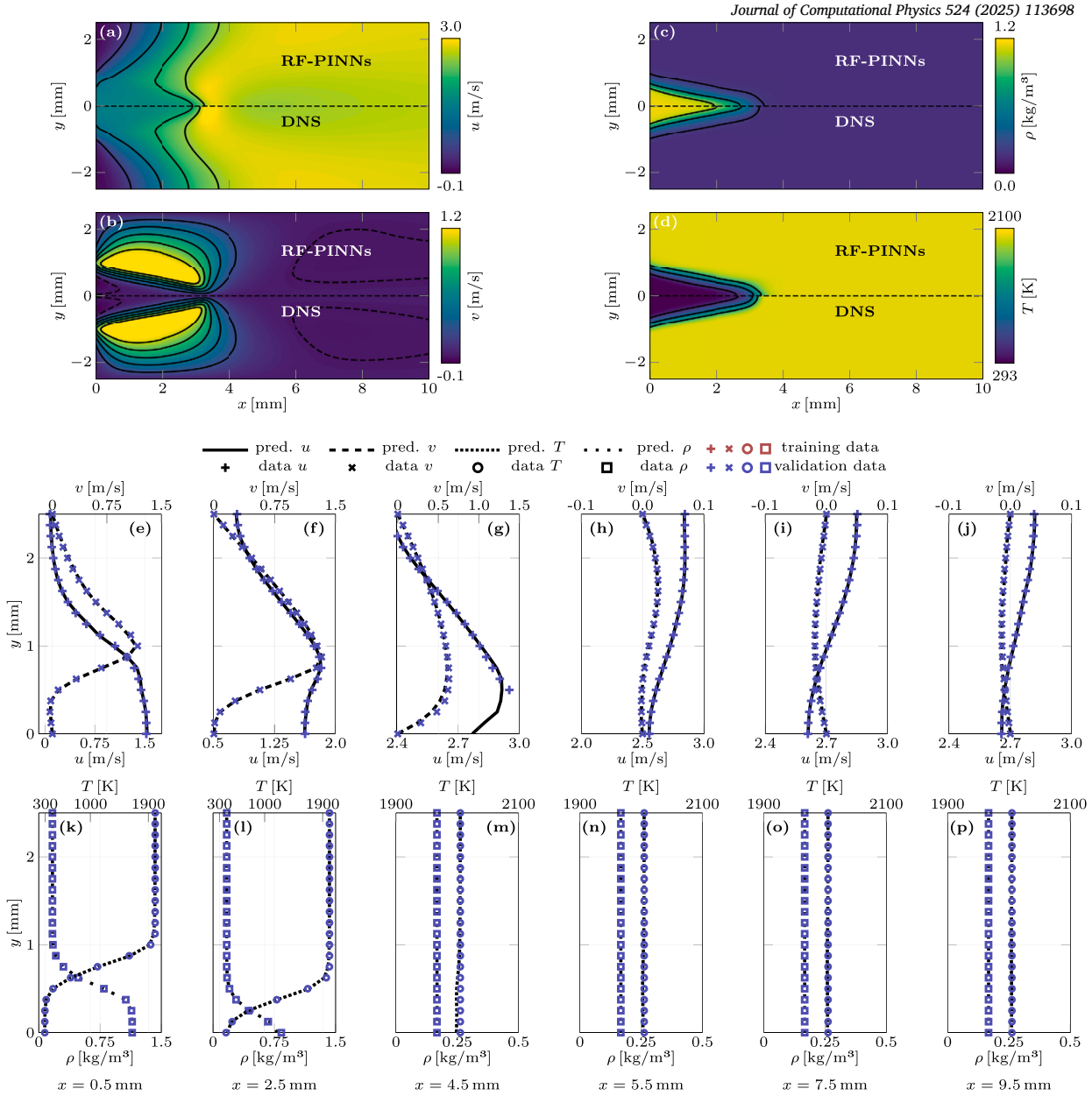

Fig. 12. RF-PINNs prediction and comparison to DNS results for case 2: (a-b) contour plots of velocity components. (c-d) contour plots of density and temperature. (e-j) streamwise and transverse velocity components. (k-p) density and temperature profiles.

6. Application of RF-PINNs to a turbulent flame

Next, we apply the RF-PINNs framework to a configuration featuring a turbulent flame. These are unsteady due to their turbulent nature. However, the scope of this work is not the reconstruction of the unsteady flow fields but the flow's turbulent mean-field quantities. The PDEs in Eqs. (7) are unsuited for this task but are the foundation to derive the mean-field describing PDEs, which we derive briefly in the following. For further details on the derivation, we refer to Poinsot and Veynante [48].

As the flow features an inhomogeneous density field, it is convenient to introduce a density-weighted average for any field quantity $\phi \in \{\rho, \mathbf{u}, p, \theta\}$:

$$\widetilde{\phi} = \frac{\overline{\rho\phi}}{\overline{\rho}}, \tag{20}$$

which is usually referred to as Favre-averaged ($\overline{\phi}$ is the temporal average of ϕ). In addition to this, we decompose any field quantity into

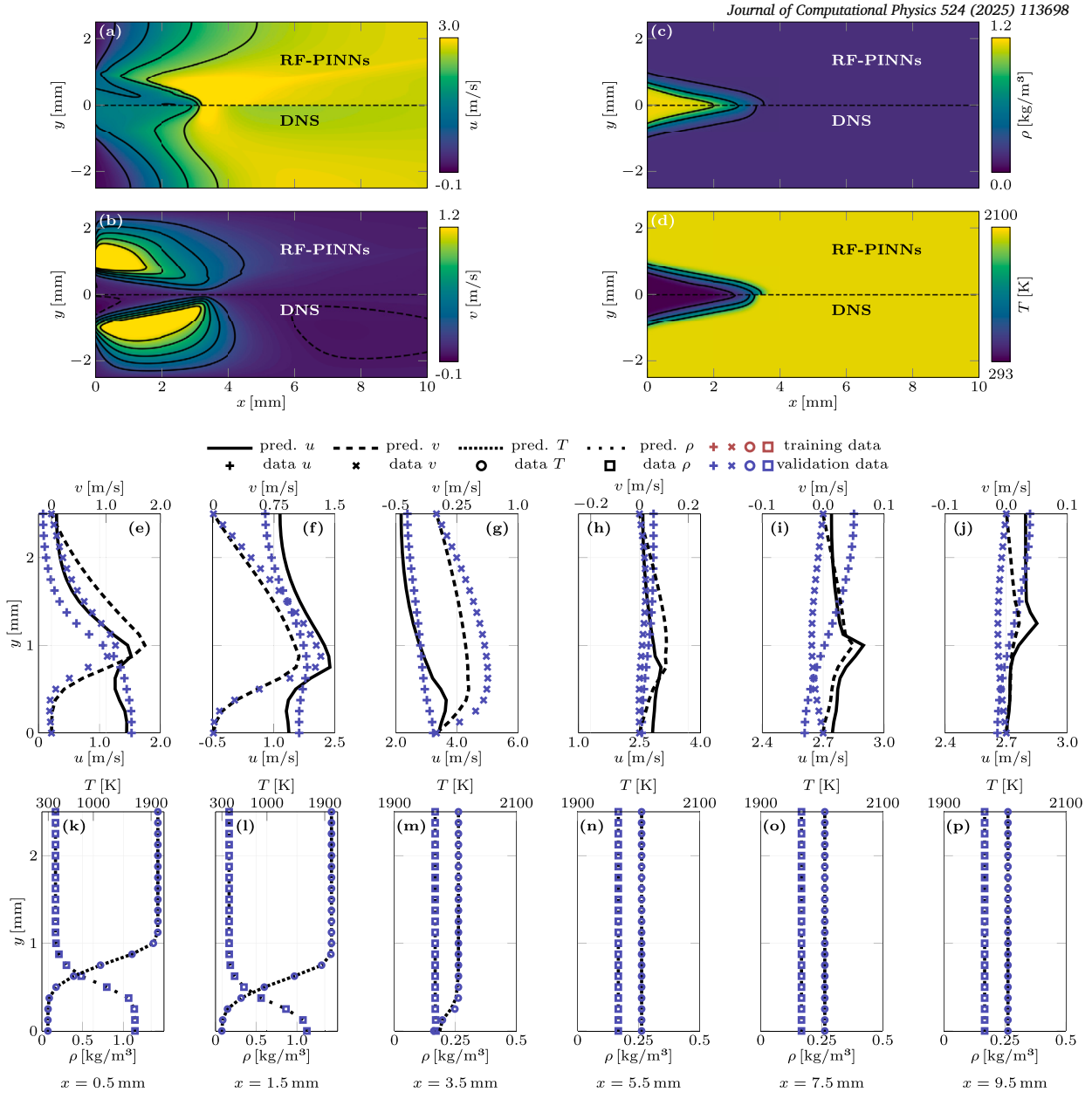

Fig. 13. RF-PINNs prediction and comparison to DNS results for case 3: (a-b) contour plots of velocity components. (c-d) contour plots of density and temperature. (e-j) streamwise and transverse velocity components. (k-p) density and temperature profiles.

$$\phi = \widetilde{\phi} + \phi'', \tag{21}$$

where ϕ'' denotes the fluctuating portion of the field variable ϕ. After substituting this ansatz into the governing, normalized Eqs. (7) for each quantity, the resulting equations are again averaged, resulting in the following set of equations.

$$\nabla \cdot (\overline{\rho}\widetilde{\mathbf{u}}) = 0, \tag{22}$$

$$\overline{\rho}(\widetilde{\mathbf{u}} \cdot \nabla)\widetilde{\mathbf{u}} + \nabla\widetilde{p} + \nabla \cdot \overline{\rho}\widetilde{\mathbf{u}''\mathbf{u}''} - \nabla \cdot \frac{1}{\mathrm{Re}}\left((\nabla\widetilde{\mathbf{u}}) + (\nabla\widetilde{\mathbf{u}})^{\mathrm{T}} - \frac{2}{3}(\nabla \cdot \widetilde{\mathbf{u}})\mathbf{I}\right) = 0, \tag{23}$$

$$\overline{\rho}(\widetilde{\mathbf{u}} \cdot \nabla)\widetilde{\theta} - \nabla \cdot \left(\frac{1}{\mathrm{Re\,Pr}}\nabla\widetilde{\theta}\right) + \nabla \cdot \overline{\rho}\widetilde{\mathbf{u}''\theta''} + \frac{L}{\rho_{\mathrm{in}}u_{\mathrm{in}}}\overline{\omega}_\theta = 0, \tag{24}$$

which governs the turbulent averaged fields. While the unsteady terms vanish on one hand, additional terms, i.e., the Reynolds stresses $\widetilde{\mathbf{u}''\mathbf{u}''}$ and turbulent scalar flux $\widetilde{\mathbf{u}''\,\theta''}$, arise on the other.

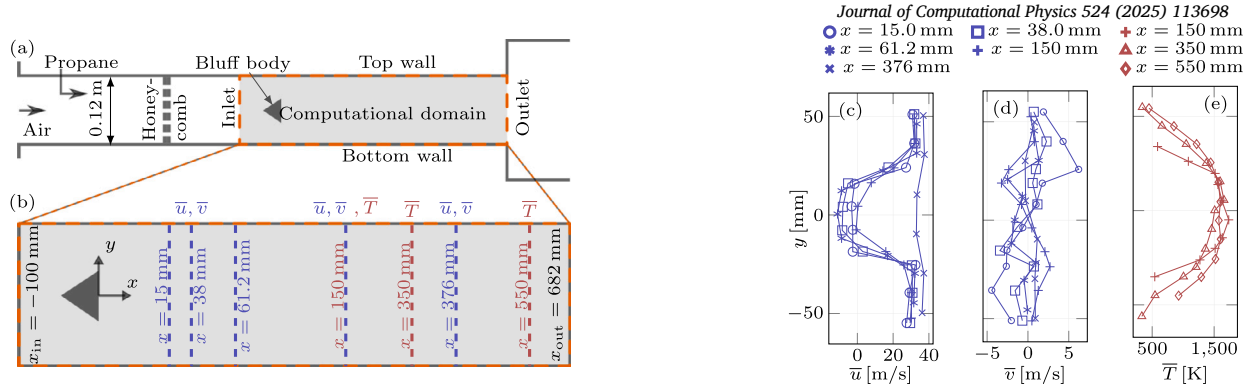

Fig. 14. The Volvo setup: (a) schematic of the test rig, (b) schematic of the computational domain for RF-PINNs, (c-e) experimentally measured mean quantities along the transverse direction for various streamwise locations (extracted from [90]), (c) mean streamwise velocity $\overline{u}$, (d) mean transverse velocity $\overline{v}$, (c) mean temperature $\overline{T}$.

Table 5
Boundary constraints for the computation domain.

	Location	Value
$\widetilde{u}_i$	Flame holder + Top + Down	0.0
$\widetilde{u}_x$	Inlet	17.3 m/s
θ	Inlet	0.0
$\partial_n \theta$	Flame holder + Outlet + Top + Down	0.0
$\partial_n \widetilde{u}_i$	Outlet	0.0

6.1. Volvo flame configuration

We test our framework on a turbulent case to reconstruct the mean fields based on sparse experimental data. Fig. 14(a) depicts a schematic of the combustion experiment under investigation, i.e., the VOLVO test rig. It consists of a channel with a rectangular cross-section (height $H = 0.12$ m, width $W = 0.24$ m) and an equilateral triangular bluff body (edge length $d = 0.04$ m) that is mounted in the center of the channel. The rig may be operated with various propane/air mixtures and bulk velocities at ambient pressure. After passing through a honeycomb section and the triangular flame holder, the mixture reacts within a turbulent 2D V-flame. Prior to various numerical studies of the test rig (e.g., by means of RANS [81–84] or LES [85–88]), it was analyzed experimentally [89,90] where mean-field quantities such as $\overline{u}, \overline{v}, \overline{T}$ were recorded along the transverse coordinate at various streamwise positions (see Fig. 14b). These are depicted in Fig. 14(c-e) for the operating point of the current study, which is defined by equivalence ratio $\phi = 0.65$, unburnt fuel-air mixture temperature $T_u = 288$ K at a bulk velocity of $u_{in} = 17.3$ m/s.

6.2. RF-PINN model

2D equations and BCs:
The configuration is 2D since we consider the mean flow in the center plane, and the mean flow is homogeneous in the depth direction. Indeed, 2D and 3D numerical simulations differ by less than 1% at the symmetric plane (Nilsson and Bai [91]). We, therefore, consider the averaged equations (Eqs. (22)-(24)) only in 2D:

$$\frac{\partial \overline{\rho}^* \widetilde{u}}{\partial x^*} + \frac{\partial \overline{\rho}^* \widetilde{v}}{\partial y^*} = 0, \tag{25}$$

$$\frac{\partial \overline{\rho}^* \widetilde{\theta u}^*}{\partial x^*} + \frac{\partial \overline{\rho}^* \widetilde{\theta v}^*}{\partial y^*} - \frac{\partial}{\partial x^*}\left(\frac{1}{\mathrm{Re\,Pr}}\frac{\partial \widetilde{\theta}}{\partial x^*}\right) - \frac{\partial}{\partial y^*}\left(\frac{1}{\mathrm{Re\,Pr}}\frac{\partial \widetilde{\theta}}{\partial y^*}\right) + \frac{\partial \overline{\rho}^* \widetilde{u^{*\prime\prime}\theta^{\prime\prime}}}{\partial x^*} + \frac{\partial \overline{\rho}^* \widetilde{v^{*\prime\prime}\theta^{\prime\prime}}}{\partial y^*} + \frac{d}{\rho_{in} u_{in}} \overline{\omega}_\theta = 0. \tag{26}$$

The scalar turbulent flux term $\widetilde{u_i^{*\prime\prime}\theta^{\prime\prime}}$ in Eq. (24), similar to the Reynolds stress term $\widetilde{u_i^{\prime\prime} u_j^{\prime\prime}}$ in Eq. (23), arises due to the decomposition of flow quantities into mean and fluctuating components and the subsequent averaging operation of the equations. Most conventional CFD codes treat this term similarly to the Reynolds stresses, e.g., with a gradient diffusion assumption depending on a turbulent viscosity. However, Veynante et al. [92] show that its behavior across the flame highly depends on the flame configuration and that a simple gradient diffusion approach may not be appropriate due to counter-gradient turbulent diffusion. Therefore, in our RF-PINNs framework, we refrain from modeling this term and instead directly predict it by the network as a dependent variable. Similar approaches were recently applied in the context of PINNs for non-reacting turbulent flows, where the Reynolds stresses were predicted [23,93,25].

In addition to the governing equations, we enforce the boundary conditions (Table 5), used as boundary constraints $\mathcal{L}_b(\Psi)$.

Table 6
Cases for testing RF-PINN on Volvo flame (avg. runtime = 13 h).

Case	Input	Output	No. of data points
1	$\widetilde{u}_x$ and $\widetilde{u}_y$	$\widetilde{T}$	122
2	$\widetilde{T}$	$\widetilde{u}_x$ and $\widetilde{u}_y$	72

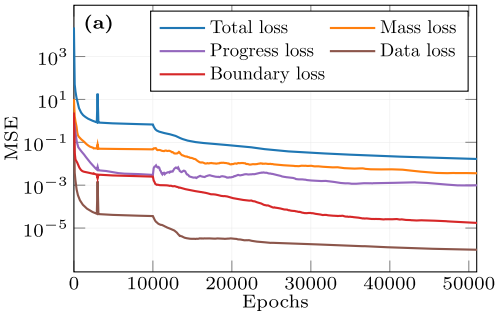

Fig. 15. Comparison of different loss terms for case 1.

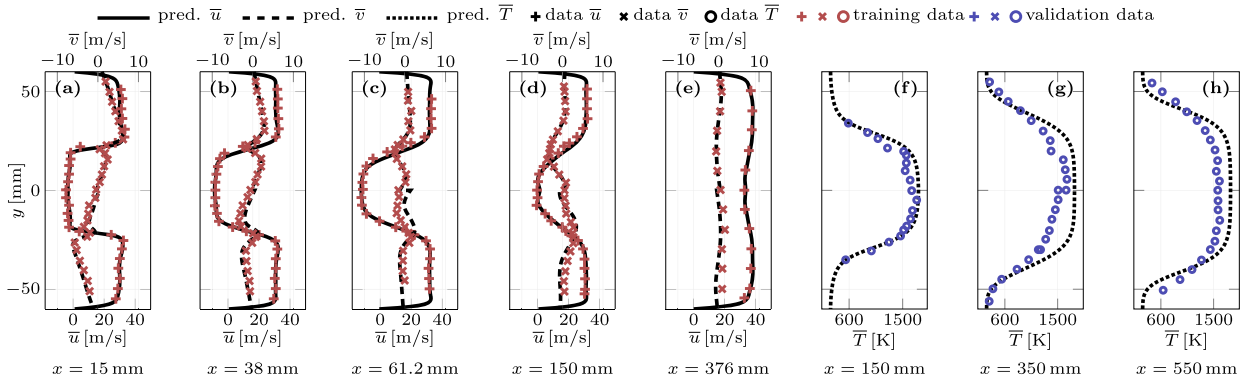

Fig. 16. RF-PINNs prediction and comparison to experimental results for case 1: (a-h) mean profiles along the transverse direction at various streamwise positions. (e-e) streamwise and transverse velocity components. (f-h) temperature.

Architecture:
As we consider the averaged equation in 2D, the input to the network are spatial coordinates ($\boldsymbol{x} := \{x, y\}$). The outputs are the physical quantities of interest ($\boldsymbol{z} := \{\widetilde{\theta}, \widetilde{u}_x, \widetilde{u}_y, \widetilde{u_x''\theta''}, \widetilde{u_y''\theta''}\}$). Unlike the laminar case, the scalar turbulent flux terms ($\widetilde{u_x''\theta''}, \widetilde{u_y''\theta''}\}$) are additionally predicted in the current turbulent case. Since the considered domain is large (≈ 0.7 m), we use a large amount of collocation points (100000) compared to the laminar flame case (5.2.2). The remaining hyperparameters (10 layers of 30 neurons) and the scalar weights on the loss terms (1000 on $\mathcal{L}_d$, $\mathcal{L}_b$, and 1 on $\mathcal{L}_f$) remain constant as that in the laminar 2D case. We train the network for 10000 and 50000 iterations with optimizers A and B, respectively.

6.3. Prediction/analysis

In this section, we present the trained network's performance on a couple of scenarios ($\overline{u}$ as data and $\overline{T}$ as data), in line with our previous test cases (1D and 2D laminar). Since the available experimental data is sparse, we test whether RF-PINNs can reconstruct the missing fields with sparse data in a larger domain.

6.3.1. Case scenarios

Similar to the laminar case, we solve an inverse problem to reconstruct missing fields. We define two scenarios, which are summarized in Table 6. In the first case, we use the PIV measurements to predict, among others, the temperature field. In the second case, we only use the recorded temperature data to reconstruct the velocity fields. The computation time required for the turbulent case is 13 h, which is slightly higher than that of the laminar case due to the larger domain and the additional step of reaction rate mapping. The performance of the models is discussed in the following sections:

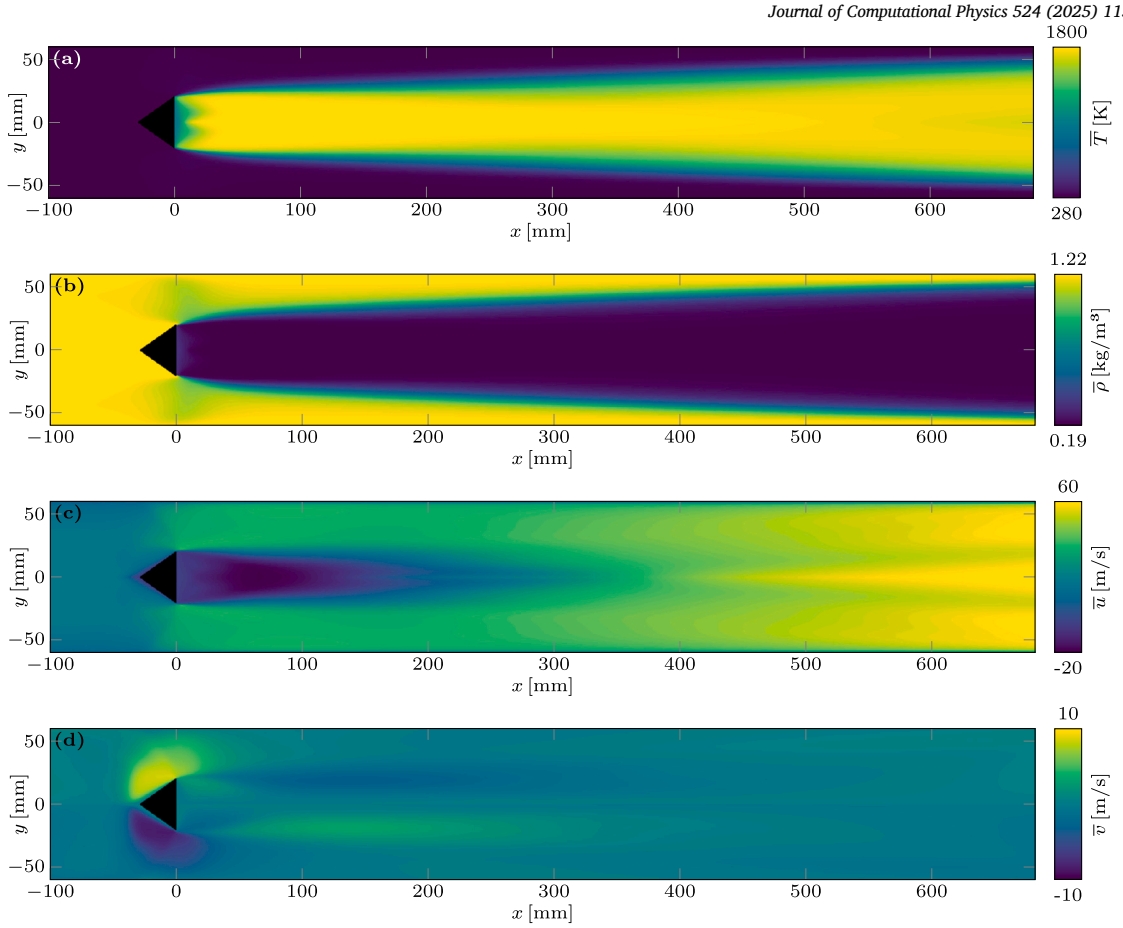

Fig. 17. RF-PINNs field reconstruction in the entire domain for case 1: (a-b) contour plots of predicted temperature and density field. (c-d) contour plots of predicted velocity components.

6.3.2. *Model performance in case 1*

We utilize the available PIV fields in the first case and train the model using optimizers A and B. The loss curve in Fig. 15 shows that the model reaches a plateau after 10000 epochs with optimizer A. The training then switches to L-BFGS, and the loss continues to decrease. After 50000 epochs, the loss curve stabilizes and reaches an approximate value of 10^{-2}. To quantify the predictions, we compare the line plots for all physical quantities in Fig. 16(a-h) with the available experimental data. We observe excellent agreement of the velocity profiles $(\bar{u}, \bar{v})$ at all sections, which is expected as the velocity fields were given as data points while training the model. However, it is exciting that the model can accurately predict the temperature profiles at all positions along the y-direction at all sections. Although we do not have a reference simulation of the considered configuration to compare the field reconstruction directly, we use the averaged LES fields of the domain presented in [94] for qualitative comparison. The contour plot in Fig. 17(a) shows that the temperature field predictions are very similar to those reported in [94]. The streamwise velocity contour (Fig. 17c) also resembles that shown in [94] with the good agreement of the velocity field. It can be seen that the model can also make accurate field predictions downstream of the flow where no data is available. Figs. 17(b and d) represent the reconstructed density and velocity in the transverse direction, respectively.

6.3.3. *Model performance in case 2*

In this scenario, we face the challenge of predicting the axial and tangential velocity fields using only the temperature profiles from three sections (Fig. 14e). The data is sparse, with only 72 data points available in the domain of interest. We trained the RF-PINN with both optimizers A and B for 50000 epochs. Fig. 18 shows the loss curves for case 2. It follows a similar trend as in the previous case, with individual loss terms converging simultaneously. The total loss achieved in this case is in the order of 10^{-2}.

Fig. 19 shows the prediction of the missing fields at different sections. The temperature profile prediction is very accurate due to the imposed data constraint. More interestingly, the mean velocities are well predicted. Slight deviations are present along $x = 150$ mm for $\bar{u}$. This slight inaccuracy is also expected from the model since few data are available. However, the model prediction closely aligns with the experimental values in the remaining domain. Correspondingly, Fig. 20 depicts the reconstructed fields in the entire domain. The overall field reconstruction follows the trend observed in Fig. 17, with accurate predictions downstream ($x > 300$ mm)

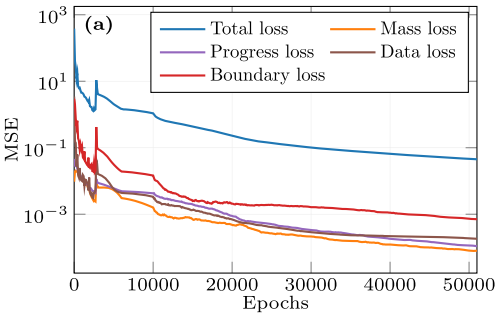

Fig. 18. Comparison of different loss terms for case 2.

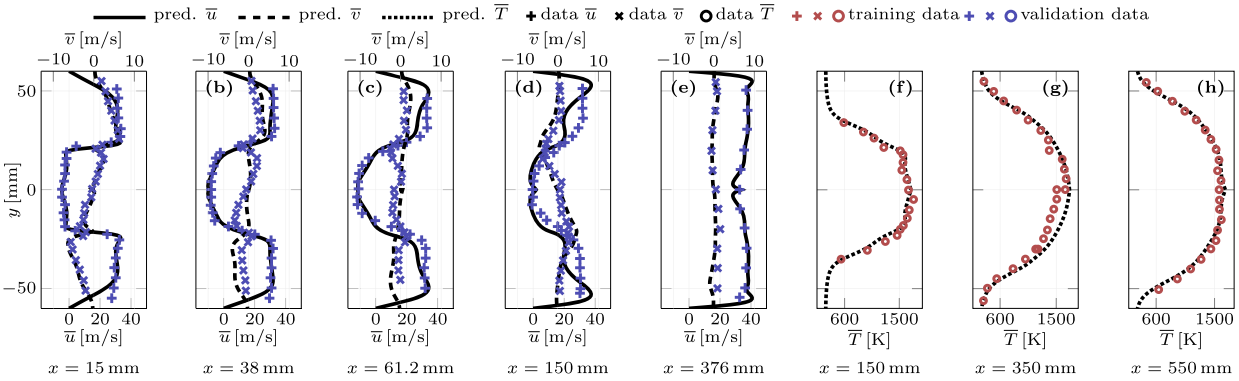

Fig. 19. RF-PINNs prediction and comparison to experimental results for case 2: (a-h) mean profiles along the transverse direction at various streamwise positions. (a-e) streamwise and transverse velocity components. (f-h) temperature.

and upstream ($x < -50$ mm) of the flame holder because of the imposed temperature data and inlet conditions in these regions, respectively. We see a slight deviation in the field reconstruction, especially in the temperature (Fig. 20a) and transverse velocity fields (Fig. 20d) around the flame holder, due to the unavailability of data in this region. Nonetheless, RF-PINNs perform well in reconstructing the fields with such sparse data for such a large domain, highlighting the strength of RF-PINN in accurately predicting fields with a small dataset.

7. Conclusion

Our work introduces the RF-PINN framework to solve inverse problems in reacting flows and predict missing fields using sparse data. The framework incorporates a reaction progress variable transport equation and an associated reaction rate model, simplifying the species and energy conservation equations. We propose an approach to employ a precomputed lookup table during NN training instead of using highly non-linear reaction rate equations to improve the convergence rate. We presented two ways of preparing the lookup table: (i) For the laminar case, we built a surrogate neural network model using computationally inexpensive 1D Cantera simulation with detailed chemical kinetics, and (ii) for the turbulent case, we used the EBU reaction rate model. We thoroughly analyze the conservation equations' requirements using a 1D freely propagating adiabatic prototype flame. We conclude that mass and progress conservation equations are sufficient for the steady-state flame in the low-Mach number limit. We tested the framework on two challenging scenarios of 2D multi-slit laminar and bluff-body stabilized turbulent flames. In the laminar case, the model performed very well in reconstructing the density and temperature fields when dense velocity data ($\approx$ 75000 data points) was provided for the data loss. However, in experiments, typically sparse data of some physical quantities are available. Therefore, we tested the limit of the model by reducing the velocity data points given to the model to as low as 140 ($\approx 0.2\%$) and still achieved accurate field reconstruction. The model had slight discrepancies in some regions when reconstructing the velocity fields when sparse temperature data was provided, but it could very accurately reconstruct the temperature and density fields. In the turbulent case, the model performed well, too, in predicting the mean temperature profiles downstream of the flow, with sparse velocity profiles (122 of each $\bar{u}, \bar{v}$). In the second scenario of the turbulent case, we further reduced the training data and only provided three temperature profiles (72 data points), which challenged the framework. Nevertheless, the model could predict the overall trends in the streamwise and transverse velocity profiles, showcasing the robustness of RF-PINN.

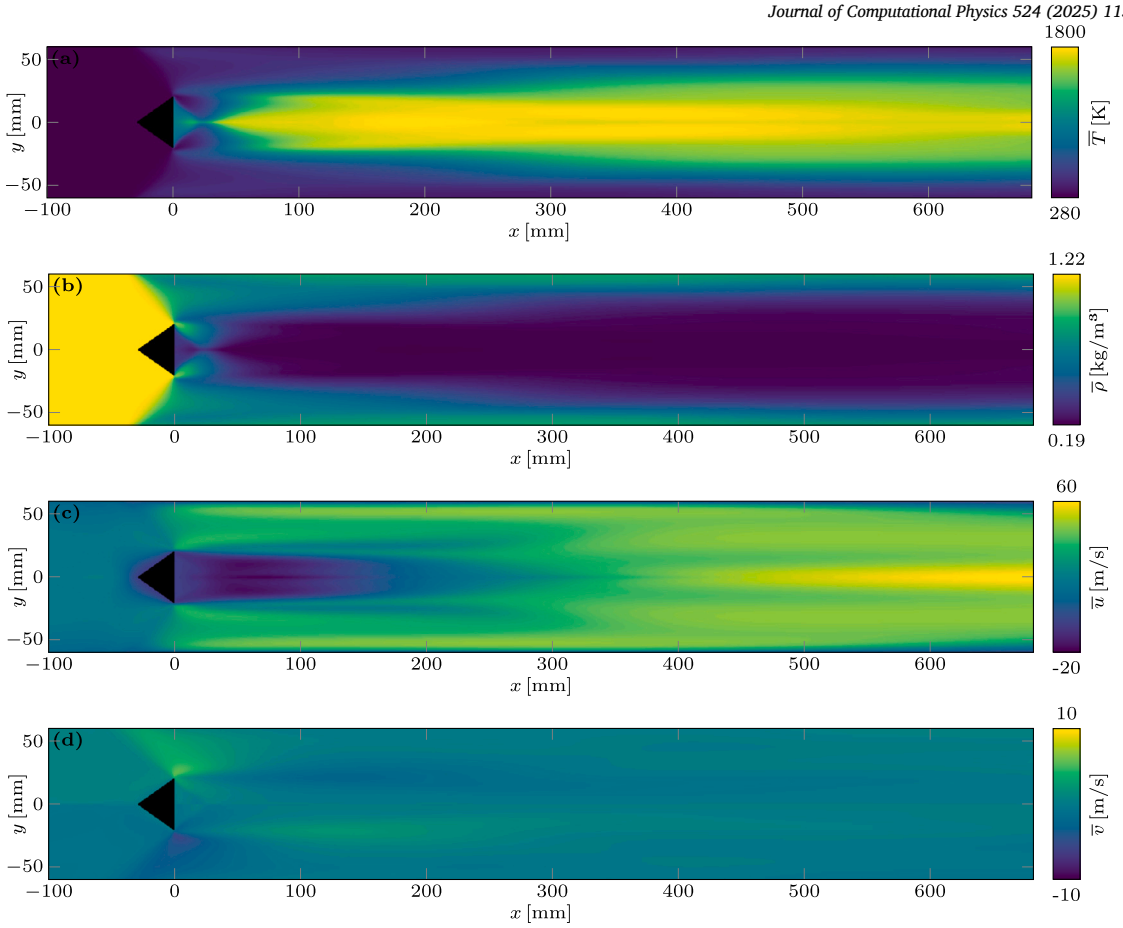

Fig. 20. RF-PINNs field reconstruction in the entire domain for case 2: (a-b) contour plots of predicted temperature and density field. (c-d) contour plots of predicted velocity components.

CRediT authorship contribution statement

Vikas Yadav: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Investigation, Conceptualization. **Mario Casel:** Writing – original draft, Methodology, Data curation. **Abdulla Ghani:** Writing – review & editing, Supervision, Methodology, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

We acknowledge funding from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - Project 384950143. Partially funded by the European Union (ERC, TACOS, 101078836). We sincerely thank Prof. Volker Mehrmann from the Institute of Mathematics (TU Berlin) for critical feedback and fruitful discussions. Views and opinions expressed are, however, those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

Data availability

Data will be made available on request.

References

[1] Z. Bai, S.L. Brunton, B.W. Brunton, J.N. Kutz, E. Kaiser, A. Spohn, B.R. Noack, Data-Driven Methods in Fluid Dynamics: Sparse Classification from Experimental Data, Springer, 2017.
[2] S.L. Brunton, B.R. Noack, P. Koumoutsakos, Machine learning for fluid mechanics, Annu. Rev. Fluid Mech. 52 (2020) 477–508.
[3] K. Fukami, K. Fukagata, K. Taira, Assessment of supervised machine learning methods for fluid flows, Theor. Comput. Fluid Dyn. 34 (2020) 497–519, https://doi.org/10.1007/s00162-020-00518-y.
[4] S.-B. Lin, Generalization and expressivity for deep nets, IEEE Trans. Neural Netw. Learn. Syst. 30 (2019) 1392–1406, https://doi.org/10.1109/TNNLS.2018.2868980.
[5] Q. Wang, L. Yan, G. Hu, C. Li, Y. Xiao, H. Xiong, J. Rabault, B.R. Noack, Drlinfluids: an open-source python platform of coupling deep reinforcement learning and openfoam, Phys. Fluids 34 (2022).
[6] T. Pfaff, M. Fortunato, A. Sanchez-Gonzalez, P. Battaglia, Learning mesh-based simulation with graph networks, in: International Conference on Learning Representations, 2021.
[7] K. Fukami, R. Maulik, N. Ramachandra, K. Fukagata, K. Taira, Global field reconstruction from sparse sensors with Voronoi tessellation-assisted deep learning, Nat. Mach. Intell. 3 (2021) 945–951, https://doi.org/10.1038/s42256-021-00402-2.
[8] A. Chattopadhyay, E. Nabizadeh, E. Bach, P. Hassanzadeh, Deep learning-enhanced ensemble-based data assimilation for high-dimensional nonlinear dynamical systems, J. Comput. Phys. 477 (2023) 111918, https://doi.org/10.1016/j.jcp.2023.111918.
[9] T. Grenga, L. Nista, C. Schumann, A.N. Karimi, G. Scialabba, A. Attili, H. Pitsch, Predictive data-driven model based on generative adversarial network for premixed turbulence-combustion regimes, Combust. Sci. Technol. 195 (2023) 3923–3946, https://doi.org/10.1080/00102202.2022.2041624.
[10] L. Han, Q. Gao, D. Zhang, Z. Feng, Z. Sun, B. Li, Z. Li, Deep neural network-based generation of planar CH distribution through flame chemiluminescence in premixed turbulent flame, Energy AI 12 (2023) 100221, https://doi.org/10.1016/j.egyai.2022.100221.
[11] M. Rywik, A. Zimmermann, A.J. Eder, E. Scoletta, W. Polifke, Spatially Resolved Modeling of the Nonlinear Dynamics of a Laminar Premixed Flame with a Multilayer Perceptron-Convolution Autoencoder Network, Zenodo, 2023.
[12] M. Ihme, W.T. Chung, A.A. Mishra, Combustion machine learning: principles, progress and prospects, Prog. Energy Combust. Sci. 91 (2022) 101010.
[13] M.W.M.G. Dissanayake, N. Phan-Thien, Neural-network-based approximations for solving partial differential equations, Commun. Numer. Methods Eng. 10 (1994) 195–201, https://doi.org/10.1002/cnm.1640100303.
[14] G.E. Karniadakis, I.G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, L. Yang, Physics-informed machine learning, Nat. Rev. Phys. 3 (2021) 422–440.
[15] S. Cuomo, V.S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, F. Piccialli, Scientific machine learning through physics–informed neural networks: where we are and what's next, J. Sci. Comput. 92 (2022) 88.
[16] S. Wang, S. Sankaran, H. Wang, P. Perdikaris, An Expert's Guide to Training Physics-Informed Neural Networks, 2023, Publisher: arXiv Version Number: 1.
[17] S. Cai, Z. Mao, Z. Wang, M. Yin, G.E. Karniadakis, Physics-informed neural networks (PINNs) for fluid mechanics: a review, Acta Mech. Sin. 37 (2021) 1727–1738, https://doi.org/10.1007/s10409-021-01148-1.
[18] P. Sharma, W.T. Chung, B. Akoush, M. Ihme, A review of physics-informed machine learning in fluid mechanics, Energies 16 (2023) 2343, https://doi.org/10.3390/en16052343.
[19] M. Raissi, P. Perdikaris, G. Karniadakis, Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.
[20] J. Stiasny, S. Chevalier, S. Chatzivasileiadis, Learning without data: physics-informed neural networks for fast time-domain simulation, in: 2021 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm), IEEE, Aachen, Germany, 2021, pp. 438–443.
[21] N. Wandel, M. Weinmann, M. Neidlin, R. Klein, Spline-PINN: approaching PDEs without data using fast, physics-informed Hermite-spline CNNs, Proc. AAAI Conf. Artif. Intell. 36 (2022) 8529–8538, https://doi.org/10.1609/aaai.v36i8.20830.
[22] K. Nath, X. Meng, D.J. Smith, G.E. Karniadakis, Physics-informed neural networks for predicting gas flow dynamics and unknown parameters in diesel engines, Sci. Rep. 13 (2023) 13683, https://doi.org/10.1038/s41598-023-39989-4.
[23] J.G.R. von Saldern, J.M. Reumschüssel, T.L. Kaiser, M. Sieber, K. Oberleithner, Mean flow data assimilation based on physics-informed neural networks, Phys. Fluids 34 (2022) 115129, https://doi.org/10.1063/5.0116218.
[24] A.D. Jagtap, Z. Mao, N. Adams, G.E. Karniadakis, Physics-informed neural networks for inverse problems in supersonic flows, J. Comput. Phys. 466 (2022) 111402.
[25] L. Sliwinski, G. Rigas, Mean flow reconstruction of unsteady flows using physics-informed neural networks, Data-Cent. Eng. 4 (2023) e4, https://doi.org/10.1017/dce.2022.37.
[26] D. Shu, Z. Li, A. Barati Farimani, A physics-informed diffusion model for high-fidelity flow field reconstruction, J. Comput. Phys. 478 (2023) 111972, https://doi.org/10.1016/j.jcp.2023.111972.
[27] A. Kontogiannis, M.P. Juniper, Physics-informed compressed sensing for PC-MRI: an inverse Navier-Stokes problem, IEEE Trans. Image Process. 32 (2023) 281–294, https://doi.org/10.1109/TIP.2022.3228172.
[28] E. Özalp, G. Margazoglou, L. Magri, Physics-informed long short-term memory for forecasting and reconstruction of chaos, in: J. Mikyška, C. De Mulatier, M. Paszynski, V.V. Krzhizhanovskaya, J.J. Dongarra, P.M. Sloot (Eds.), Computational Science – ICCS 2023, in: Lecture Notes in Computer Science, vol. 10476, Springer Nature Switzerland, Cham, 2023, pp. 382–389.
[29] M. Bode, M. Gauding, Z. Lian, D. Denker, M. Davidovic, K. Kleinheinz, J. Jitsev, H. Pitsch, Using physics-informed enhanced super-resolution generative adversarial networks for subfilter modeling in turbulent reactive flows, Proc. Combust. Inst. 38 (2021) 2617–2625.
[30] C. Chi, S. Sreekumar, D. Thévenin, Data-driven discovery of heat release rate markers for premixed NH 3 /H 2 /air flames using physics-informed machine learning, Fuel 330 (2022) 125508, https://doi.org/10.1016/j.fuel.2022.125508.
[31] V. Yadav, M. Casel, A. Ghani, Physics-informed recurrent neural networks for linear and nonlinear flame dynamics, Proc. Combust. Inst. 39 (2023) 1597–1606.
[32] H. Son, M. Lee, A PINN approach for identifying governing parameters of noisy thermoacoustic systems, J. Fluid Mech. 984 (2024) A21, https://doi.org/10.1017/jfm.2024.219.
[33] K. Liu, K. Luo, Y. Cheng, A. Liu, H. Li, J. Fan, S. Balachandar, Surrogate modeling of parameterized multi-dimensional premixed combustion with physics-informed neural networks for rapid exploration of design space, Combust. Flame 258 (2023) 113094.
[34] R. Sankaran, E.R. Hawkes, J.H. Chen, T. Lu, C.K. Law, Structure of a spatially developing turbulent lean methane–air Bunsen flame, Proc. Combust. Inst. 31 (2007) 1291–1298, https://doi.org/10.1016/j.proci.2006.08.025.
[35] A. Krishnapriyan, A. Gholami, S. Zhe, R. Kirby, M.W. Mahoney, Characterizing possible failure modes in physics-informed neural networks, Adv. Neural Inf. Process. Syst. 34 (2021) 26548–26560.
[36] T.G. Grossmann, U.J. Komorowska, J. Latz, C.-B. Schönlieb, Can physics-informed neural networks beat the finite element method?, IMA J. Appl. Math. 89 (2024) 143–174, https://doi.org/10.1093/imamat/hxae011.
[37] M.P. Sitte, N.A.K. Doan, Velocity reconstruction in puffing pool fires with physics-informed neural networks, Phys. Fluids 34 (2022) 087124, https://doi.org/10.1063/5.0097496.
[38] S. Liu, H. Wang, Z. Sun, K.K. Foo, G.J. Nathan, X. Dong, M.J. Evans, B.B. Dally, K. Luo, J. Fan, Reconstructing soot fields in acoustically forced laminar sooting flames using physics-informed machine learning models, Proc. Combust. Inst. 40 (2024) 105314, https://doi.org/10.1016/j.proci.2024.105314, https://linkinghub.elsevier.com/retrieve/pii/S154074892400124X.

[39] S. Liu, H. Wang, J.H. Chen, K. Luo, J. Fan, High-resolution reconstruction of turbulent flames from sparse data with physics-informed neural networks, Combust. Flame 260 (2024) 113275, https://doi.org/10.1016/j.combustflame.2023.113275.
[40] M. Nechita, Solving ill-posed Helmholtz problems with physics-informed neural networks, J. Numer. Anal. Approx. Theory 52 (2023) 90–101, https://doi.org/10.33993/jnaat521-1305.
[41] L. Lu, P. Jin, G. Pang, Z. Zhang, G.E. Karniadakis, Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators, Nat. Mach. Intell. 3 (2021) 218–229, https://doi.org/10.1038/s42256-021-00302-5.
[42] N. Peters, Turbulent combustion, Meas. Sci. Technol. 12 (2001) 2022, https://doi.org/10.1088/0957-0233/12/11/708.
[43] L. Vervisch, B. Labegorre, J. Réveillon, Hydrogen–sulphur oxy-flame analysis and single-step flame tabulated chemistry, Fuel 83 (2004) 605–614, https://doi.org/10.1016/j.fuel.2003.09.008.
[44] G. Ribert, O. Gicquel, N. Darabiha, D. Veynante, Tabulation of complex chemistry based on self-similar behavior of laminar premixed flames, Combust. Flame 146 (2006) 649–664, https://doi.org/10.1016/j.combustflame.2006.07.002.
[45] F. Proch, P. Domingo, L. Vervisch, A.M. Kempf, Flame resolved simulation of a turbulent premixed bluff-body burner experiment. Part I: analysis of the reaction zone dynamics with tabulated chemistry, Combust. Flame 180 (2017) 321–339, https://doi.org/10.1016/j.combustflame.2017.02.011.
[46] M. Bode, N. Collier, F. Bisetti, H. Pitsch, Adaptive chemistry lookup tables for combustion simulations using optimal B-spline interpolants, Combust. Theory Model. 23 (2019) 674–699, https://doi.org/10.1080/13647830.2019.1583379.
[47] S. Popp, S. Hartl, D. Butz, D. Geyer, A. Dreizler, L. Vervisch, C. Hasse, Assessing multi-regime combustion in a novel burner configuration with large eddy simulations using tabulated chemistry, Proc. Combust. Inst. 38 (2021) 2551–2558, https://doi.org/10.1016/j.proci.2020.06.098.
[48] T. Poinsot, D. Veynante, Theoretical and Numerical Combustion, 3. ed., CNRS, Toulouse, 2011.
[49] X. Jin, S. Cai, H. Li, G.E. Karniadakis, NSFnets (Navier-Stokes flow nets): physics-informed neural networks for the incompressible Navier-Stokes equations, J. Comput. Phys. 426 (2021) 109951.
[50] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Netw. 2 (1989) 359–366.
[51] P. Sibi, S.A. Jones, P. Siddarth, Analysis of different activation functions using back propagation neural networks, J. Theor. Appl. Inf. Technol. 47 (2013) 1264–1268.
[52] B. Ding, H. Qian, J. Zhou, Activation functions and their characteristics in deep neural networks, in: 2018 Chinese Control and Decision Conference (CCDC), IEEE, Shenyang, 2018, pp. 1836–1841.
[53] A.S. Weigend, On overfitting and the effective number of hidden units, in: M.C. Mozer, P. Smolensky, D.S. Touretzky, J.L. Elman, A.S. Weigend (Eds.), Proceedings of the 1993 Connectionist Models Summer School, 0 ed., Psychology Press, 2014.
[54] J. Tompson, K. Schlachter, P. Sprechmann, K. Perlin, Accelerating Eulerian fluid simulation with convolutional networks, in: D. Precup, Y.W. Teh (Eds.), Proceedings of the 34th International Conference on Machine Learning, in: Proceedings of Machine Learning Research, vol. 70, PMLR, 2017, pp. 3424–3433.
[55] K. Um, R. Brand, Y.R. Fei, P. Holl, N. Thuerey, Solver-in-the-loop: learning from differentiable physics to interact with iterative pde-solvers, in: H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin (Eds.), Advances in Neural Information Processing Systems, vol. 33, Curran Associates, Inc., 2020, pp. 6111–6122.
[56] W. E, B. Yu, The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems, Commun. Math. Stat. 6 (2018) 1–12, https://doi.org/10.1007/s40304-018-0127-z.
[57] D.E. Ozan, L. Magri, Hard-constrained neural networks for modeling nonlinear acoustics, Phys. Rev. Fluids 8 (2023) 103201, https://doi.org/10.1103/PhysRevFluids.8.103201.
[58] L. Lu, R. Pestourie, W. Yao, Z. Wang, F. Verdugo, S.G. Johnson, Physics-informed neural networks with hard constraints for inverse design, SIAM J. Sci. Comput. 43 (2021) B1105–B1132, https://doi.org/10.1137/21M1397908.
[59] N.A.K. Doan, W. Polifke, L. Magri, Physics-informed echo state networks for chaotic systems forecasting, in: Computational Science–ICCS 2019: 19th International Conference, Proceedings, Part IV 19, Faro, Portugal, June 12–14, 2019, Springer, 2019, pp. 192–198.
[60] M.A. Nabian, R.J. Gladstone, H. Meidani, Efficient training of physics-informed neural networks via importance sampling, Comput.-Aided Civ. Eng. 36 (2021) 962–977, https://doi.org/10.1111/mice.12685.
[61] C. Wu, M. Zhu, Q. Tan, Y. Kartha, L. Lu, A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks, Comput. Methods Appl. Mech. Eng. 403 (2023) 115671, https://doi.org/10.1016/j.cma.2022.115671.
[62] A.D. Jagtap, K. Kawaguchi, G.E. Karniadakis, Adaptive activation functions accelerate convergence in deep and physics-informed neural networks, J. Comput. Phys. 404 (2020) 109136, https://doi.org/10.1016/j.jcp.2019.109136.
[63] A.D. Jagtap, K. Kawaguchi, G. Em Karniadakis, Locally adaptive activation functions with slope recovery for deep and physics-informed neural networks, Proc. R. Soc. A 476 (2020) 20200334, https://doi.org/10.1098/rspa.2020.0334.
[64] L.D. McClenny, U.M. Braga-Neto, Self-adaptive physics-informed neural networks, J. Comput. Phys. 474 (2023) 111722, https://doi.org/10.1016/j.jcp.2022.111722.
[65] D.G. Goodwin, R.L. Speth, H.K. Moffat, B.W. Weber, Cantera: An Object-Oriented Software Toolkit for Chemical Kinetics, Thermodynamics, and Transport Processes, 2021, https://doi.org/10.5281/ZENODO.4527812.
[66] F.M. Rohrhofer, S. Posch, C. Gößnitzer, B.C. Geiger, On the apparent Pareto front of physics-informed neural networks, IEEE Access (2023).
[67] Y. Liu, L. Cai, Y. Chen, B. Wang, Physics-informed neural networks based on adaptive weighted loss functions for Hamilton-Jacobi equations, Math. Biosci. Eng. 19 (2022) 12866–12896.
[68] A.D. Jagtap, G.E. Karniadakis, How important are activation functions in regression and classification? A survey, performance comparison, and future directions, J. Mach. Learn. Model. Comput. 4 (2023) 21–75.
[69] O. Delalleau, Y. Bengio, Shallow vs. Deep sum-product networks, Adv. Neural Inf. Process. Syst. 24 (2011).
[70] M. Frenklach, H. Wang, M. Goldenberg, G.P. Smith, D.M. Golden, Gri-mech: an optimized detailed chemical reaction mechanism for methane combustion, topical report, September 1992-August 1995, Technical Report, SRI International, Menlo Park, CA (United States), 1995.
[71] V. Kornilov, K. Schreel, L. de Goey, Experimental assessment of the acoustic response of laminar premixed bunsen flames, Proc. Combust. Inst. 31 (2007) 1239–1246.
[72] M. Haeringer, M. Merk, W. Polifke, Inclusion of higher harmonics in the flame describing function for predicting limit cycles of self-excited combustion instabilities, Proc. Combust. Inst. 37 (2019) 5255–5262.
[73] A. Ghani, W. Polifke, Control of intrinsic thermoacoustic instabilities using hydrogen fuel, Proc. Combust. Inst. 38 (2021) 6077–6084.
[74] M. Casel, A. Ghani, Analysis of the flame dynamics in methane/hydrogen fuel blends at elevated pressures, Proc. Combust. Inst. 39 (2023) 4631–4640, https://doi.org/10.1016/j.proci.2022.07.211.
[75] A. Bonhomme, Numerical study of laminar and turbulent flames propagating in a fan-stirred vessel, Theses, Institut National Polytechnique de Toulouse - INPT, 2014.
[76] E. Courtine, L. Selle, T. Poinsot, DNS of intrinsic ThermoAcoustic modes in laminar premixed flames, Combust. Flame 162 (2015) 4331–4341, https://doi.org/10.1016/j.combustflame.2015.07.002.
[77] C.F. Silva, T. Emmert, S. Jaensch, W. Polifke, Numerical study on intrinsic thermoacoustic instability of a laminar premixed flame, Combust. Flame 162 (2015) 3370–3378, https://doi.org/10.1016/j.combustflame.2015.06.003.
[78] S. Berger, S. Richard, F. Duchaine, L. Gicquel, Variations of anchoring pattern of a bluff-body stabilized laminar premixed flame as a function of the wall temperature, in: Volume 5B: Heat Transfer, American Society of Mechanical Engineers, Seoul, South Korea, 2016, V05BT17A004.

[79] D. Mejia, M. Miguel-Brebion, A. Ghani, T. Kaiser, F. Duchaine, L. Selle, T. Poinsot, Influence of flame-Holder temperature on the acoustic flame transfer functions of a laminar flame, Combust. Flame 188 (2018) 5–12, https://doi.org/10.1016/j.combustflame.2017.09.016.
[80] B. Franzelli, E. Riber, M. Sanjosé, T. Poinsot, A two-step chemical scheme for kerosene–air premixed flames, Combust. Flame 157 (2010) 1364–1373.
[81] S. Olovsson, Combustion calculations on a premixed system with a bluff body flameholder, in: 28th Joint Propulsion Conference and Exhibit, American Institute of Aeronautics and Astronautics, Nashville, TN, U.S.A., 1992.
[82] V. Zimont, V. Battaglia, Joint RANS/LES approach to premixed flames modelling in the context of the TFC combustion model, in: Engineering Turbulence Modelling and Experiments 6, Elsevier, 2005, pp. 905–914.
[83] P. Eriksson, The Zimont TFC model applied to premixed bluff body stabilized combustion using four different RANS turbulence models, in: Volume 2: Turbo Expo 2007, ASMEDC, 2007, pp. 353–361.
[84] B. Manickam, J. Franke, S.P.R. Muppala, F. Dinkelacker, Large-eddy simulation of triangular-stabilized lean premixed turbulent flames: quality and error assessment, Flow Turbul. Combust. 88 (2012) 563–596.
[85] P.A. Cocks, M.C. Soteriou, V. Sankaran, Impact of numerics on the predictive capabilities of reacting flow LES, Combust. Flame 162 (2015) 3394–3411.
[86] A. Ghani, T. Poinsot, L. Gicquel, G. Staffelbach, LES of longitudinal and transverse self-excited combustion instabilities in a bluff-body stabilized turbulent premixed flame, Combust. Flame 162 (2015) 4075–4083, https://doi.org/10.1016/j.combustflame.2015.08.024.
[87] A. Ghani, M. Miguel-Brebion, L. Selle, F. Duchaine, T. Poinsot, Effect of wall heat transfer on screech in a turbulent premixed combustor, in: Center for Turbulence Research Proceedings of the Summer Program, 2016, pp. 133–141.
[88] N. Zettervall, K. Nordin-Bates, E. Nilsson, C. Fureby, Large eddy simulation of a premixed bluff body stabilized flame using global and skeletal reaction mechanisms, Combust. Flame 179 (2017) 1–22, https://doi.org/10.1016/j.combustflame.2016.12.007.
[89] A. Sjunnesson, S. Olovsson, B. Sjoblom, Validation rig- a tool for flame studies, in: International Symposium on Air Breathing Engines, 10th, Nottingham, England, 1991, pp. 385–393.
[90] A. Sjunnesson, C. Nelsson, E. Max, Lda measurements of velocities and turbulence in a bluff body stabilized flame, in: Fourth International Conference on Laser Anemometry – Advances and Application, vol. 3, ASME, Cleveland, 1991, pp. 83–90.
[91] P. Nilsson, X. Bai, Level-set flamelet library approach for premixed turbulent combustion, Exp. Therm. Fluid Sci. 21 (2000) 87–98, https://doi.org/10.1016/S0894-1777(99)00058-8.
[92] D. Veynante, A. Trouvé, K.N.C. Bray, T. Mantel, Gradient and counter-gradient scalar transport in turbulent premixed flames, J. Fluid Mech. 332 (1997) 263–293, https://doi.org/10.1017/S0022112096004065.
[93] H. Eivazi, M. Tahani, P. Schlatter, R. Vinuesa, Physics-informed neural networks for solving Reynolds-averaged Navier–Stokes equations, Phys. Fluids 34 (2022) 075117, https://doi.org/10.1063/5.0095270.
[94] D. Maestro, A. Ghani, L.Y. Gicquel, T. Poinsot, LES reliability of the Volvo bluff-body stabilized ame dynamics, in: 55th AIAA Aerospace Sciences Meeting, AIAA, Grapevine, Texas, 2017.