mathematics

MDPI

Article

Fractional-Order System Identification: Efficient Reduced-Order Modeling with Particle Swarm Optimization and AI-Based Algorithms for Edge Computing Applications

Ignacio Fidalgo Astorquia [1,*], Nerea Gómez-Larrakoetxea [2], Juan J. Gude [1] and Iker Pastor [1]

[1] Department of Computing, Electronics and Communication Technologies, University of Deusto, Avenida de las Universidades 24, 48007 Bilbao, Spain; jgude@deusto.es (J.J.G.); iker.pastor@deusto.es (I.P.)
[2] DeustoTech-Deusto Institute of Technology, University of Deusto, Avenida de las Universidades 24, 48007 Bilbao, Spain; ngomez@deusto.es
* Correspondence: ignacio.fidalgo@deusto.es

Abstract: Fractional-order systems capture complex dynamic behaviors more accurately than integer-order models, yet their real-time identification remains challenging, particularly in resource-constrained environments. This work proposes a hybrid framework that combines Particle Swarm Optimization (PSO) with various artificial intelligence (AI) techniques to estimate reduced-order models of fractional systems. First, PSO optimizes model parameters by minimizing the discrepancy between the high-order system response and the reduced model output. These optimized parameters then serve as training data for several AI-based algorithms—including neural networks, support vector regression (SVR), and extreme gradient boosting (XGBoost)—to evaluate their inference speed and accuracy. Experimental validation on a custom-built heating system demonstrates that both PSO and the AI techniques yield precise reduced-order models. While PSO achieves slightly lower error metrics, its iterative nature leads to higher and more variable computation times compared to the deterministic and rapid inference of AI approaches. These findings highlight a trade-off between estimation accuracy and computational efficiency, providing a robust solution for real-time fractional-order system identification on edge devices.

Keywords: fractional-order systems; reduced-order modeling; Particle Swarm Optimization; artificial intelligence; edge computing; real-time control

MSC: 93C99

Academic Editors: Marjan Mernik, Jonathan Blackledge, Zheyi Chen, Zhengxin Yu and Wang Miao

Received: 28 February 2025
Revised: 3 April 2025
Accepted: 13 April 2025
Published: 16 April 2025

Citation: Fidalgo Astorquia, I.; Gómez-Larrakoetxea, N.; Gude, J.J.; Pastor, I. Fractional-Order System Identification: Efficient Reduced-Order Modeling with Particle Swarm Optimization and AI-Based Algorithms for Edge Computing Applications. *Mathematics* **2025**, *13*, 1308. https://doi.org/10.3390/math13081308

1. Introduction

In the field of industrial control, it is essential to characterize process dynamics to properly design and tune closed-loop controllers. For this purpose, simplified mathematical models are used to capture key aspects of the system without introducing unnecessary complexity [1]. An appropriate reduced-order model must meet two key requirements: (a) it must be easily identifiable from experimental data, and (b) it must accurately represent the process behavior near its operating point [2].

Within this context, the proportional-integral-derivative (PID) controller remains the predominant choice in industrial settings due to its effectiveness and ease of implementation [3]. For tuning purposes, reduced-order models are commonly used, with the first-order plus dead-time (FOPDT) model being the most widely employed in industrial settings. Model accuracy significantly impacts control loop performance, and previous

studies have shown that using more representative models can improve PID tuning rules, leading to optimized system responses [4].

In recent years, advancements in fractional-order calculus and computational methods have facilitated the development of sophisticated identification techniques for modeling processes governed by fractional differential equations [5]. Fractional-order systems extend classical integer-order models by incorporating derivatives and integrals of non-integer orders. This generalization offers greater modeling flexibility and more accurate representations of real-world dynamic behaviors [6]. One of the main advantages of these models is their ability to more accurately capture memory effects and hereditary dynamics, which are prevalent in many physical systems, including thermal, electrical, mechanical, and biological domains. Fractional derivatives enable the modeling of long-term memory effects and non-local behavior, providing a more accurate representation of real-world dynamics compared to integer-order models. Additionally, fractional models offer greater flexibility in parameter tuning based on experimental data, allowing for a more precise characterization of processes whose dynamic responses cannot be adequately captured by traditional models.

As a result, the use of fractional models in industrial environments has been increasing. For a comprehensive overview of fractional-order systems in automatic control, see [7]. The FOPDT model is widely used in practice for characterizing process dynamics and designing control systems [3]. However, given the enhanced capability of fractional models to capture complex dynamics, the fractional first-order plus dead-time (FFOPDT) model extends the traditional FOPDT framework [8], offering a better balance between simplicity and accuracy in system identification.

An increasing number of methods have been developed for estimating fractional-order models based on the reaction curve. These techniques require only a step test to obtain the reaction curve and demand minimal system information, making them widely applicable in industrial settings.

Broadly, these methods generally fall into two categories: analytical and optimization-based approaches. Among analytical techniques, Tavakoli-Kakhki's pioneering work in [9] introduced analytical techniques for estimating FFOPDT model parameters directly from step response data.

Recent efforts have focused on adapting classical analytical identification methods for fractional-order processes. In particular, Ref. [10] examines Sundaresan's technique for modeling fractional-order systems, highlighting a limitation related to the convergence of the response derivative in the frequency domain. To address this issue, a corrective equation and a simplified formulation that avoids the inversion of the Mittag-Leffler function are proposed.

Within this context, Gude and García Bringas introduced an identification procedure for FFOPDT models based on three arbitrary points of the process reaction curve (x_1-x_2-x_3%) [11]. Building on this work, Ref. [12] introduced a simplified approach by exploiting the symmetrical distribution of three key points on the reaction curve (x-50-$(100-x)$%). Both techniques are designed for overdamped processes evaluated in an open-loop system subjected to a step input. Furthermore, new analytical techniques have been introduced to enhance the accuracy of fractional model estimation. In [13], a method is presented that estimates the fractional order of the model by leveraging the asymptotic behavior of the Mittag-Leffler function, while determining the parameters in the time domain using two arbitrary points, x_1 and x_3, from the process reaction curve. It has also been demonstrated that the accuracy of the identified model is influenced by the placement of these points.

In fractional-order model identification, nonlinear optimization is the most commonly used approach in industrial applications. This method seeks to minimize the discrepancy between the step response of the estimated model and the process reaction curve. Several techniques have been proposed to achieve this. For instance, Ref. [14] used curve fitting and analytical solutions in the time domain, incorporating the Mittag-Leffler function to identify fractional-order models. Similarly, Ref. [15] applied integral equations for model identification from step test data. Additionally, Ref. [16] utilized both the Mittag-Leffler and Grünwald–Letnikov methods to identify FFOPDT models based on step response measurements.

Another relevant approach uses the CRONE methodology, which was applied in [17] for fractional-order model identification. This method has also been used in the design of fractional-order PI and PID controllers, as discussed in [18].

Analytical identification methods are known for their simplicity and low computational cost, whereas optimization-based techniques are recognized for offering greater accuracy at the expense of higher computational demands. In the technical literature, there is an ongoing debate on this topic, highlighting the need to find a balance between accuracy, robustness, and computational efficiency. Expanding on this approach, the distinctiveness of the hybrid method presented in [19] lies in its combination of single-variable optimization to estimate the fractional order α with analytical expressions for determining the remaining FFOPDT model parameters.

Within this context, reduced-order models simplify complex dynamic systems while preserving key behaviors, making them highly valuable for real-time applications and resource-constrained edge devices. Their main advantages include the following:

- Computational efficiency: They simplify system representation for faster calculations.
- Lower memory and processing requirements: This makes them suitable for embedded controllers and IoT devices.
- Faster response times: They help ensure that control algorithms meet strict timing constraints.
- Improved controller design and model interpretability: By highlighting essential dynamics, they facilitate controller development and enhance model understanding.
- Reduced power consumption: This is especially crucial for battery-powered systems.

Due to these advantages, reduced-order modeling is widely used in industrial control.

Traditional analytical methods for fractional-order system identification often rely on approximations that, while computationally efficient, can lead to reduced accuracy when capturing complex system dynamics. Optimization-based approaches improve accuracy but are computationally expensive, making them unsuitable for real-time deployment in edge computing scenarios. The proposed hybrid framework leverages PSO to refine parameter estimation with high fidelity while employing AI-based inference for rapid, deterministic execution. By combining these techniques, the framework achieves a balance between precision and computational efficiency, outperforming purely analytical methods in accuracy and surpassing traditional optimization techniques in execution speed. This dual benefit enables real-time identification and control applications on resource-constrained devices, an aspect that is critical for industrial automation and embedded system implementations.

The main objective of this study is to derive a reduced-order fractional model from the dynamic response of a high-order system using an innovative combination of PSO optimization and AI-based algorithms. By leveraging information from the process reaction curve obtained through a simple step test, the proposed method enhances estimation accuracy, computational efficiency, and model generalization. Specifically designed for overdamped step responses, it integrates PSO with AI techniques to improve parameter

estimation for the FFOPDT model. Its effectiveness is demonstrated through illustrative examples, highlighting advantages over analytical and optimization-based methods. Finally, the identification algorithm is implemented on microprocessor-based hardware, validating its applicability to real-world thermal process identification in experimental prototypes.

The key contributions of this work can be summarized as follows:

1. A hybrid identification framework that combines PSO-based optimization with AI-driven inference to derive reduced-order models from high-order fractional dynamic systems.
2. An innovative edge computing integration that enables real-time deployment of the identification framework, ensuring low latency and efficient resource usage.
3. A comprehensive experimental validation using a custom-built heating system, demonstrating the proposed method's effectiveness in accurately capturing the dynamics of overdamped systems and enabling real-time control on edge devices.
4. A detailed comparative analysis of PSO and AI-based approaches, highlighting the trade-offs between accuracy and computational efficiency, and providing insights for deploying these methods in resource-constrained environments.

The remainder of this paper is organized as follows. Section 2 presents the theoretical background, including the fundamentals of fractional calculus, the formulation of fractional-order systems, and an overview of the optimization techniques employed, with a particular focus on PSO and various AI-based approaches. Section 3 details the materials and methods used in this study, covering data generation and balancing strategies, as well as the implementation of both PSO-based parameter identification and AI-based estimation methods. Section 4 describes the experimental setup and presents the results, providing a comprehensive evaluation of model accuracy and computational performance across different edge devices. The Discussion section (Section 5) examines the trade-offs between estimation accuracy and execution speed, exploring the practical implications of deploying these methods in real-time edge computing environments. Finally, Section 6 summarizes the key findings, outlines the theoretical and practical contributions of this work, and suggests directions for future research.

2. Theoretical Background

This section provides an overview of the fundamental concepts and methodologies used in this work. It covers key aspects of fractional calculus, optimization algorithms, and AI-based techniques applicable to system identification.

The section is organized as follows. Section 2.1 presents the fundamental concepts of fractional calculus necessary for this work. Section 2.2 describes the PSO algorithm, used to estimate optimal FFOPDT model parameters by minimizing the error between its and the process's response. Finally, Section 2.3 introduces various AI-based techniques employed for the identification of dynamic systems.

2.1. Fractional Calculus

This section introduces fundamental concepts in fractional calculus, which extends classical differentiation and integration to non-integer orders. For a more in-depth discussion, refer to [20,21].

Fractional calculus generalizes the conventional derivative $\frac{d^n}{dt^n}$ to the non-integer-order operator ${}_aD_t^\alpha$, where $\alpha \in \mathbb{R}$ denotes the fractional order, and a and t represent the lower and upper limits of the differ-integral operator. This operator is defined as follows:

$$ {}_aD_t^{\alpha} = \begin{cases} \frac{d^{\alpha}}{dt^{\alpha}}, & \alpha > 0, \\ 1, & \alpha = 0, \\ \int_a^t (d\tau)^{\alpha}, & \alpha < 0, \end{cases} \tag{1} $$

Although α is typically a real number, it can also be complex. In this study, we focus exclusively on real-valued fractional orders.

The literature presents various definitions of fractional-order operators. Below is a brief overview of those used in this work.

Definition 1. *Riemann–Liouville:*

The Riemann–Liouville fractional operator is one of the most commonly used formulations in fractional calculus. It is important to note that this is the definition used for the mathematical development in this work. The fractional derivative of a function $f(t)$ is defined as follows:

$$ {}_0D_t^{\alpha} f(t) = \frac{1}{\Gamma(m-\alpha)} \frac{d^m}{dt^m} \int_0^t (t-\tau)^{m-\alpha-1} f(\tau) d\tau, \tag{2} $$

where m is the smallest integer greater than α (i.e., $m-1 < \alpha < m$, with $m \in \mathbb{Z}^+$), $\Gamma(\cdot)$ is the Gamma function [20], $t \geq 0$, and $\alpha \in \mathbb{R}^+$.

Definition 2. *Grünwald–Letnikov:*

The Grünwald–Letnikov operator provides a discrete approximation to fractional differentiation and integration, making it well-suited for numerical computations. It is defined as follows:

$$ {}_0D_t^{\alpha} f(t) = \lim_{h \to 0} \frac{1}{h^{\alpha}} \sum_{k=0}^{\infty} (-1)^k \binom{\alpha}{k} f(t-kh), \tag{3} $$

where $\alpha > 0$ corresponds to differentiation, $\alpha < 0$ to integration, $\binom{\alpha}{k}$ represents the binomial coefficient, and h is the step size.

Unlike the Riemann–Liouville formulation, the Grünwald–Letnikov approach aligns more closely with numerical methods, making it useful for computational implementations of fractional operators. This definition is the one used for the implementation of the fractional operators in the results of this work.

Definition 3. *Mittag-Leffler Function:*

The Mittag-Leffler function plays a fundamental role in fractional calculus, serving as a generalization of the exponential function. The one-parameter Mittag-Leffler function is defined as follows [22]:

$$ E_{\alpha}(z) = \sum_{r=0}^{\infty} \frac{z^r}{\Gamma(\alpha r + 1)}. \tag{4} $$

where $\Gamma(\cdot)$ is the Gamma function, $\alpha \in \mathbb{R}^+$, and $z \in \mathbb{C}$.

Because it generalizes exponential behavior, the Mittag-Leffler function is fundamental for modeling fractional-order systems, including the FFOPDT model [23].

Definition 4. *FFOPDT model and step response:*

The FFOPDT model is one of the most widely used reduced-order fractional models. Its transfer function is given as follows:

$$ P(s) = \frac{K \cdot e^{-Ls}}{1 + Ts^{\alpha}}, \tag{5} $$

where K is the process gain, $T > 0$ is the time constant, $L \geq 0$ is the apparent time delay, and α is the fractional order of the model. Note that the FFOPDT model reduces to the standard FOPDT model when $\alpha = 1$.

The time-domain expression for the response of the FFOPDT model to a step signal of magnitude Δu is the following:

$$y_\alpha(t) = \begin{cases} 0, & 0 \leq t < L \\ K\Delta u\left\{1 - E_\alpha\left[-\frac{1}{T}(t-L)^\alpha\right]\right\}, & t \geq L \end{cases} \tag{6}$$

where the output signal change is $\Delta y = K \cdot \Delta u$ and E_α represents the one-parameter Mittag-Leffler function [20].

2.2. PSO Optimization

PSO is a population-based stochastic optimization technique inspired by the social behavior of bird flocking and fish schooling [24]. In PSO, a swarm of candidate solutions (particles) is initialized in the parameter space, and each particle iteratively updates its position and velocity based on its own best-found solution and the global best solution discovered by the swarm. This process allows PSO to efficiently explore complex, multidimensional search spaces, making it particularly well-suited for challenging optimization problems in system identification [25].

In our work on fractional-order system identification, PSO is employed to identify the optimal parameters $\{K, T, L, \alpha\}$ that define the reduced-order fractional model. The optimization objective is to minimize the discrepancy between the step response of the high-order system and that of the reduced model, quantified by error metrics such as the mean squared error (MSE).

The PSO Algorithm 1 used in this study can be summarized as follows:

Algorithm 1: PSO algorithm for reduced-order parameter identification.

Data: High-order system output, parameter bounds for $\{K, T, L, \alpha\}$, random initialization of particles

Result: Optimized parameter set $\{K, T, L, \alpha\}$ minimizing the discrepancy between the high-order system and reduced model

Initialization: Generate a swarm of particles (candidate solutions) in the parameter space $\{K, T, L, \alpha\}$ randomly;

Assign initial velocities to each particle within predefined bounds;

Evaluate the fitness (e.g., MSE) for each particle;

Determine each particle's personal best (p_{best});

Identify the global best (g_{best}) among all particles;

while Stopping criterion not met (e.g., maximum iterations or minimal error improvement) **do**

  foreach particle in the swarm **do**

    Update velocity using inertia, cognitive (distance to p_{best}), and social (distance to g_{best}) components;

    Update particle position within parameter bounds;

    Evaluate the fitness of the updated position;

    Update personal best if current fitness is better than p_{best};

  Update the global best (g_{best}) among all particles;

The use of PSO in our framework is justified by its ability to handle non-convex, high-dimensional optimization problems without requiring gradient information, making it robust against local minima. This attribute, coupled with its proven success in previous system identification studies [19], underscores its suitability for our approach. By leveraging PSO, we ensure that the reduced-order model closely approximates the dynamic behavior of the high-order system, providing a reliable foundation for subsequent control and real-time implementation on edge devices.

2.3. AI-Based Algorithms for System Identification

AI is increasingly used to identify reduced-order models that approximate complex dynamical systems while preserving essential behavior at a lower computational cost. AI-driven techniques facilitate the estimation of these models from experimental or simulated data, improving efficiency in control and optimization tasks.

The use of AI in the modeling and control of fractional systems has gained significant attention in recent years. Fractional systems, characterized by the presence of non-integer-order derivatives, have proven to be more suitable for describing physical systems with memory and hysteresis. Identifying key parameters such as K, T, L, and α in these systems is essential for their control and optimization. In this context, various machine learning techniques have been employed to improve the accuracy of parameter estimation.

Raubitzek et al. (2023) [26] present a comprehensive review on the combination of fractional derivatives with AI. They classify the approaches into three main categories: data preprocessing, where fractional operators are used to enhance input features in learning algorithms; machine learning-based fractional modeling, approaches that integrate machine learning with fractional differential equations to improve the accuracy of system dynamics prediction; and hyperparameter optimization, where machine learning is used to fine-tune fractional models based on observational data.

2.3.1. Comparison with Traditional Optimization Techniques

Supervised learning has been a key tool for parameter estimation in fractional control models. Cheng et al. (2022) [27] apply fractional derivatives along with supervised learning techniques such as SVR and random forest to improve the estimation of the content of photosynthetic pigment in apple leaves. This work demonstrates that using fractional operators in combination with machine learning improves the accuracy of estimating relevant physical variables. Moreover, it has also been applied in the biomedical field, as Annadurai et al. (2024) [28] introduce an approach that integrates transfer learning techniques with fractional operators to enhance medical image quality. Their method, called ETLFOD, combines pre-trained deep neural network models (DenseNet121, ResNet50, etc.) with fractional operators for noise reduction in MRI, CT, and X-ray images. The application of fractional operators in this case helps preserve crucial image features, demonstrating their applicability in signal processing tasks.

2.3.2. AI-Based Parameter Estimation vs. Traditional Methods

While traditional optimization techniques have been widely used in system identification, AI-based approaches offer notable advantages:

- Robustness to nonlinearities: AI techniques handle nonlinearity better than traditional mathematical models.
- Flexibility and adaptability: Machine learning models can generalize across different datasets, improving the estimation of parameters such as K, T, L, and α.
- Higher efficiency in large datasets: AI-based optimization can process large volumes of data faster, whereas conventional methods may struggle with scalability.

Given these advantages, the integration of AI-based methodologies in system identification and parameter estimation represents a promising direction to enhance traditional optimization frameworks. This complements the PSO approach [24] by refining the parameter search process and improving convergence to optimal solutions.

2.3.3. Machine Learning Techniques

Machine learning has become an essential tool for modern data-driven applications, providing diverse methodologies for predictive modeling and optimization. Different algorithms have been used to address complex system identification problems, balancing interpretability, computational efficiency, and predictive performance. In this subsection, we describe the machine learning algorithms used in this experiment, highlighting their advantages and common applications in parameter estimation and predictive modeling.

Linear Regression

Linear regression is one of the most fundamental statistical methods used for predictive modeling. It assumes a linear relationship between the dependent variable and one or more independent variables, making it efficient and interpretable. However, its performance is often limited in cases where nonlinear relationships exist or when high multicollinearity is present in the data [29].

Random Forest

Random forest is an ensemble learning method based on decision trees, introduced by Breiman (2001). It builds multiple decision trees and combines their predictions to improve accuracy and reduce overfitting. This algorithm is widely used for classification and regression tasks due to its robustness to noise and ability to capture nonlinear relationships in data [30].

Neural Networks

Artificial neural networks (ANNs) are inspired by the structure of the human brain and have become a crucial technique in deep learning. They consist of multiple interconnected layers of artificial neurons that extract complex patterns from data. Although ANNs require large datasets and significant computational resources, they have demonstrated superior performance in predictive modeling and classification across various domains [31].

Deep Learning

Deep learning is an extension of ANNs with multiple hidden layers, often referred to as deep neural networks (DNNs). These models excel in recognizing complex patterns and modeling sequential or structured data. Methods such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have been highly effective in image processing and time-series forecasting [32].

Extreme Gradient Boosting—XGBoost

XGBoost is an optimized version of the boosting technique, designed for high-performance classification and regression tasks. It improves accuracy by combining weak models iteratively, minimizing errors and reducing overfitting. XGBoost has gained popularity in data science competitions due to its efficiency and effectiveness in handling structured data [33].

Support Vector Regression—SVR

SVR is an extension of support vector machines (SVMs) for regression tasks. It aims to find an optimal hyperplane that minimizes prediction errors while maintaining robustness against outliers. SVR is particularly effective for high-dimensional data and applications requiring strong generalization capabilities [34].

Gaussian Process Regression—GPR

GPR is a probabilistic approach that models distributions over functions, making it highly suitable for uncertainty quantification. Unlike traditional regression techniques, GPR

provides confidence intervals for predictions, making it particularly useful in applications where understanding the model's uncertainty is essential [35].

3. Materials and Methods

3.1. Simulation Setup

The simulation framework was designed to accurately capture the dynamics of fractional-order systems, with particular emphasis on overdamped systems. This framework ensures that both fast transient dynamics and slow steady-state behaviors are faithfully reproduced, providing a robust foundation for evaluating and optimizing system identification and control strategies in practical, real-world scenarios.

3.1.1. Software and Tools

The simulations were conducted using the FOMCON [36] Python 3.11 package, a specialized toolbox for the analysis and simulation of fractional-order systems. This package provides a robust environment for modeling dynamic systems with non-integer-order derivatives, ensuring an accurate representation of fractional dynamics. Its capabilities were essential for generating the high-fidelity simulations required for this study.

3.1.2. System Description

The dynamic system under investigation is a high-order, overdamped model characterized by the following commensurate transfer function:

$$P(s) = \frac{K}{(1 + T_1 s^\alpha)(1 + T_2 s^\alpha)(1 + T_3 s^\alpha)} \tag{7}$$

where the following definitions apply:

- Gain (K): A scalar value representing the system's amplification factor.
- Time constants (T_1, T_2, T_3): Parameters that define the system's transient response.
- Fractional order (α): A non-integer order characterizing the system's fractional dynamics.

This model structure is chosen for its versatility and proven ability to approximate a wide range of real systems. In this work, the focus is on overdamped systems, where the dynamic response exhibits non-oscillatory, exponentially decaying transients—a common behavior in thermal processes and other applications requiring stability and smooth responses. The combination of multiple time constants with a fractional derivative enables the model to capture both fast and slow dynamics, effectively representing transient and steady-state behaviors across diverse physical and engineering applications.

The primary objective of this research is to estimate an FFOPDT model from the step response of a high-order system. The FFOPDT model provides a simplified yet accurate representation of the system's dynamics, facilitating efficient real-time control, particularly in edge computing applications. In essence, the high-order model generates realistic, noisy data, from which our hybrid identification framework—integrating PSO-based optimization with AI-driven inference—derives the reduced model parameters.

3.2. Data Generation

3.2.1. Sampling Strategy

To ensure reproducibility and determinism, a fixed seed was initialized for the random number generator prior to parameter sampling. This approach guarantees that the same sequence of pseudo-random numbers is generated across independent simulation runs. The system parameters were generated using a uniform sampling strategy, ensuring a

systematic and unbiased exploration of the parameter space. The specific ranges for each parameter are shown below:

- Gain (K): Uniformly sampled from $[1, 100]$.
- Time constants (T_1, T_2, T_3): Independently sampled from a uniform distribution in $[0.1, 100]$.
- Fractional order (α): Uniformly sampled from $[0.2, 1]$.

The chosen parameter ranges are justified through a combination of rigorous theoretical analysis and extensive empirical testing. This ensures the dataset effectively covers a broad and realistic array of system behaviors, facilitating robust analysis and model training. Additionally, these ranges capture the variability in amplification factors observed in many practical dynamic systems [37–41].

3.2.2. Input and Output

A step input was applied to the high-order system. For each simulation run, the time horizon was dynamically determined by continuously monitoring the system's output until steady state was reached. Specifically, the simulation analyzed the transient response and extended the recording duration until the variation in the output fell below a predefined threshold over a sufficient period, indicating stabilization. The sampling frequency was carefully chosen to capture both the rapid transient dynamics immediately following the step input and the gradual approach to steady state.

3.2.3. Dataset Balancing

To ensure that the dataset uniformly covers the parameter space and represents the majority of possible combinations of system parameters, a comprehensive numeric space analysis was performed. Scatter plots were employed to evaluate the distribution of the sampled values for K, T_1, T_2, T_3, and α (see Figure 1). This analysis verified that the uniform sampling strategy effectively spanned the entire range of each parameter, minimizing bias and avoiding clustering in specific regions. Where underrepresented regions were identified, the sampling process was iteratively refined to improve coverage. This rigorous approach to dataset balancing ensures that the generated dataset is both comprehensive and representative, facilitating robust system identification and enhancing the generalization capability of the trained models.

3.3. Performance Metrics

The performance of the proposed identification methodologies is quantified using a range of statistical metrics that assess both model prediction accuracy and computational efficiency of the algorithms. These metrics are essential to ensure that the identified models are reliable and that the inference process remains feasible in real-time, edge-based applications. In the following subsection, we detail the computation of each metric and explain its importance in the context of dynamic system identification.

3.3.1. Mean Squared Error (MSE)

MSE quantifies the average squared difference between the measured output $y(t)$ and the predicted output $\hat{y}(t)$. It is computed as follows:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2, \tag{8}$$

where N is the total number of data points. MSE is important because it penalizes larger errors more than smaller ones, providing a sensitive measure of the overall discrepancy.

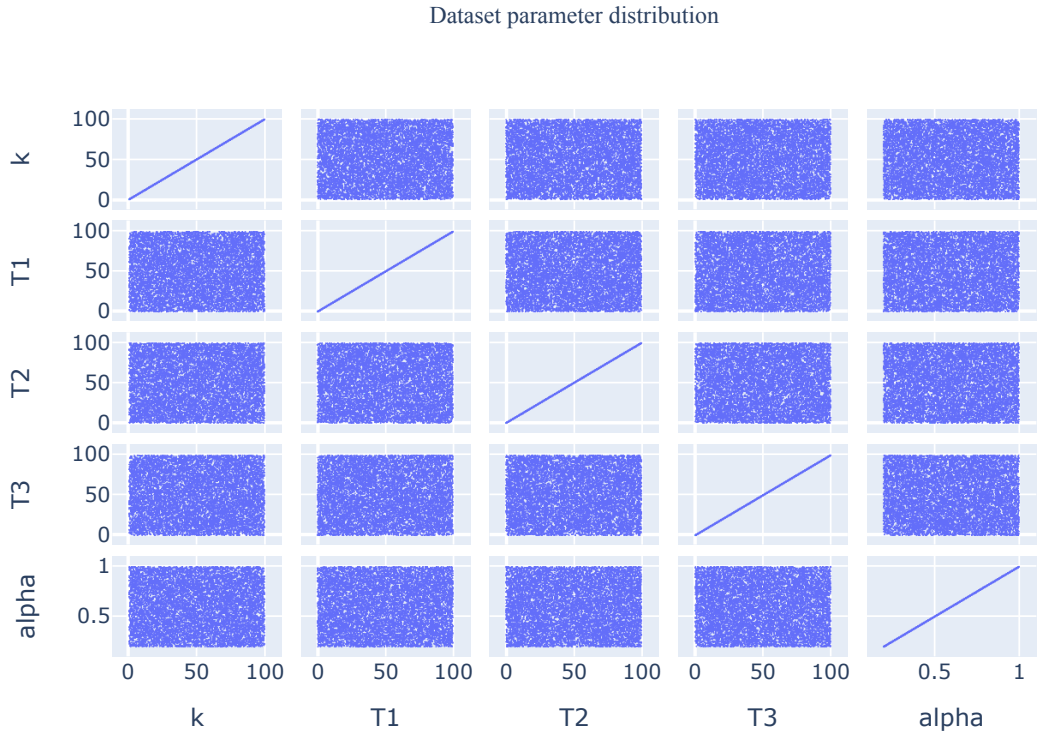

Figure 1. Pairwise scatter matrix visualization of the dataset parameters (K, T_1, T_2, T_3, and α). The diagonal elements represent the marginal distributions of each parameter, while the off-diagonal scatter plots illustrate their pairwise relationships.

3.3.2. Root Mean Squared Error (RMSE)

RMSE is the square root of the MSE:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}. \tag{9}$$

Expressed in the same units as the output, RMSE offers an intuitive measure of the error magnitude, making it particularly useful for comparing prediction errors relative to the natural scale of the measured output.

3.3.3. Mean Absolute Error (MAE)

MAE represents the average absolute difference between the measured and predicted output:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|. \tag{10}$$

Unlike MSE and RMSE, MAE is less sensitive to outliers, providing a robust measure of typical error. This robustness is crucial in dynamic system identification, where occasional noise or anomalies may occur.

3.3.4. Coefficient of Determination (R^2)

The coefficient of determination, R^2, measures the proportion of variance in the observed data that is explained by the model:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}, \tag{11}$$

where $\bar{y}$ is the mean of the observed values. A higher R^2 indicates a better model fit, validating that the identification method accurately captures the underlying system dynamics.

3.3.5. Relative Mean Absolute Error (RMAE)

To facilitate comparison across different scales, MAE is normalized by the range of observed values:

$$\text{RMAE} = \frac{\text{MAE}}{y_{\max} - y_{\min}}, \tag{12}$$

where $y_{\max}$ and $y_{\min}$ denote the maximum and minimum observed values, respectively. This scale-independent metric provides meaningful comparisons between systems with different output ranges.

3.3.6. Relative Root Mean Squared Error (RRMSE)

Similarly, RMSE is normalized to yield a relative measure:

$$\text{RRMSE} = \frac{\text{RMSE}}{y_{\max} - y_{\min}}. \tag{13}$$

RRMSE is crucial when comparing performance across different scenarios, as it accounts for variations in the dynamic range of the output.

3.3.7. Computation Time

In addition to accuracy metrics, the computation time required for the identification process is recorded. This time, measured in seconds, encompasses the entire duration of the parameter estimation and model evaluation phases. Monitoring computation time is crucial, particularly for edge applications where deterministic execution is essential.

3.3.8. Summary

Together, these performance metrics provide a comprehensive framework for evaluating both the accuracy and efficiency of the proposed identification methods:

- MSE, RMSE, and MAE offer insights into the absolute prediction errors, with RMSE and MSE being sensitive to larger errors.
- R^2 quantifies the proportion of variance explained by the model, serving as a measure of overall model fit.
- Relative error metrics (RMAE and RRMSE) allow for scale-independent comparisons.
- Computation time provides an evaluation of the algorithm's efficiency, an important factor for practical applications.

3.4. PSO-Based Parameter Identification

The reduced-order model parameters were estimated using PSO, a widely used optimization technique in system identification that provides accurate and robust parameter estimates [19]. Unlike in the cited text, where the focus may have been on optimizing parameters for a specific system, our goal is to develop a general set of parameters that can be applied to estimate any generic system. This approach leverages the PSO algorithm to iteratively update a swarm of candidate solutions, optimizing the parameters based on the MSE between the high-order system's output and the response of the reduced model. PSO has been widely used in system identification tasks, obtaining accurate and robust parameter estimates.

3.4.1. Objective and Implementation

The primary objective of the PSO-based parameter identification is to estimate the parameters of the reduced-order model—gain K, time constant T, time delay L, and frac-

tional order α—by minimizing the discrepancy between the high-order system's output and that of the reduced model. For each simulated sample, PSO determines the optimal set of parameters. The optimization process is driven by an objective function defined as the MSE (see Equation (8)), which quantifies the difference between the measured and predicted output:

- Measured output: $y_{\text{high}}(t_i)$ denotes the high-order system's output at time t_i.
- Predicted output: $y_{\text{red}}(t_i; K, T, L, \alpha)$ represents the reduced-order model's response computed using parameters K, T, L, and α.

The search space for the reduced-order parameters is constrained to the following intervals:

- Gain (K): $1 \leq K \leq 100$
- Time constant (T): $0.1 \leq T \leq 100$ s
- Time delay (L): $0 \leq L \leq 20$ s
- Fractional order (α): $0.2 \leq \alpha \leq 1$

The PSO algorithm iteratively updates a swarm of candidate solutions by evaluating the MSE for each candidate and adjusting their positions based on both the individual best positions and the global best found by the swarm. The algorithm is implemented using a custom framework based on the methodology described in [25]. To ensure convergence while avoiding excessive computation time, the following stopping criteria were enforced:

- A maximum of 300 iterations per optimization run.
- Convergence of the swarm, indicated by negligible parameter changes across successive iterations.

For each simulated sample, a swarm of 50 particles was used. These hyperparameters were selected based on empirical observations and extensive testing, ensuring efficient convergence and reliable performance.

3.4.2. Resulting Dataset

Upon completion of the PSO-based optimization for each simulated sample, the dataset is enriched by appending the optimal reduced-order model parameters alongside the high-order system's output and the corresponding time vector. Specifically, the resulting dataset includes the following data:

- The high-order system's output.
- The associated time vector.
- The optimal parameters derived from PSO $\{K, T, L, \alpha\}$.
- The reduced-order model's response computed using these optimal parameters.
- All performance metrics described in Section 3.3.

This augmented dataset provides a comprehensive basis for subsequent analyses and serves as a benchmark for evaluating and comparing the performance of AI-based parameter estimation methods.

3.5. AI-Based Algorithm Implementation

To facilitate AI-based parameter estimation, the dataset was split into multiple subsets, each targeting a specific parameter. Each dataset includes real system observations and corresponding time vectors, which serve as input features for AI-based algorithms.

The AI-based models in this study were implemented using the Scikit-learn and XGBoost libraries. The dataset was split into 80% training and 20% testing, ensuring a balanced distribution of system parameters. Data preprocessing involved standardization using the StandardScaler function for neural networks and support vector regression (SVR),

while tree-based methods such as random forest and XGBoost were applied directly to raw data.

Algorithm Deployment

Each model was configured with specific hyperparameters to optimize its performance:

- Linear regression: A simple baseline model using the least-squares approach to fit the data. No regularization was applied.
- Random forest: Configured with 100 trees, using a mean squared error (MSE) criterion for node splitting and a minimum of two samples per leaf. The random_state parameter was fixed at 42 for reproducibility.
- Support vector regression: Used a radial basis function (RBF) kernel with C = 100 and ε = 0.1, optimized via grid search. Feature scaling was applied to ensure numerical stability.
- Extreme gradient boosting (XGBoost): Configured with a maximum tree depth of 6, a learning rate of 0.05, and 1000 boosting rounds, employing early stopping based on validation loss.
- Neural networks: Implemented using the MLPRegressor class with three hidden layers (128, 64, and 32 neurons) using ReLU activations. The model was trained with the Adam optimizer (learning rate = 0.001), batch size = 32, and early stopping based on validation performance. The maximum number of iterations was set to 100.
- Deep learning: A more complex neural network with three hidden layers (256, 128, 64 neurons) and L2 regularization (alpha = 0.0005). Training was performed using batch size = 64 and early stopping with a 10% validation split.
- Gaussian Process Regression (GPR): Used a default kernel with automatic hyperparameter tuning. While GPR allows for probabilistic predictions, it exhibited high computational cost and suboptimal performance in our experiments.

All models were evaluated using the error metrics presented in Section 3.3 to quantify their accuracy. Additionally, inference speed was measured using execution time for single-instance predictions and all-instance predictions, ensuring suitability for real-time edge computing deployment.

Each model was trained and validated on its respective dataset to determine the most accurate approach for system parameter estimation.

3.6. Edge Device Implementation

In this study, AI models were evaluated using three edge computing devices: the NVIDIA Jetson Xavier NX [42], the NVIDIA Jetson Nano [43], and the NVIDIA Jetson Orin Nano [44]. These devices are designed for executing AI workloads in resource-constrained environments, offering a balance between computational performance and energy efficiency.

3.6.1. NVIDIA Jetson Xavier NX

The Jetson Xavier NX is a high-performance platform optimized for advanced AI tasks in embedded systems. Its key specifications include the following:

- CPU: 6-core ARM v8.2 64-bit;
- GPU: 384-core NVIDIA Volta with 48 Tensor Cores;
- RAM: 8 GB LPDDR4x;
- STORAGE: 16 GB eMMC 5.1;
- POWER CONSUMPTION: Configurable between 10 W and 15 W.

3.6.2. NVIDIA Jetson Orin Nano

The Jetson Orin Nano is a next-generation edge AI platform that offers an excellent balance between performance and energy efficiency, making it suitable for more demanding edge applications than the Jetson Nano, while maintaining a compact form factor. Its key specifications are the following:

- CPU: 6-core ARM Cortex-A78AE;
- GPU: 512-core NVIDIA Ampere;
- RAM: 4 GB LPDDR5;
- STORAGE: microSD card;
- POWER CONSUMPTION: Configurable between 7 W and 10 W.

3.6.3. NVIDIA Jetson Nano

The Jetson Nano is a more compact and energy-efficient alternative, suitable for edge AI applications with moderate computational requirements. Its main specifications are as follows:

- CPU: 4-core ARM Cortex-A57;
- GPU: 128-core NVIDIA Maxwell;
- RAM: 4 GB LPDDR4;
- STORAGE: 16 GB eMMC 5.1;
- POWER CONSUMPTION: Configurable between 5 W and 10 W.

Although its processing power is lower than that of the Jetson Xavier NX, the Jetson Orin Nano remains a viable option for lightweight AI models and applications where power efficiency is a priority.

In this study, the selected metrics (Section 3.3) are critical to determine the feasibility of deploying AI solutions in edge environments, ensuring an optimal balance between processing speed, power consumption, and predictive performance.

4. Results

This section presents a comprehensive evaluation of the proposed hybrid identification methodology from two different perspectives. First, we discuss the training and validation phase, where AI-based algorithms are trained using inputs generated from simulations and PSO-derived output. Second, we describe the inference phase for both the PSO and AI-based algorithms, in which the trained models are applied to real experimental data. This dual analysis provides insights into the accuracy of parameter estimation during model training and the practical effectiveness of the approach when deployed in real-world scenarios.

4.1. Training and Validation Phase Results

The experimental evaluation was divided into four independent tasks, each dedicated to predicting one of the key system parameters: K, T, L, and α. Initial experiments attempted to predict all parameters simultaneously; however, this approach yielded suboptimal results, leading to the decision to decouple the prediction tasks. For each target parameter, several machine learning models were applied and evaluated using standard regression metrics, including MSE, RMSE, MAE, and R^2 score. The results of each prediction task are summarized below.

4.1.1. Prediction of K

The results for predicting variable K are presented below in Table 1.

Table 1. Performance metrics of different AI-based algorithms for estimating parameter *K*.

Algorithm	MSE	RMSE	MAE	R^2	RMAE	RRMSE
Linear Regression	0.329	0.573	0.272	0.948	3.04	6.42
Random Forest	0.617	0.785	0.357	0.903	4.00	8.79
Neural Networks	0.552	0.743	0.466	0.913	5.21	8.31
Deep Learning	3.900	1.975	1.584	0.390	17.72	22.08
XGBoost	0.687	0.828	0.360	0.892	4.03	9.27
SVR	0.342	0.584	0.217	0.946	2.43	6.54
GPR	30.655	5.536	4.763	−3.793	53.26	61.91

The first experiment aimed to estimate the *K* parameter, which had a mean of 5.632 and a standard deviation of 2.537, with values ranging across 8.943 units. The best-performing models in terms of error minimization were linear regression ($MSE = 0.329$, $R^2 = 0.948$) and SVR ($MSE = 0.342$, $R^2 = 0.946$), both achieving high accuracy relative to the variable's natural variation. In contrast, deep learning showed significantly higher errors ($MSE = 3.900$, $RMSE = 1.975$), suggesting overfitting and poor generalization. The worst performance was observed with GPR, which yielded an MSE of 30.655 and a negative R^2 score (-3.793), failing to approximate the system dynamics effectively.

4.1.2. Prediction of T

The results for predicting variable *T* are presented below in Table 2.

Table 2. Performance metrics of different AI-based algorithms for estimating parameter *T*.

Algorithm	MSE	RMSE	MAE	R^2	RMAE	RRMSE
Linear Regression	1.767	1.329	0.916	0.311	11.66	16.92
Random Forest	2.084	1.443	0.925	0.187	11.78	18.37
Neural Networks	1.657	1.287	0.940	0.354	11.97	16.38
Deep Learning	3.736	1.933	1.664	−0.456	21.19	24.60
XGBoost	2.276	1.508	0.932	0.112	11.87	19.20
SVR	2.733	1.653	0.803	−0.065	10.22	21.04
GPR	39.169	6.258	4.233	−14.264	53.88	79.65

For the *T* parameter, which had a mean of 9.201, a standard deviation of 1.603, and a range of 7.857, all models exhibited greater difficulty in achieving precise predictions compared to *K* estimation. The highest R^2 score was 0.354 achieved by ANNs, while linear regression ($MSE = 1.767$, $R^2 = 0.311$) provided competitive performance despite the parameter's complexity. GPR performed the worst, with an MSE of 39.169 and a highly negative R^2 (-14.264), confirming its inadequacy in capturing the underlying dynamics of *T*. These results indicate greater variability in estimation accuracy, consistent with the relatively lower dispersion of the *T* variable.

4.1.3. Prediction of L

The results for predicting variable *L* are presented below in Table 3.

The estimation of the *L* parameter, which had a mean of 6.215, a higher standard deviation of 3.226, and a range of 10.0, proved more challenging than *K* and *T*, as indicated by higher error values across all models. Linear regression achieved the best performance ($MSE = 7.336$, $R^2 = 0.369$), though the overall predictability of *L* remained limited. Deep learning and XGBoost failed to generalize effectively, displaying higher error metrics, while GPR once again yielded the worst results ($MSE = 26.273$, $RMSE = 5.125$, $R^2 = -1.259$), reaffirming its unsuitability for this estimation task.

Table 3. Performance metrics of different AI-based algorithms for estimating parameter L.

Algorithm	MSE	RMSE	MAE	R^2	RMAE	RRMSE
Linear Regression	7.336	2.708	2.183	0.369	21.83	27.09
Random Forest	11.502	3.391	2.780	0.010	27.81	33.92
Neural Networks	9.714	3.116	2.595	0.164	25.96	31.17
Deep Learning	15.296	3.911	3.268	−0.315	32.69	39.11
XGBoost	13.269	3.642	2.964	−0.141	29.64	36.43
SVR	11.871	3.445	3.035	−0.020	30.35	34.46
GPR	26.273	5.125	4.171	−1.259	41.72	51.26

4.1.4. Prediction of α

The results for predicting variable α are presented below in Table 4.

Table 4. Performance metrics of different AI-based algorithms for estimating parameter α.

Algorithm	MSE	RMSE	MAE	R^2	RMAE	RRMSE
Linear Regression	0.014	0.118	0.081	0.641	8.63	12.63
Random Forest	0.013	0.116	0.086	0.667	18.264	15.40
Neural Networks	0.011	0.107	0.075	0.706	7.96	11.43
Deep Learning	0.032	0.180	0.146	0.206	25.899	17.97
XGBoost	0.015	0.124	0.087	0.622	18.539	15.79
SVR	0.013	0.114	0.082	0.678	18.537	16.62
GPR	0.250	0.500	0.413	−5.106	61.625	28.97

The α parameter presented the narrowest range (0.942) and smallest standard deviation (0.198) among all targets, indicating a more stable and constrained distribution. The models performed consistently well, with neural networks achieving the highest R^2 (0.706) and the lowest MSE (0.011). SVR ($R^2 = 0.678$, $MSE = 0.013$) and linear regression ($R^2 = 0.641$, $MSE = 0.014$) followed closely, showing comparable performance. Deep learning models exhibited slightly higher error values ($MSE = 0.032$), while GPR once again underperformed ($MSE = 0.250$, $R^2 = -5.106$). The lower variability of α appears to have contributed to model stability, enabling most approaches to achieve better estimation accuracy compared to the other targets.

4.2. Experimental Setup and Validation

To evaluate the performance of the proposed identification methodologies, both the PSO-based parameter identification and the AI-driven parameter estimation approaches were implemented and tested on a custom-built heating system. This experimental platform, originally developed for research and educational purposes [45], offers a versatile environment that emulates the dynamics of thermal processes exhibiting fractional behavior.

Figure 2 displays the experimental apparatus used in this section, along with the block diagrams representing the controlled process for both configurations. The thermal process occurs at the top of the prototype, featuring a thermal module where heat transfer is primarily governed by a heating element and controlled airflow. The experimental prototype supports two configurations: in the first, the heating resistor acts as the actuator while the fan remains at a fixed setting; in the second, the heating resistor operates at a constant power level, and the air fan serves as the actuator. More details about this experimental setup can be found in [45].

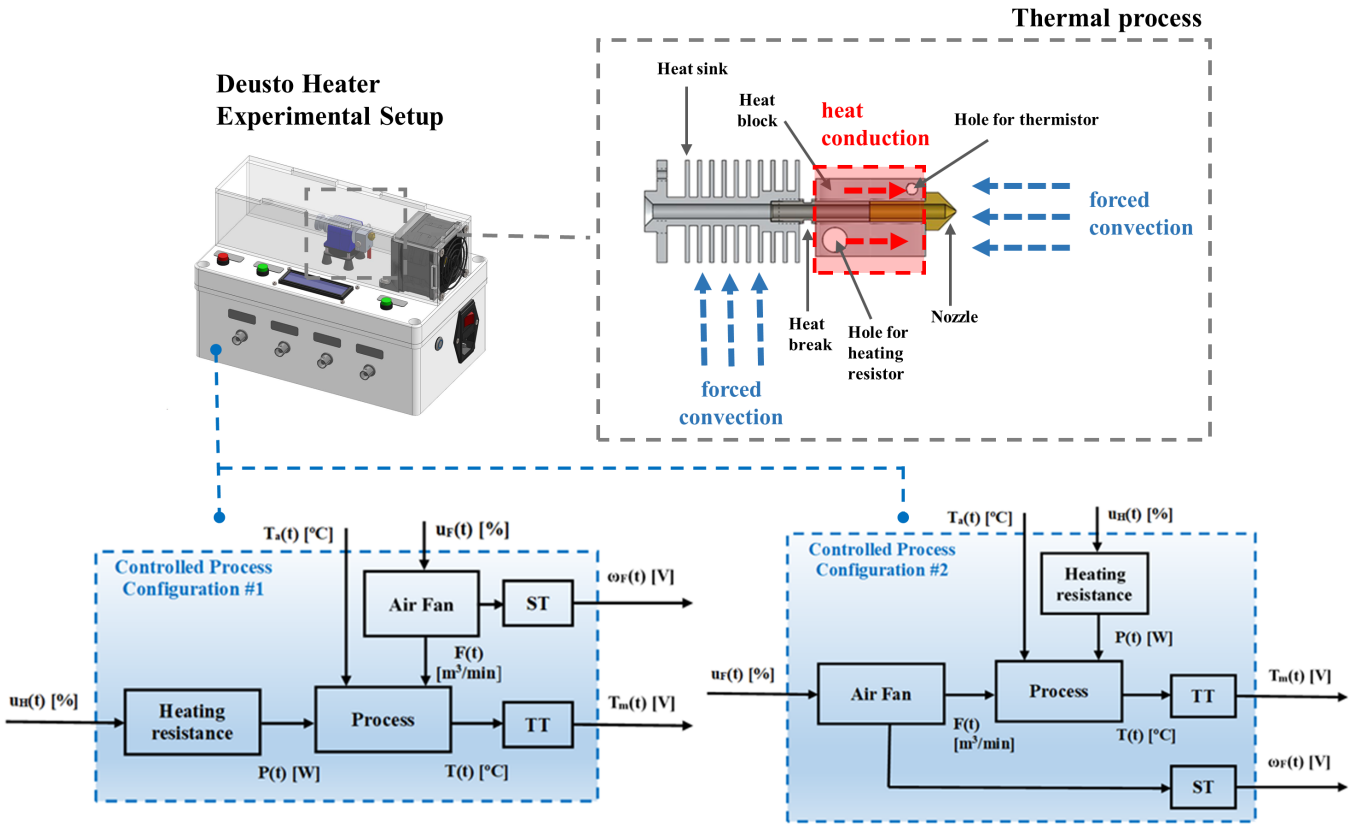

Figure 2. Image of the experimental setup, including details of the thermal process occurring in the extruder head. The figure also illustrates the block diagram of the controlled process for both available configurations.

Computation Time

To evaluate the computational efficiency of the proposed AI-based parameter estimation methods, the execution times required to predict the system parameters (K, T, L, and α) were measured on three edge devices: NVIDIA Xavier NX, NVIDIA Jetson Nano, and NVIDIA Jetson Orin Nano. Table 5 presents a detailed comparative analysis of the execution times for each algorithm across all four parameters and hardware platforms.

It is important to note that the AI-based algorithm metrics are computed as simple deterministic numbers. That is, given the same input, these algorithms produce consistent output with negligible variance in execution time; any observed latency primarily stems from low-level system operations. This determinism ensures that the reported execution times for AI methods are highly reproducible across runs.

Table 5. Execution times (in seconds) of different AI-based algorithms on three edge devices for estimating system parameters (K, T, L, and α).

Algorithm	NVIDIA Xavier NX				NVIDIA Jetson Nano				NVIDIA Jetson Orin Nano			
	K (s)	T (s)	L (s)	α (s)	K (s)	T (s)	L (s)	α (s)	K (s)	T (s)	L (s)	α (s)
Linear Regression	0.024	0.052	0.026	0.020	0.038	0.039	0.037	0.082	0.027	0.013	0.026	0.036
Random Forest	0.039	0.043	0.041	0.038	0.062	0.060	0.038	0.060	0.029	0.028	0.038	0.028
Neural Networks	0.001	0.001	0.001	0.007	0.002	0.009	0.002	0.002	0.001	0.0007	0.002	0.003
Deep Learning	0.002	0.002	0.002	0.001	0.002	0.003	0.002	0.034	0.001	0.0009	0.002	0.0008
XGBoost	0.037	0.039	0.040	0.041	0.019	0.026	0.024	0.025	0.012	0.014	0.017	0.019
SVR	0.026	0.030	0.043	0.001	0.004	0.008	0.022	0.0016	0.004	0.005	0.016	0.001
GPR	0.035	0.051	0.080	0.002	0.030	0.084	0.152	0.006	0.012	0.031	0.052	0.006

For the PSO algorithm, a similar execution time analysis was performed on the same set of edge devices. However, unlike AI-based algorithms, which yield deterministic execution times due to their fixed computational steps, the PSO algorithm exhibits variable execution times. This variability arises from its inherently iterative nature and the use of adaptive stopping criteria. The convergence of the PSO process depends on the specific characteristics of the optimization landscape for each sample, which can lead to differences in the number of iterations required to satisfy the convergence threshold. As a result, Figure 3 illustrates the distribution of PSO execution times.

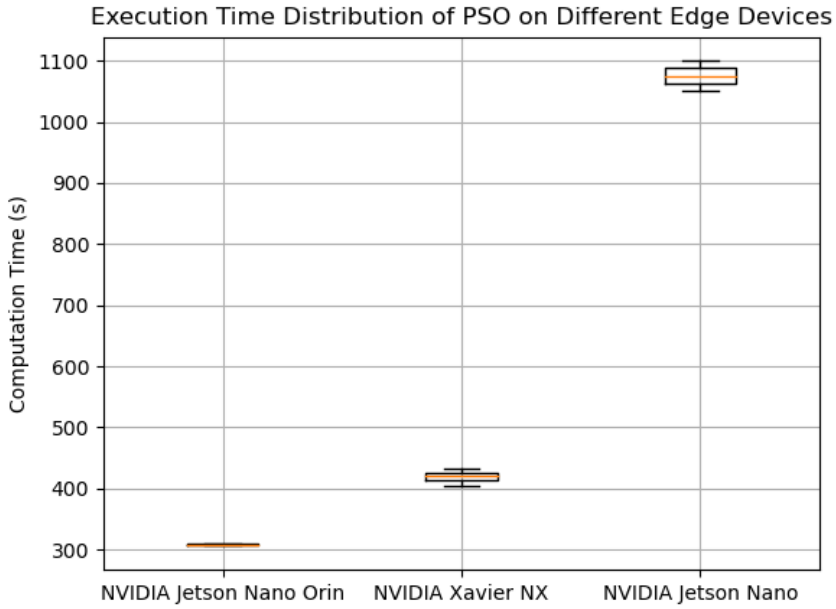

Figure 3. Boxplot of PSO execution times on three edge devices (NVIDIA Xavier NX, NVIDIA Jetson Nano, and NVIDIA Jetson Orin Nano), illustrating the variability due to the iterative optimization process and adaptive stopping criteria.

In Table 6, we present the total execution times for the various AI-based algorithms and the PSO algorithm across the three evaluated edge devices. It is important to emphasize that the total execution time for the AI-based algorithms is not simply the sum of the individual metric execution times, as these algorithms are capable of executing the parameter predictions in parallel, thus reducing the overall latency. In contrast, the execution time for the PSO algorithm is reported as the median value from multiple runs; however, this median does not fully capture the distribution of execution times, as shown by the variability in Figure 3.

Table 6. Total execution time (in seconds) of different AI-based algorithms and PSO on edge devices for estimating system parameters.

Algorithm	NVIDIA Xavier NX	NVIDIA Jetson Nano	NVIDIA Jetson Orin Nano
Linear Regression	0.057	0.092	0.039
Random Forest	0.049	0.071	0.043
Neural Networks	0.007	0.009	0.003
Deep Learning	0.002	0.038	0.002
XGBoost	0.048	0.029	0.021
SVR	0.051	0.026	0.018
GPR	0.096	0.186	0.062
PSO (median)	420.470	1076.751	308.133

4.3. Algorithm Performance Evaluation

To assess the effectiveness of the proposed identification algorithms under different operating conditions, two separate experiments were conducted using our custom-built heating system. Both experiments employed an open-loop step test procedure but corresponded to distinct configurations of the experimental apparatus, as illustrated in Figure 2. Notably, the dynamic response of the system varies significantly due to the different actuators modulated in each experiment.

When using configuration #1 (Experiment 1), the air fan is maintained at a constant power level, and the heating resistor is used as the actuator. For configuration #2 (Experiment 2), the air fan acts as the primary actuator, while the heating resistor remains at a fixed setting. For each experiment, the process reaction curve was recorded over a dynamically determined time horizon, extending until steady state was reached.

In both cases, the recorded input was fed to both the PSO and AI-based algorithms. The recorded real system output serves as the reference signal against which the estimated outputs are compared. Although all AI-based algorithms were evaluated during our investigation, neural networks consistently demonstrated superior performance in terms of both error minimization and model consistency. Therefore, for the final analysis, only the neural network results are presented as the representative AI-based estimation.

4.3.1. Experiment 1 (Configuration #1)

In Experiment 1, the experimental setup was used in its configuration #1, so that the air fan remained at a constant setting and the heating resistor acted as the actuator. In this configuration, a step-input signal was applied directly to the heating element.

Specifically, an initial step input of magnitude $u_H = 30\%$ was applied to the heating resistor, while the command signal to the air fan was kept constant at $u_F = 10\%$. This modification generated a process reaction curve with a temperature variation from 60.5 °C to 102.5 °C, corresponding to a ΔT_m of 42 °C.

Table 7 provides the estimated reduced-order parameters for Experiment 1, and Table 8 reports the corresponding error metrics.

Table 7. Estimated reduced-order parameters for Experiment 1 (Configuration #1), highlighting the comparative performance of PSO and neural network methods, demonstrating the accuracy and robustness of the proposed identification framework.

Parameter	PSO Estimate	Neural Network Estimate
K (°C/%)	1.37317	1.37239
T (s)	63.42542	63.97536
L (s)	16.81552	16.84059
α	0.99286	0.99096

Table 8. Error metrics for Experiment 1, obtained by comparing the real system output with the estimated output from PSO and the neural network. (Placeholder values).

Metric	PSO	Neural Network
MSE ($°C^2$)	0.15983	0.18715
	0.15983	0.18715
RMSE (°C)	0.39979	0.43261
MAE (°C)	0.23269	0.27063
R^2	0.99884	0.99865
RMAE	0.24952	0.24952
RRMSE	0.42870	0.46389

Figure 4 presents a representative comparison of the real system output and the estimated output from both PSO and the neural network for Experiment 1.

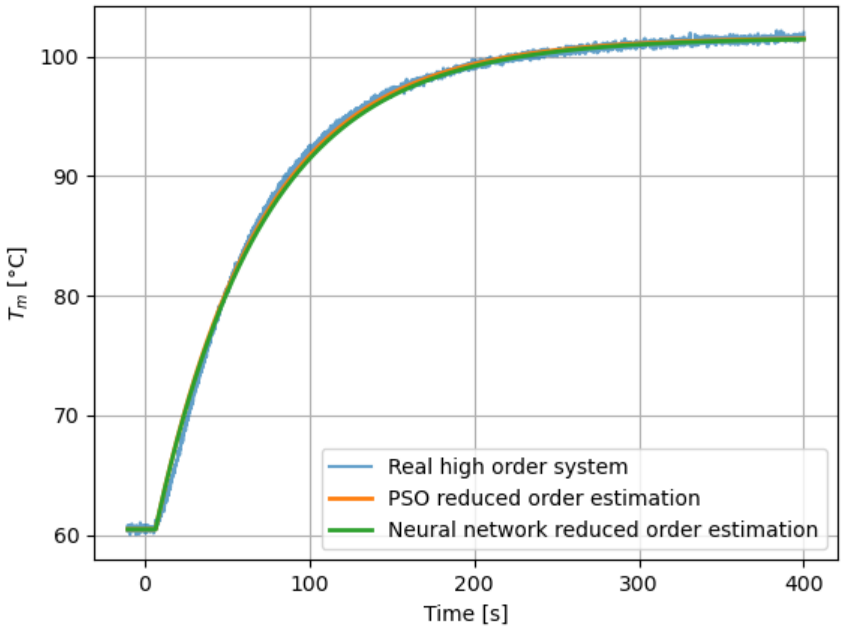

Figure 4. Comparison of the real system output (blue) with the estimated output obtained via PSO (orange) and the neural network (green) for Experiment 1 (Configuration #1). The figure illustrates the accuracy of both methods in capturing the transient and steady-state behaviors of the system, with a focus on the temperature variations in response to a step input.

4.3.2. Experiment 2 (Configuration #2)

For Experiment 2, the experimental setup was used in its configuration #2. The system was stimulated by a sudden reduction in fan speed, while the heating resistor maintained a steady power output. This configuration produced an overdamped response that was recorded over a dynamically determined time horizon until steady state was achieved.

In particular, the control signal to the air fan was set to $u_F = 40\%$ while the heating resistor remained at a constant command signal of $u_H = 100\%$. At $t = 0$ s, a step change with an amplitude of $\Delta u_F = -20\%$ was applied to the fan, resulting in a process reaction curve characterized by a temperature increase from 113.75 °C to 139.75 °C (i.e., $\Delta T_m = 26$ °C).

Table 9 summarizes the estimated reduced-order model parameters (K, T, L, and α) for Experiment 2, while Table 10 presents the corresponding error metrics computed by comparing the estimated output with the real system output.

Table 9. Estimated reduced-order parameters for Experiment 2 (Configuration #2). These parameters are derived from both PSO and neural network-based methods, demonstrating the accuracy and robustness of the proposed identification framework.

Parameter	PSO Estimate	Neural Network Estimate
K (°C/%)	−1.30235	−1.3228
T (s)	46.7780	47.8970
L (s)	12.2130	11.3650
α	0.9740	0.9643

Table 10. Error metrics for Experiment 2, obtained by comparing the real system output with the estimated output from PSO and the neural network. (Placeholder values).

Metric	PSO	Neural Network
MSE ($°C^2$)	0.04676	0.09743
RMSE (°C)	0.21623	0.31215
MAE (°C)	0.16998	0.25093
R^2	0.99896	0.99784
RMAE	0.12542	0.18515
RRMSE	0.15955	0.23032

Figure 5 displays a representative comparison of the real system output with the estimated output obtained via PSO and the neural network for Experiment 2.

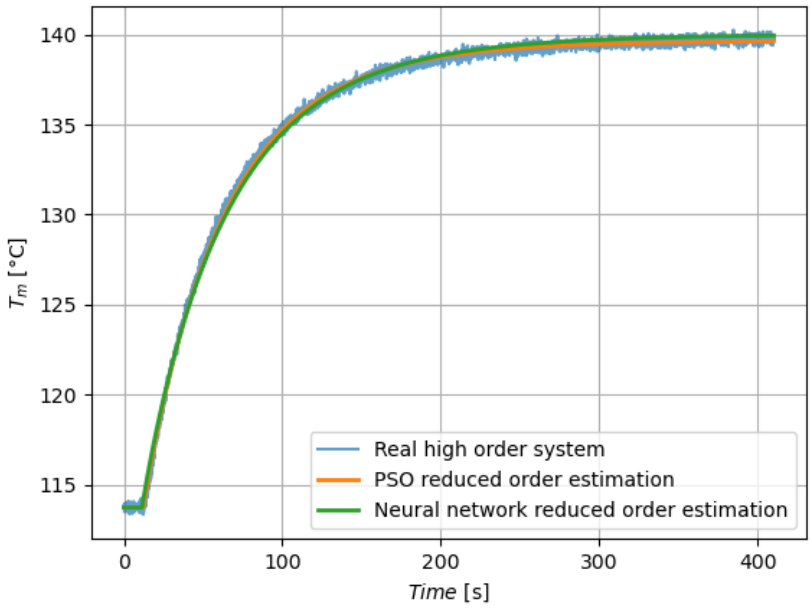

Figure 5. Comparison of the real system output (blue) with the estimated output obtained via PSO (orange) and the neural network (green) for Experiment 2 (Configuration #2). The figure illustrates the accuracy of both methods in capturing the transient and steady-state behaviors of the system, with a focus on the temperature variations in response to a step input.

5. Discussion

5.1. Comparative Analysis of PSO and AI-Based Parameter Estimation Methods

Our experimental evaluation comprised two real-world experiments conducted on our custom-built heating system. Both experiments aimed to assess the performance of the hybrid identification framework, where PSO-based optimization is used to generate high-fidelity parameter estimates that subsequently train neural network models under actual operating conditions. The parameters of the reduced-order model and the corresponding performance metrics are summarized in Tables 7 and 8 for Experiment 1, and in Tables 9 and 10 for Experiment 2. In addition, Figures 4 and 5 visually compare the real system output with the estimates provided by both the PSO and neural network approaches.

In both experiments, the PSO-based method consistently achieved very high accuracy by directly minimizing the mean squared error between the high-order system output and the reduced model output. Although the absolute error metrics varied slightly between experiments—reflecting minor differences in environmental conditions and system noise—the PSO approach maintained an excellent fit overall. However, its inherent iterative

optimization process and the adaptive stopping criteria, which are set loosely to ensure broad applicability, led to variable and generally higher execution times.

In contrast, the neural network component of our hybrid approach, trained on the PSO-generated data, exhibited slightly higher error metrics compared to PSO but offered significant advantages in terms of speed and determinism. The neural network's fixed computational steps ensured consistent, low-latency inference, making it particularly suitable for real-time applications on edge devices. Across both experiments, the performance trends of the neural network remained robust, with error metrics that were closely aligned despite the slightly elevated values relative to PSO.

Moreover, the reduced-order parameters—namely, K, T, L, and α—estimated by both methods were highly comparable between the two experiments (see Tables 7 and 9). This consistency confirms the robustness and generalizability of the hybrid identification framework, as the neural network was able to replicate the high-fidelity estimates generated by PSO across varying real-world conditions.

5.2. Model Accuracy and Trade-Offs

Our analysis of the reduced-order models derived via PSO and AI-based identification methods indicates that they effectively capture the key dynamic behaviors of the original high-order system. However, several trade-offs and potential sources of error emerge when simplifying complex dynamics into a lower order representation.

5.2.1. Accuracy Assessment

The error metrics presented in Tables 8 and 10 and the visual comparison in Figures 4 and 5 demonstrate that both the PSO and neural network approaches yield models that closely approximate the real system output. High R^2 values, along with low MSE, RMSE, and MAE, indicate that the reduced-order models accurately reproduce both the transient and steady-state dynamics. This confirms that the simplified models are capable of representing the essential characteristics required for control and prediction tasks.

5.2.2. Potential Sources of Error

Despite the promising accuracy, several factors may contribute to discrepancies between the reduced-order models and the original high-order system:

- Model simplification: Reducing a high-order system to a lower order model necessitates certain approximations, which may omit higher frequency dynamics or nonlinear effects inherent in the full-order system. With our approach, the PSO estimates, which capture the intricate behavior of the full-order system, serve as a robust training dataset for the neural network. As a result, even though the reduced-order model inherently simplifies the system dynamics, the neural network is trained to generalize from the detailed PSO output, thereby compensating for potential oversimplifications.
- Parameter estimation: Both PSO and AI-based algorithms introduce estimation errors. For instance, the PSO algorithm employs a relatively loose stopping criterion to ensure generality across various scenarios, which can result in suboptimal parameter convergence. Similarly, while neural networks offer rapid inference, their performance is dependent on the quality and representativeness of the simulated training data. Our experimental results provide clear evidence that the hybrid approach effectively addresses challenges in parameter estimation. These consistently low errors across experiments affirm that the hybrid approach successfully mitigates parameter estimation challenges.
- Noise and uncertainty: Measurement noise, sensor inaccuracies, and environmental variations in the experimental setup can lead to errors in the recorded output. These

inaccuracies directly affect the fidelity of the parameter estimation process. In our experiments, the raw acquired signal—complete with inherent measurement noise and environmental uncertainty—was directly fed into both the PSO and the hybrid approaches. Remarkably, despite the presence of such noise in the raw data, both methods yielded low error metrics. The PSO-generated estimates maintained excellent accuracy, while the neural network trained on these high-fidelity outputs. These results demonstrate that our hybrid approach is robust against noise and uncertainty, effectively extracting the true system dynamics from real-world measurements.

- Computational trade-offs: There is a clear trade-off between computational efficiency and model accuracy. The PSO approach, while delivering high-fidelity estimates, incurs longer and more variable execution times, which may limit its applicability in real-time scenarios. In contrast, the neural network-based component of our hybrid approach offers a dramatic reduction in computational time, enabling rapid, deterministic inference. Importantly, the slight increase in error metrics observed with the neural network is minimal and remains within acceptable bounds. Thus, the significant decrease in compute time more than compensates for the marginal increase in error, making the hybrid approach highly suitable for real-time applications where determinism and speed are critical.

5.3. Novel Contributions

Our approach introduces several novel contributions that significantly advance the field of fractional-order system identification. First, we propose a hybrid framework that integrates PSO-based optimization with AI-driven inference. The novelty of our approach does not stem from the use of PSO alone but rather from its integration within a hybrid system identification framework. This integration enables the accurate estimation of reduced-order model parameters from high-order systems by leveraging the high precision of PSO and the deterministic, rapid execution of AI methods. As a result, our methodology achieves a favorable balance between estimation accuracy and real-time performance, which is crucial for edge computing applications.

Second, our comprehensive evaluation encompasses both simulated data and experiments on a real heating system. By validating our methods in a practical setting, we demonstrate that the proposed framework is robust and applicable in real-world scenarios. The detailed error metric analysis and execution time measurements further confirm that our approach can reliably capture the dynamics of complex fractional-order systems while remaining computationally efficient.

Third, we provide a systematic comparison between PSO and AI-based techniques, highlighting the trade-offs associated with each. Our results indicate that while PSO achieves marginally better accuracy, its variable and higher execution times may limit its use in time-sensitive applications. In contrast, the AI-based method, particularly using neural networks, offers consistent and low-latency performance, making it a more attractive option for deployment in resource-constrained environments.

Finally, our work advances the current state-of-the-art by introducing a dynamic sampling strategy for parameter estimation and by rigorously analyzing both the accuracy and computational costs involved in the identification process. Collectively, these contributions pave the way for more efficient and reliable fractional-order system identification, particularly in applications that require real-time operation on edge devices.

5.4. Implications for Edge Device Deployment

Deploying the proposed hybrid identification models on edge devices presents both significant benefits and notable challenges. One of the primary advantages is the capability for

low-latency, real-time processing. The deterministic execution times of the AI-based approach, particularly the neural network implementation, enable rapid inference that is crucial for real-time control and monitoring tasks. This has been evidenced by the performance metrics on platforms such as the NVIDIA Jetson Xavier NX, Jetson Nano, and Jetson Orin Nano (see Tables 5 and 6). The consistent, low-latency performance makes these models well-suited for edge computing scenarios where prompt decision-making is critical.

In summary, the combination of high-speed AI-based inference with the localized processing capabilities of edge devices presents a powerful solution for real-time dynamic system identification. Future improvements in algorithm efficiency and hardware compatibility will further enhance these speed-ups, making the deployment of sophisticated identification models on resource-constrained edge devices increasingly feasible and effective.

6. Conclusions

6.1. Summary of Findings

Our study demonstrates the successful derivation of a reduced-order fractional model that accurately approximates the dynamics of a fractional-order system using a hybrid identification framework, with a strong emphasis on its applicability for edge computing. The key findings of this work are summarized as follows:

- The PSO-based approach achieved high accuracy in estimating the reduced-order model parameters, as evidenced by low MSE values and high R^2 scores. However, its iterative optimization process results in variable execution times, which may limit its suitability for time-critical edge applications.
- The AI-based method, particularly the neural network implementation, provided deterministic and rapid inference times, making it ideally suited for deployment on edge devices. Although its error metrics were marginally higher compared to the PSO approach, the neural network's speed and consistency offer a significant advantage in real-time scenarios.
- Experimental validation on a custom-built heating system confirmed that both approaches can effectively capture the essential dynamics of the high-order system. The reduced-order parameters estimated by both methods closely matched the expected values, demonstrating the robustness of the proposed framework.
- A comprehensive analysis of error metrics and computational performance revealed a clear trade-off between accuracy and speed. While the PSO method offers slightly higher accuracy, its increased and variable computational demands make it less favorable for edge deployment. In contrast, the neural network approach strikes a favorable balance, delivering competitive accuracy with the deterministic low-latency performance required for edge computing.
- The integration of our identification framework with edge devices underscores its potential for real-time dynamic system identification, enabling rapid decision-making and control in resource-constrained environments.

Overall, the proposed hybrid identification framework effectively balances the trade-off between model accuracy and computational efficiency. By leveraging the strengths of PSO for high-fidelity parameter estimation and AI-based methods for real-time execution, our approach provides a robust solution for fractional-order system identification. The experimental validation across different configurations confirms the framework's adaptability to varying system dynamics, reinforcing its viability for real-time deployment on edge devices. These findings emphasize the potential of hybrid optimization–AI methodologies in engineering applications, particularly where computational constraints and execution speed are critical factors.

6.2. Practical and Theoretical Implications

Our work has several broader implications for both system identification and control system design. From a practical standpoint, the hybrid identification framework—integrating PSO-based optimization with AI-driven inference—demonstrates that high-fidelity reduced-order models can be derived and deployed on edge devices for real-time applications. The deterministic, low-latency performance of the AI approach, combined with the high accuracy of PSO, enables rapid, adaptive control in resource-constrained environments.

Theoretically, our study advances the understanding of trade-offs between computational complexity and estimation accuracy in fractional-order systems. The comparative analysis between PSO and AI methods highlights the benefits and limitations inherent in each approach. Furthermore, the dynamic sampling strategy and comprehensive dataset generation employed in our study provide a robust foundation for extending these methods to more complex or nonlinear systems.

In summary, our findings establish a clear pathway for the adoption of hybrid optimization–AI approaches in dynamic system identification, particularly for fractional-order models. The demonstrated improvements in computational efficiency, coupled with real-time implementation feasibility, provide valuable insights for future developments in control system design. This research not only highlights the practical benefits of deploying hybrid identification frameworks in edge computing environments but also sets the foundation for further exploration into scalable and adaptive modeling strategies for more complex dynamical systems.

6.3. Future Research Directions

In light of the findings and limitations of the current study, several avenues for future research emerge:

- Optimization technique enhancements: Explore hybrid approaches that compute only the most critical performance metrics using AI, while deriving the remaining metrics analytically. This strategy could further optimize computational efficiency without sacrificing accuracy.
- Larger system scalability: While our current study focuses on a representative thermal system, an important direction for future work is evaluating the scalability of the proposed hybrid PSO-AI framework for larger and more complex fractional-order systems.
- Non-commensurate fractional-order systems: An important extension of this study would involve applying the hybrid PSO–AI framework to non-commensurate fractional-order systems, where different components exhibit distinct fractional orders. This would further validate the method's robustness and generalizability across a broader range of fractional-order dynamic behaviors.
- Real-time adaptation via DRL: Develop approaches based on deep reinforcement learning (DRL) that can adapt "on the fly" to changing system dynamics. This would enable continuous optimization and improved performance under varying operational conditions.
- Enhanced experimental validation: Extend validation of the proposed methods to a broader set of experimental data, encompassing a wider range of operating conditions and dynamic behaviors. This would help to generalize the findings and improve the robustness of the identification framework.
- Alternative AI Models: Evaluate the performance of alternative AI architectures (e.g., convolutional neural networks, recurrent neural networks, or transformer-based models) for parameter estimation.

These future research directions aim not only to improve the computational efficiency and accuracy of the identification process but also to extend the applicability of the framework to a wider range of dynamic systems and edge computing scenarios.

Author Contributions: Conceptualization, I.F.A., N.G.-L. and J.J.G.; Methodology, I.F.A., N.G.-L. and J.J.G.; Software, I.F.A., N.G.-L. and J.J.G.; Validation, I.F.A., N.G.-L. and J.J.G.; Formal analysis, I.F.A., N.G.-L. and J.J.G.; Investigation, I.F.A., N.G.-L. and J.J.G.; Resources, I.F.A., N.G.-L. and J.J.G.; Data curation, I.F.A., N.G.-L. and J.J.G.; Writing—original draft, I.F.A., N.G.-L. and J.J.G.; Writing—review & editing, I.F.A., N.G.-L. and J.J.G.; Visualization, I.F.A., N.G.-L. and J.J.G.; Supervision, I.F.A., N.G.-L. and J.J.G.; Project administration, I.F.A., N.G.-L., J.J.G. and I.P.; Funding acquisition, I.F.A. and I.P. All authors have read and agreed to the published version of the manuscript.

Funding: This work was conducted within the GRECO project, "Transformation of AI systems engineering to improve efficiency and environmental impact through GREen COmputing", funded by the SPRI (Basque Business Development Agency) under the ELKARTEK program, grant number KK-2024/0090.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
ANN	Artificial Neural Network(s)
CRONE	Commande Robuste d'Ordre Non Entier
DRL	Deep Reinforcement Learning
FFOPDT	Fractional First-Order Plus Dead-Time
FoS	Fractional-Order System(s)
GPR	Gaussian Process Regression
MDPI	Multidisciplinary Digital Publishing Institute
MSE	Mean Squared Error
PSO	Particle Swarm Optimization
RMSE	Root Mean Squared Error
RMAE	Relative Mean Absolute Error
RRMSE	Relative Root Mean Squared Error
SVR	Support Vector Regression
XGBoost	Extreme Gradient Boosting

References

1. Smith, C.A.; Corripio, A.B. *Principles and Practices of Automatic Process Control*; John Wiley & Sons: Hoboken, NJ, USA, 2005.
2. Wang, L. *From Plant Data to Process Control: Ideas for Process Identification and PID Design*; CRC Press: Boca Raton, FL, USA, 2000.
3. Åström, K.J.; Hägglund, T. *Advanced PID Control*; ISA-The Instrumentation, Systems and Automation Society: Durham, NC, USA, 2006.
4. Garpinger, O.; Hägglund, T.; Åström, K.J. Performance and robustness trade-offs in PID control. *J. Process Control* **2014**, *24*, 568–577. [CrossRef]
5. Mehta, U.; Bingi, K.; Saxena, S. *Applied Fractional Calculus in Identification and Control*; Springer: Berlin/Heidelberg, Germany, 2022.
6. Sun, H.; Zhang, Y.; Baleanu, D.; Chen, W.; Chen, Y. A new collection of real world applications of fractional calculus in science and engineering. *Commun. Nonlinear Sci. Numer. Simul.* **2018**, *64*, 213–231. [CrossRef]
7. Efe, M.Ö. Fractional order systems in industrial automation—A survey. *IEEE Trans. Ind. Inform.* **2011**, *7*, 582–591. [CrossRef]
8. Muresan, C.I.; Ionescu, C.M. Generalization of the FOPDT model for identification and control purposes. *Processes* **2020**, *8*, 682. [CrossRef]

9. Tavakoli-Kakhki, M.; Haeri, M.; Tavazoei, M.S. Simple fractional order model structures and their applications in control system design. *Eur. J. Control* **2010**, *16*, 680–694. [CrossRef]
10. Campos, M.W.; Ayres, F.A., Jr.; de Bessa, I.V.; de Medeiros, R.L.; Martins, P.R.; kaminski Lenzi, E.; João Filho, E.; Vilchez, J.R.; Lucena, V.F., Jr. Fractional-order identification system based on Sundaresan's technique. *Chaos Solitons Fractals* **2024**, *185*, 115132. [CrossRef]
11. Gude, J.J.; García Bringas, P. Proposal of a General Identification Method for Fractional-Order Processes Based on the Process Reaction Curve. *Fractal Fract.* **2022**, *6*, 526. [CrossRef]
12. Gude, J.J.; García Bringas, P. Influence of the selection of reaction curve's representative points on the accuracy of the identified fractional-order model. *J. Math.* **2022**, *2022*, 7185131. [CrossRef]
13. Gude, J.J.; Di Teodoro, A.; Camacho, O.; García Bringas, P. A New Fractional Reduced-Order Model-Inspired System Identification Method for Dynamical Systems. *IEEE Access* **2023**, *11*, 103214–103231. [CrossRef]
14. Malek, H.; Luo, Y.; Chen, Y. Identification and tuning fractional order proportional integral controllers for time delayed systems with a fractional pole. *Mechatronics* **2013**, *23*, 746–754. [CrossRef]
15. Ahmed, S. Parameter and delay estimation of fractional order models from step response. *IFAC-PapersOnLine* **2015**, *48*, 942–947. [CrossRef]
16. Alagoz, B.B.; Tepljakov, A.; Ates, A.; Petlenkov, E.; Yeroglu, C. Time-domain identification of one non-integer order plus time delay models from step response measurements. *Int. J. Model. Simul. Sci. Comput.* **2019**, *10*, 1941011. [CrossRef]
17. Guevara, E.; Meneses, H.; Arrieta, O.; Vilanova, R.; Visioli, A.; Padula, F. Fractional order model identification: Computational optimization. In Proceedings of the 2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA), Luxembourg, 8–11 September 2015; pp. 1–4.
18. Meneses, H.; Arrieta, O.; Padula, F.; Visioli, A.; Vilanova, R. FOPI/FOPID Tuning Rule Based on a Fractional Order Model for the Process. *Fractal Fract.* **2022**, *6*, 478. [CrossRef]
19. Gude, J.J.; García Bringas, P.; Herrera, M.; Rincón, L.; Di Teodoro, A.; Camacho, O. Fractional-order model identification based on the process reaction curve: A unified framework for chemical processes. *Results Eng.* **2024**, *21*, 101757. [CrossRef]
20. Podlubny, I. *Fractional Differential Equations: An Introduction to Fractional Derivatives, Fractional Differential Equations, to Methods of Their Solution and Some of Their Applications*; Elsevier: Amsterdam, The Netherlands, 1998.
21. Kilbas, A.A.; Srivastava, H.M.; Trujillo, J.J. *Theory and Applications of Fractional Differential Equations*; Elsevier: Amsterdam, The Netherlands, 2006; Volume 204.
22. Gorenflo, R.; Kilbas, A.A.; Mainardi, F.; Rogosin, S.V. *Mittag-Leffler Functions, Related Topics and Applications*; Springer: Berlin/Heidelberg, Germany, 2020.
23. Rogosin, S. The role of the Mittag-Leffler function in fractional modeling. *Mathematics* **2015**, *3*, 368–381. [CrossRef]
24. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95-International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948. [CrossRef]
25. Miranda, L.J.V. PySwarms, a research-toolkit for Particle Swarm Optimization in Python. *J. Open Source Softw.* **2018**, *3*, 433. [CrossRef]
26. Raubitzek, S.; Mallinger, K.; Neubauer, T. Combining fractional derivatives and machine learning: A review. *Entropy* **2022**, *25*, 35. [CrossRef]
27. Cheng, J.; Yang, G.; Xu, W.; Feng, H.; Han, S.; Liu, M.; Zhao, F.; Zhu, Y.; Zhao, Y.; Wu, B.; et al. Improving the estimation of apple leaf photosynthetic pigment content using fractional derivatives and machine learning. *Agronomy* **2022**, *12*, 1497. [CrossRef]
28. Annadurai, A.; Sureshkumar, V.; Jaganathan, D.; Dhanasekaran, S. Enhancing medical image quality using fractional order denoising integrated with transfer learning. *Fractal Fract.* **2024**, *8*, 511. [CrossRef]
29. Montgomery, D.C.; Peck, E.A.; Vining, G.G. *Introduction to Linear Regression Analysis*; John Wiley & Sons: Hoboken, NJ, USA, 2021.
30. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
31. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
32. Goodfellow, I.; Bengio, Y.; Courville, A.; Bengio, Y. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
33. Chen, T.; Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the the 22nd ACM Sigkdd International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
34. Smola, A.J.; Schölkopf, B. A tutorial on support vector regression. *Stat. Comput.* **2004**, *14*, 199–222. [CrossRef]
35. Williams, C.K.; Rasmussen, C.E. *Gaussian Processes for Machine Learning*; MIT Press: Cambridge, MA, USA, 2006; Volume 2.
36. Tepljakov, A.; Petlenkov, E.; Belikov, J. FOMCON toolbox for modeling, design and implementation of fractional-order control systems. In *Volume 6 Applications in Control*; Petráš, I., Ed.; De Gruyter: Berlin, Germany; Boston, MA, USA, 2019; pp. 211–236. [CrossRef]
37. Malek, H.; Dadras, S.; Yin, C.; Chen, Y. Fractional Order Proportional-Resonant Controller. In Proceedings of the 2018 Annual American Control Conference (ACC), Milwaukee, WI, USA, 27–29 June 2018; pp. 3086–3091. [CrossRef]

38. Jayaram, S.; Venkatesan, N. Design and implementation of the fractional-order controllers for a real-time nonlinear process using the AGTM optimization technique. *Sci. Rep.* **2024**, *14*, 31714. [CrossRef] [PubMed]
39. Chopade, A.S.; Khubalkar, S.W.; Junghare, A.S.; Aware, M.V.; Das, S. Design and implementation of digital fractional order PID controller using optimal pole-zero approximation method for magnetic levitation system. *IEEE/CAA J. Autom. Sin.* **2018**, *5*, 977–989. [CrossRef]
40. Pan, I.; Das, S. Fractional Order AGC for Distributed Energy Resources Using Robust Optimization. *IEEE Trans. Smart Grid* **2016**, *7*, 2175–2186. [CrossRef]
41. Gude, J.J.; García Bringas, P. Improving a reaction curve-based analytical identification technique for fractional models. *Int. J. Dynam. Control* **2025**, *13*, 27. [CrossRef]
42. NVIDIA. Jetson NX Xavier Series. Available online: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-xavier-nx (accessed on 15 February 2025).
43. NVIDIA. Jetson Nano. Available online: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-nano/product-development (accessed on 15 February 2025).
44. NVIDIA. NVIDIA Jetson Orin Nano. Available online: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin (accessed on 15 February 2025).
45. Gude, J.J.; García Bringas, P. A Novel Control Hardware Architecture for Implementation of Fractional-Order Identification and Control Algorithms Applied to a Temperature Prototype. *Mathematics* **2022**, *11*, 143. [CrossRef]