A Physics-Informed Neural Network Approach to Predicting Dynamics and Identifying Unknown Parameters in Diesel Engines

Dimitrios Kalkantzis, Aris Moustakas
National and Kapodistrian University of Athens

September 2025

Abstract

This work proposes a Physics-Informed Neural Network (PINN) framework for simulating state dynamics and inferring unknown parameters of a mean diesel engine system. The underlying system is governed by six differential equations and two algebraic equations, involving a total of eight state variables. In addition, four unknown parameters are identified as quantities to be estimated.

Normally, the loss function of the PINN consists of three principal components: the initial condition loss, the equation residual loss and the data loss. Without proper treatment, this imbalance can hinder optimization and lead to biased learning. To address this issue, each loss component is normalized by its characteristic order of magnitude at the initial operating condition, thereby ensuring that all terms contribute comparably during training.

Furthermore, certain analytical functions embedded in the residual equations exhibit complex nonlinearities and are computationally expensive to evaluate directly. To address this, pre-trained neural networks were employed as surrogates, enabling efficient and accurate approximation of these terms within the loss function. Finally, in order to reduce the overall training complexity, the PINN framework was restricted to learn the dynamics of state variables whose governing equations are not coupled, thereby simplifying optimization while preserving predictive capability.

The proposed method demonstrated strong performance under both clean and noisy data conditions. Robustness was maintained for noise levels up to 4%, and additional tests were conducted under data sparsity scenarios. Across these settings, the framework successfully reproduced the dynamics of the state variables. Moreover, despite the challenging conditions, the model was able to reliably infer two out of the four unknown parameters.

1 Introduction

Over the past decades, the growing availability of data and increased computational power have shifted modeling approaches from explicitly programmed systems to data-driven learning. Machine learning (ML) enables algorithms to infer patterns from data, and deep learning (DL), using multi-layered neural networks, has proven particularly effective at capturing complex patterns in fields ranging from image recognition to engineering.

Despite their success, purely data-driven models can produce unrealistic or inconsistent predictions when extrapolating beyond observed data. This limitation motivates the integration of physical knowledge into learning frameworks. Physics-Informed Neural Networks (PINNs) address this challenge by embedding differential and algebraic equations directly into the training process, enabling models to learn from data while respecting the governing laws of the system.

We chose a PINN-based approach over classical numerical solvers because it offers several advantages: it can infer unknown parameters directly from limited or noisy measurements, does not require repeated high-fidelity simulations for each scenario, and naturally combines observed data with physical constraints. In contrast, traditional numerical methods can be computationally intensive, sensitive to discretization errors, and less flexible when handling sparse or uncertain data.

In this work, we employ a pre-trained PINN framework to model a mean-value diesel engine system. This approach allows accurate prediction of state dynamics and estimation of unknown parameters, even with limited measurements and complex nonlinearities, while ensuring that predictions remain physically consistent

2 Relevant Work

Our work is primarily motivated by the Physics-Informed Neural Network (PINN) framework developed in [4], which demonstrated the simultaneous training of all state variables for multi-physics systems. While effective, their approach does not reduce the problem dimensionality and does not normalize the loss terms according to the magnitude of each state variable. This can lead to imbalanced optimization and biased learning, particularly in multi-scale systems. In contrast, our approach explicitly normalizes each component of the loss function, ensuring balanced contributions from all state variables, and is designed to perform robustly under sparse data conditions, which are common in practical diesel engine modeling.

Another relevant line of work involves Deep Operator Networks (DeepONets) [3], which learn the solution operators of complex dynamical systems in a fully data-driven manner. While highly flexible, DeepONets require large amounts of high-fidelity training data and incur significant computational cost during training. This limits their practicality in scenarios with limited experimental or simulated data, in contrast to our PINN-based methodology that leverages physics-based constraints to maintain predictive accuracy under sparse data.

3 Methodology

3.1 Mean Diesel Engine System

The system we are going to deal with is a diesel engine, which is initially proposed in Ref[6]. In summary, the engine model mainly consists of a variable – geometry turbocharger (VGT), an exhaust gas recirculatory (EGR), the intake manifold, the exhaust manifold, the cylinder and an EGR valve system.

The control input vector of the system is $u = \{u_\delta, u_{egr}, u_{vgt}\}$, -in which u_δ is the mass of injected fuel, u_{egr} is the EGR valve position and u_{vgt} is the VGT actuator position. The engine speed n_e is considered an input vector as well. Also, this model includes six states: the intake manifold pressure p_{im}, the exhaust manifold pressure p_{em}, the turbo speed ω_t, the VGT actuator dynamics $\tilde{u}_{vgt}$ and both mass fractions in the intake and exhaust manifold $X_{O_{im}}$, $X_{O_{em}}$ respectively. These mass fractions are assumed to be constant and known.

The governing ordinary differential equations that are used to describe the gas flow dynamics are the following:

$$\frac{dp_{im}}{dt} = \frac{R_a T_{im}}{Vim}(W_c + W_{egr} - W_{ei}) \tag{1}$$

$$\frac{dp_{em}}{dt} = \frac{R_e T_{em}}{Vem}(W_{eo} - W_t - W_{egr}) \tag{2}$$

$$\frac{d\omega_t}{dt} = \frac{P_t \eta_m - P_c}{J_t \omega_t} \tag{3}$$

$$\frac{d\tilde{u}_{egr1}}{dt} = \frac{1}{\tau_{egr1}}[u_{egr}(t - \tau_{degr}) - \tilde{u}_{egr1}] \tag{4}$$

$$\frac{d\tilde{u}_{egr2}}{dt} = \frac{1}{\tau_{egr2}}[u_{egr}(t - \tau_{degr}) - \tilde{u}_{egr2}] \tag{5}$$

$$\frac{d\tilde{u}_{vgt}}{dt} = \frac{1}{\tau_{vgt}}[u_{egr}(t - \tau_{dvgt}) - \tilde{u}_{vgt}] \tag{6}$$

The are also two important equations, that indicate a relationship between residual gas fraction x_r and the temperature T_1 when the inlet valve closes after the intake stroke and mixing. The two equations are significant for calculating terms of the forementioned equations of the state variables and also play a crucial role for the solution of our problem. They have the following expressions:

$$T_1 = x_r T_e + (1 - x_r) T_{im} \tag{7}$$

$$x_r = \frac{\Pi_e^{1/\gamma_\alpha}\, x_p^{-1/\gamma_\alpha}}{r_c\, x_v} \tag{8}$$

Moreover, the main parameters of interest we are interested in predicting are : the compensation factor of non-ideal cycles η_{sc}, the total heat transfer coefficient of the exhaust pipes

h_{tot}, the maximum effective area of the Variable Geometry Turbocharger (VGT) A_{vgtmax} and the maximum effective area of the Exhaust Gas Recirculation (EGR) valve A_{egrmax}. These parameters are embedded within the governing equations of the system.

Further mathematical and physical details regarding equations 1–8 and unknown parameters can be found in Ref. [6].

Unknown	$\boldsymbol{\eta_c}$	$\boldsymbol{h_{tot}}$	$\boldsymbol{A_{egrmax}}$	$\boldsymbol{A_{vgtmax}}$
Value	1.1015	96.2755	4.0×10^{-4}	8.4558×10^{-4}

Table 1: Theoretical values of unknown parameters for the diesel engine system.

3.2 PINN Framework

Let's assume that the actual value of a variable the neural network is trying to predict is y. Then, we symbolize as $\hat{y}$ the predicted value of the network. If the input of the neural network is time t and its trainable parameters are $\boldsymbol{\theta}$ then $\hat{y} = \hat{y}(t; \boldsymbol{\theta})$.

The total loss function of a PINN can be defined as the sum of three quantities, the initial condition loss, the equation loss and the data loss. In fact, data mismatch term is applied only in case there is data provided for the variable we want to predict. In other cases, in can be neglected. Considering that data is necessary as long as the solution of an ODE cannot be solved analytically, the total loss is given by:

$$\mathcal{L}_{PINN} = \underbrace{\left(\hat{y}(t_0; \boldsymbol{\theta}) - y_0\right)^2}_{\text{Initial Condition Loss}} + \underbrace{\frac{1}{N}\sum_{i=1}^{N} \left|r(t_i; \boldsymbol{\theta})\right|^2}_{\text{Equation Residual Loss}} + \underbrace{\frac{1}{M}\sum_{i=1}^{M} \left|\hat{y}(t_i; \boldsymbol{\theta}) - y(t_i)\right|^2}_{\text{Data Loss}} \tag{9}$$

where $r(t_i; \boldsymbol{\theta})$ is the residual of the corresponding differential equation, N is the number of residual points and M represents the total number of measurements (data points). Note that the number of residual points N might differ from the number of measurements M.

The loss function presented in equation 9 is formulated for approximating the solution of a single ordinary differential equation (ODE). In our case, however, the physical system is governed by eight equations in total and the corresponding state variables exhibit significantly different magnitudes. To address this imbalance and ensure a stable and effective training process, each component of the loss function is multiplied by a corresponding weight. Consequently, for systems with multiple unknown fields, the total loss function can be generalized as follows:

$$\mathcal{L}_{\text{Total}} = \sum_{l=1}^{L} \left[\lambda_{l,IC} \left(\hat{y}_l(t_0; \boldsymbol{\theta_l}) - y_{0,l}\right)^2 + \frac{1}{N_l} \lambda_{l,Eq} \left|r_l(t_i; \boldsymbol{\theta_l})\right|^2 \right] + \sum_{k=1}^{K} \left[\frac{1}{M_k} \sum_{i=1}^{M_k} \lambda_{k,Data} \left|\hat{y}_k(t_i; \boldsymbol{\theta_k}) - y_k(t_i)\right|^2 \right] \tag{10}$$

where $\lambda_{l,IC}$, $\lambda_{l,Eq}$, $\lambda_{k,Data}$ are the corresponding weights for initial condition, equation and data loss terms respectively.

An important aspect to emphasize is that the set of trainable parameters $\boldsymbol{\theta}$ of a single PINN comprises not only the weights $\mathbf{W}$ and biases $\mathbf{b}$ of the neural network but also the unknown physical parameters $\boldsymbol{\Lambda}$. Formally, this can be written as

$$\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{b}, \boldsymbol{\Lambda}\}, \quad \boldsymbol{\Lambda} = \{\eta_{sc}, h_{tot}, A_{vgtmax}, A_{egrmax}\}.$$

3.3 Surrogate Neural Networks

As previously mentioned, a subset of the analytical functions appearing in the governing equations are replaced with data-driven surrogate neural networks. These networks are pre-trained prior to integration into the PINN framework, thereby handling highly nonlinear functional dependencies that would otherwise complicate the residual loss evaluation.

The training of each surrogate network is formulated as a standard regression problem. For the i-th surrogate, the objective function is defined as

$$\mathcal{L}_i(\boldsymbol{\theta}_i^P) = \frac{1}{n_i} \sum_{j=1}^{n_i} \left(y_i^{(j)} - \mathcal{N}_i^{(P)}(\mathbf{x}_i^{(j)}; \boldsymbol{\theta}_i^P) \right)^2, \quad i = 1, 2, \ldots, 6, \tag{11}$$

where $\mathcal{N}_i^{(P)}$ denotes the surrogate network parameterized by $\boldsymbol{\theta}_i^P$, trained with input–output pairs $\{\mathbf{x}_i^{(j)}, y_i^{(j)}\}_{j=1}^{n_i}$.

Empirical Data Generation. Labelled data for surrogate training are generated using empirical formulae derived from the diesel engine system's physical modeling. These formulae provide target values $y_i^{(j)}$ over the operational domain. Table 2 summarizes the quantities of interest, the surrogate networks assigned to each, and their generating expressions.

Surrogates and Inputs. The symbolisms, inputs, and corresponding analytical expressions approximated by each surrogate are provided in Table 3.

Network Architectures. Table 4 details the architectures and hyperparameters employed for pre-training. Each network was trained with a mean-squared error loss, a learning rate as specified in the table, and additive Gaussian white noise at 4% applied to the data to promote robustness.

3.4 Training Process

The full training pipeline involves both the pre-trained surrogates and the system PINNs tasked with predicting the state variables' dynamics.

Data Generation. All training and evaluation data were generated using a high-fidelity Simulink model of the diesel engine system. In total, 2200 samples were produced, covering both the dynamics of the state variables and the auxiliary quantities required for surrogate training. The dataset was partitioned as follows:

#	Quantity	Symbol	Surrogate	Empirical Formula
1	Volumetric efficiency	η_{vol}	$\mathcal{N}_1^{(P)}$	$\frac{120\, W_{ei}\, R_a\, T_{im}}{p_{im}\, n_e\, V_d}$
2	EGR effective area ratio	f_{egr}	$\mathcal{N}_2^{(P)}$	$\frac{W_{egr}\sqrt{T_{em}R_e}}{A_{egr,max}\, p_{im}\, \Psi_{egr}}$
3	VGT area $\times$ choking	$f_{vgt} f_{\Pi_t}$	$\mathcal{N}_3^{(P)}$	$\frac{W_t\sqrt{T_{em}R_e}}{A_{vgt,max}\, p_{em}}$
4	Turbine mech. efficiency	η_{tm}	$\mathcal{N}_4^{(P)}$	$\frac{P_t\, \eta_m}{W_t\, c_{pe}\, T_{em}(1-\Pi_t^{1-1/\gamma_e})}$
5	Compressor efficiency	η_c	$\mathcal{N}_5^{(P)}$	$\frac{T_{amb}(\Pi_c^{1-1/\gamma_a}-1)}{T_c - T_{amb}}$
6	Volumetric flow	Φ_c	$\mathcal{N}_6^{(P)}$	$\frac{R_a T_{amb}}{p_{amb}\pi R_c^3 \omega_t} W_c$

Table 2: Empirical formulae used to generate labelled data for pre-trained surrogate neural networks.

Variable	$\boldsymbol{\eta_{vol}}$	$\boldsymbol{f_{egr}}$	$\boldsymbol{F_{vgt,\Pi_t}}$	$\boldsymbol{\eta_{tm}}$	$\boldsymbol{\eta_c}$	$\boldsymbol{\Phi_c}$
Surrogate	$\mathcal{N}_1^{(P)}(\boldsymbol{x}, \boldsymbol{\theta_1^P})$	$\mathcal{N}_2^{(P)}(\boldsymbol{x}, \boldsymbol{\theta_2^P})$	$\mathcal{N}_3^{(P)}(\boldsymbol{x}, \boldsymbol{\theta_3^P})$	$\mathcal{N}_4^{(P)}(\boldsymbol{x}, \boldsymbol{\theta_4^P})$	$\mathcal{N}_5^{(P)}(\boldsymbol{x}, \boldsymbol{\theta_5^P})$	$\mathcal{N}_6^{(P)}(\boldsymbol{x}, \boldsymbol{\theta_6^P})$
Input ($\boldsymbol{x}$)	$\{p_{im}, n_e\}$	$\{\tilde{u}_{egr}\}$	$\{\tilde{u}_{vgt}, \Pi_t\}$	$\{\omega_t, T_{em}, \Pi_t\}$	$\{W_c, \Pi_c\}$	$\{T_{amb}, \Pi_c, \omega_t\}$

Table 3: Pre-trained NN surrogates for the approximation of analytical functions.

Pre-Trained NNs	Network size	Output	Transformation	Learning rate
$\mathcal{N}_1^P(\boldsymbol{x}_1^P;\boldsymbol{\theta}_1^P)$	[2,10,10,10,1]	η_{vol}	-	10^{-3}
$\mathcal{N}_2^P(\boldsymbol{x}_2^P;\boldsymbol{\theta}_2^P)$	[1,32,32,32,1]	f_{egr}	$S(\boldsymbol{x_2^P})$	10^{-3}
$\mathcal{N}_3^P(\boldsymbol{x}_3^P;\boldsymbol{\theta}_3^P)$	[2,32,32,1]	F_{vgt,Π_t}	-	5×10^{-3}
$\mathcal{N}_4^P(\boldsymbol{x}_4^P;\boldsymbol{\theta}_4^P)$	[2,16,16,16,16,1]	η_{tm}	-	10^{-3}
$\mathcal{N}_5^P(\boldsymbol{x}_5^P;\boldsymbol{\theta}_5^P)$	[2,4,4,4,1]	η_c	-	10^{-3}
$\mathcal{N}_6^P(\boldsymbol{x}_6^P;\boldsymbol{\theta}_6^P)$	[3,20,20,20,1]	Φ_c	$S(\boldsymbol{x_6^P;\theta_6^P})$	10^{-3}

Table 4: Pre- trained Neural network architectures, outputs, transformations and learning rates.

- **Surrogate networks:** 80% training and 20% validation, selected randomly with enforced sparsity to ensure domain coverage.
- **PINNs:** 301 samples for training and 60 samples for validation (time window: $T = 60\,s$ with $\Delta t = 0.2\,s$). The remaining samples were used exclusively for testing and evaluation.

PINN Training Strategy. A total of six PINNs were trained, with time t as the sole input. Their architectures, outputs, and scaling factors are presented in Table 5.

PINNs	Network size	Output	Outputs Transformation	Scaling
$\mathcal{N}_1(t;\theta_1)$	[1,128,128,128,2]	p_{im}, p_{em}	$S_p(p_{im}) + 0.5,\ S_p(p_{im})$	$\times 10^5$
$\mathcal{N}_2(t;\theta_2)$	[1,128,128,1]	x_r	$S_p(x_r)$	$\times 0.03$
$\mathcal{N}_3(t;\theta_3)$	[1,128,128,128,1]	T_1	$S_p(T_1)$	$\times 300$
$\mathcal{N}_4(t;\theta_4)$	[1,64,64,64,2]	u_{egr1}, u_{egr2}	$S(\tilde{u}_{egr1}, \tilde{u}_{egr2})$	$\times 100$
$\mathcal{N}_5(t;\theta_5)$	[1,64,64,1]	ω_t	$S_p(\omega_t)$	$\times 5 \times 10^3$
$\mathcal{N}_6(t;\theta_6)$	[1,65,64,1]	$\tilde{u}_{vgt}$	$S(\tilde{u}_{vgt})$	$\times 100$

Table 5: Neural network architectures, outputs, transformations and scaling factors.

The training was conducted in two phases:

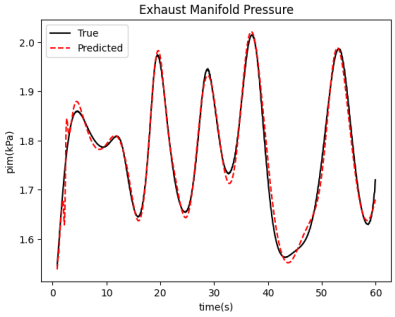
1. **Decoupled Training:** The networks predicting $\tilde{u}_{egr1}$, $\tilde{u}_{egr2}$, and $\tilde{u}_{vgt}$ were trained first. These variables are decoupled from the other system equations, enabling efficient standalone training.
2. **Integrated Training:** Once satisfactory convergence was achieved, the parameters of these decoupled networks were frozen and reused as pre-trained modules. The remaining PINNs were then trained jointly to minimize the composite loss function described in Section 3.2. This staged approach reduces computational complexity and enhances stability during optimization.

Outputs and Scaling. To stabilize training, outputs of each PINN were normalized through appropriate scaling factors (see Table 5). In addition, sigmoid or softplus trans-

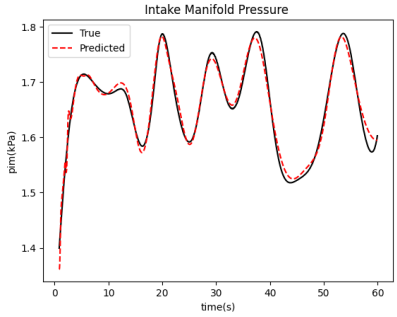
formations were applied where positivity or boundedness of the physical quantity had to be preserved.

4 Results

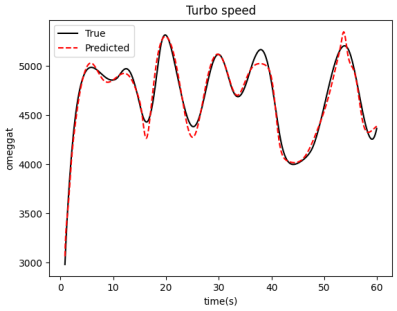
4.1 State Prediction



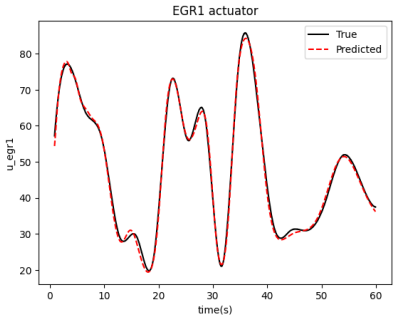
(a) $p_{em}(t)$



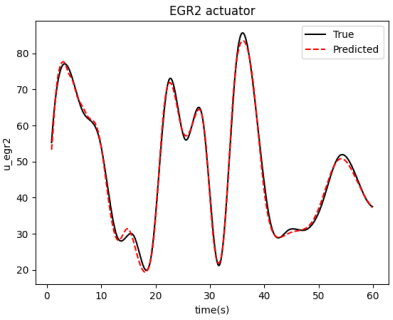
(b) $p_{im}(t)$



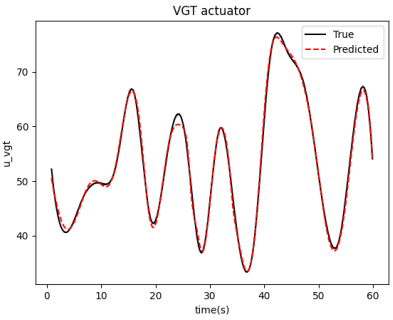
(c) ω_t



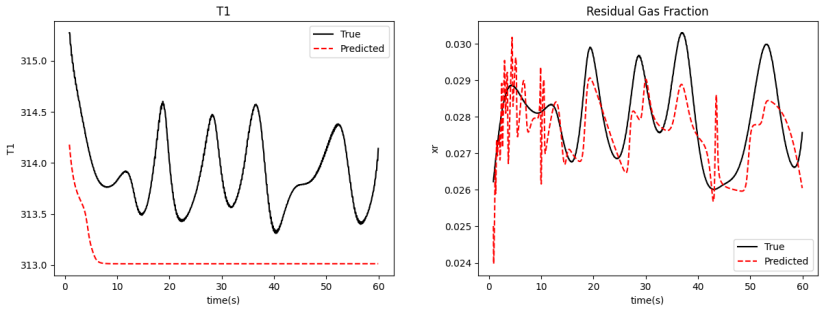
(d) $\tilde{u}_{egr1}$



(e) $\tilde{u}_{egr2}$



(f) $\tilde{u}_{vgt}(t)$



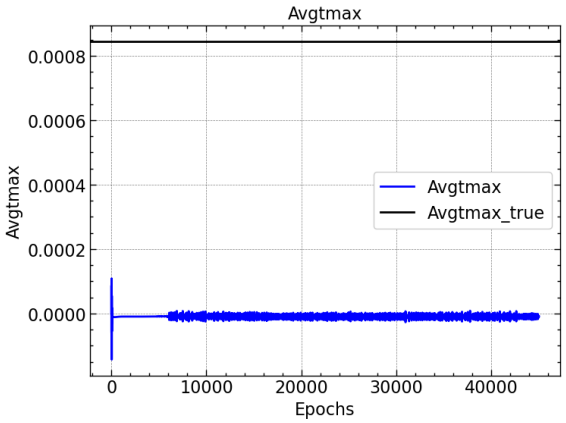
(g) $T_1(t)$ (h) $x_r(t)$

Figure 1: Performance of the model showing state variable dynamics over time while 4% white noise was applied on clean data signals for p_{im}, p_{em}, ω_t and W_{egr}. The training process follwed as well as the data sizes for training, testing and evaluation dataset are the same with the ones we discussed in the case were clean data were provided for training. The model seems to poorly on predicting variable x_r while it completely fails to predict variable $T1$. The rest of the variables seem to be satisfactorily predicted even under noisy conditions.

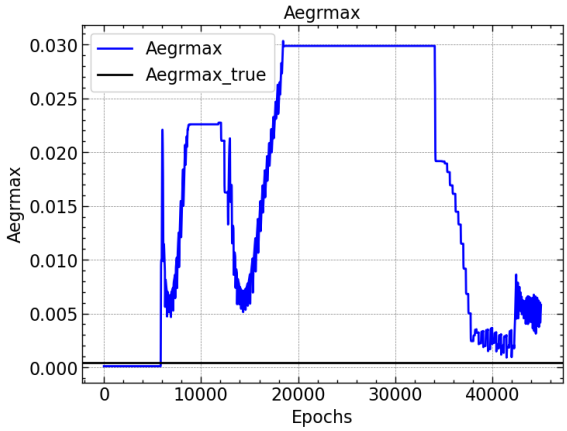
Variables	p_{im}	p_{em}	ω_t	u_{egr1}	u_{egr2}	u_{vgt}	T_1	x_r
Maximum Noise (4%) — Evaluation								
L2 Error	5.94e-03	7.64e-03	1.15e-02	2.04e-02	2.05e-02	1.18e-02	2.73e-03	2.87e-02
R^2 Score	0.984	0.987	0.980	0.996	0.996	0.997	-4.40	0.473
Maximum Sparsity (2 s) — Evaluation								
L2 Error	3.54e-02	4.58e-02	1.62e-02	1.07e-01	9.32e-02	1.25e-01	8.08e-04	3.70e-02
R^2 Score	0.653	0.604	0.992	0.892	0.917	0.600	0.762	0.222

Table 6: PINN evaluation performance under the most challenging conditions. The first block corresponds to maximum noise (4%), and the second block to maximum sparsity (2s time step). Values report L2 error and R^2 score for each state variable. T_1 and x_r exhibit the largest degradation.

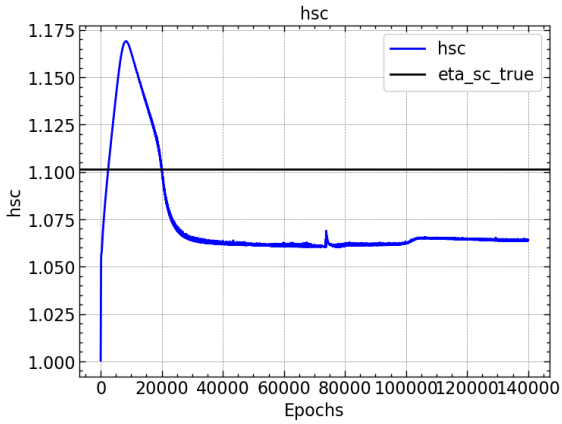
4.2 Parameter Estimation



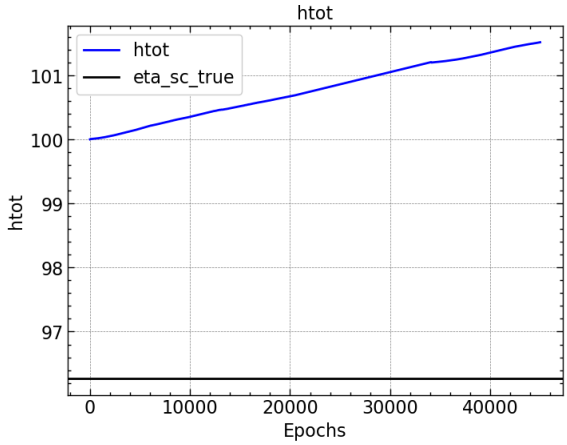
(a) A_{vgtmax}



(b) A_{egrmax}



(c) h_{sc}



(d) h_{tot}

Figure 2: Progressive predictions for unknown parameters' values during the training process with noisy data. Comparing with the case where the model was trained with clean data, in this case we observe that all the trainable parameters are at least updated during the training process. Parameters A_{egrmax} and h_{sc} somehow approximate the actual true values while model completely fails to approximate parameters A_{vgtmax} and h_{tot}. More specifically, for the parameters that the model has managed to approximate we have the following values: $A_{egrmax} = 4.9 \times 10^{-3}$ and $h_{sc} = 1.065$.

4.3 Surrogates

Dataset	Noise	η_{vol}	f_{egr}	F_{vgt}	η_c	η_{tm}	F_c
Mean Squared Error (MSE)							
Training	4%	$4.26 \cdot 10^{-5}$	$5.65 \cdot 10^{-3}$	$6.13 \cdot 10^{-4}$	$2.78 \cdot 10^{-4}$	$6.07 \cdot 10^{-4}$	$1.02 \cdot 10^{-4}$
Testing	4%	$4.52 \cdot 10^{-5}$	$4.87 \cdot 10^{-3}$	$5.79 \cdot 10^{-4}$	$1.56 \cdot 10^{-4}$	$5.90 \cdot 10^{-4}$	$8.65 \cdot 10^{-5}$
R^2 **Score**							
Training	4%	−3.85	0.688	0.909	0.878	0.471	0.973
Testing	4%	−5.65	0.717	0.880	0.763	0.255	0.954

Table 7: Prediction performance of the pre-trained surrogate NNs under 4% noisy conditions. MSE and R^2 scores are reported for training and testing datasets. Variables η_{tm} and η_{vol} are the most sensitive to noise.

4.4 Conclusion and Discussion

The proposed PINN model demonstrates satisfactory performance in predicting diesel engine state dynamics under noisy and sparse data conditions. With clean data, it captures all state variables accurately, although only two of the four unknown parameters are reliably estimated. Among the state variables, x_r and T_1 are the most challenging to predict, particularly under noise and sparsity, while the remaining variables are robustly approximated. Parameter estimation is limited due to the manual weighting strategy in the loss function, which can create imbalanced optimization dynamics, particularly under noisy or sparse observations. Adaptive loss balancing could improve parameter inference in future work.

The staged training approach effectively reduced model complexity and improved predictions for the more sensitive variables, albeit at the cost of increased computational time. Limitations were also imposed by the use of basic CPU resources, which restricted hyperparameter exploration and model experimentation. Compared to purely data-driven models such as DeepONets, the PINN approach requires significantly less data by leveraging known physical laws, making it particularly advantageous in scenarios with scarce or expensive measurements, though it may generalize less across unseen input functions.

Future work should focus on validating the model under broader environmental conditions, including variations in temperature, pressure, fuel composition, and humidity. Such studies would further assess robustness, reliability, and practical applicability for engine monitoring and control in real-world scenarios.

References

[1] John B Heywood. Combustion engine fundamentals. *1ª Edição. Estados Unidos*, 25:1117–1128, 1988.

[2] George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

[3] Varun Kumar, Somdatta Goswami, Daniel Smith, and George Em Karniadakis. Real-time prediction of gas flow dynamics in diesel engines using a deep neural operator framework. *Applied Intelligence*, 54(1):14–34, 2024.

[4] Kamaljyoti Nath, Xuhui Meng, Daniel J Smith, and George Em Karniadakis. Physics-informed neural networks for predicting gas flow dynamics and unknown parameters in diesel engines. *Scientific Reports*, 13(1):13683, 2023.

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

[6] Johan Wahlström and Lars Eriksson. Modelling diesel engines with a variable-geometry turbocharger and exhaust gas recirculation by optimization of model parameters for capturing non-linear system dynamics. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 225(7):960–986, 2011.

[7] Johan Wahlström and Lars Eriksson. Software packages for vehicular systems. `http://www.fs.isy.liu.se/Software`. Accessed: 2025-04-01.