https://mathco.journals.pnu.ac.ir

Payame Noor University

Control and Optimization in Applied Mathematics

Volume 11, Issue 2, Summer-Autumn 2026, Pages 1–19

DOI:10.30473/coam.2025.75192.1322

OPEN ACCESS

Control and Optimization in Applied Mathematics

Research Article

Exploration of Physics Informed Neural Network for Solving Optimal Tracking Control Problems

Fidelis Nofertinus Zai[1]✉, **Rian Kurnia** [2], **Juan Prihanda Nainggolan**[1]

[1]Department of Mathematics, Universitas Sumatera Utara, Medan, 20155, Indonesia,

[2]Department of Data Science, Institut Teknologi Sumatera, Lampung, 35365, Indonesia.

Article History

Received: July 18, 2025

Accepted: December 30, 2025

Available online: May 21, 2026

Abstract. In this study, we examine solutions to Optimal Tracking Control (OTC) problems for both Linear Quadratic (LQ) and nonlinear systems. Classical approaches to OTC rely on formulating and solving the Hamilton-Jacobi-Bellman (HJB) equation, which typically requires numerical solutions of the state, co-state, and stationary equations using the forward-backward method. Such methods often involve intricate mathematical analysis and substantial computational effort. To address these challenges, we explored the use of Physics Informed Neural Networks (PINN) as an alternative framework for solving OTC problems. The PINN approach is implemented by constructing a problem-specific loss function that directly incorporates the governing dynamics and control objectives. This method is comparatively simpler and more flexible to implement. The performance of PINNs is evaluated through quantitative error analysis and benchmarked against the classical Runge-Kutta (RK) method. A detailed comparison is presented using tabulated error metrics and time-domain plots of absolute errors. Numerical results demonstrate that PINNs achieve lower approximation errors than Runge-Kutta method for both LQ and nonlinear tracking problems, indicating their effectiveness as a viable alternative solution strategy for OTC problems.

Keywords. Linear quadratic, Optimal tracking problem, Physics informed neural network, Hamiltonian-Jacobi-Bellman equation.

MSC. 34H05; 68T07.

How to Cite

Zai F.N., Kurnia R., Nainggolan J.P. (2026). "Exploration of physics informed neural network for solving optimal tracking control problems". *Control and Optimization in Applied Mathematics*, 11(2), 1-19. https://doi.org/10.30473/coam.2025.75192.1322

✉Corresponding author. **E-mail:** fidelis@usu.ac.id

1 Introduction

Optimal Tracking Control (OTC) aims to design control inputs that force a dynamical system to follow a prescribed reference trajectory over a finite time horizon while minimizing a specified performance index [5, 13]. OTC has been extensively studied and successfully applied in a wide range of industrial and engineering systems, including optimal control schemes for coal gasification processes [21] and trajectory tracking of hypersonic vehicles [16].

Traditional approaches to solving OTC problems often rely on plant inversion and system linearization via series expansions [8, 19]. While effective in certain settings, these methods typically require intricate mathematical analysis, particularly for nonlinear systems [5, 20]. A major challenge arises from the need to formulate and solve the Hamilton–Jacobi–Bellman (HJB) equation, which is central to optimal control theory [4, 13]. To address the intractability of closed-form solutions, various numerical approximation techniques have been developed, such as the Successive Galerkin Approximation (SGA) method [3].

Several alternative strategies have been proposed to circumvent the explicit solution of the HJB equation. For instance, the backstepping method has been applied to bilinear control systems in normal form [17]. More recently, Ahmadin et al. (2024) designed asymptotic tracking controllers for minimum nonphase bilinear systems by representing the tracking states as Fourier series and determining the corresponding coefficients via Particle Swarm Optimization (PSO) [1]. In parallel, Artificial Neural Networks (ANNs) have emerged as powerful tools for solving complex problems across diverse fields, including computer vision and natural language processing. Their application to control and tracking problems has gained increasing attention due to their strong approximation capabilities.

The integration of physical knowledge into neural networks has led to the development of Physics-Informed Neural Networks (PINNs), which have proven effective in solving partial differential equations (PDEs) and related inverse problems [10]. PINNs extend traditional ANN frameworks by embedding governing equations, boundary conditions, and initial conditions directly into the loss function. This approach has demonstrated considerable success in handling complex and nonlinear PDEs across engineering, physics, and finance [2]. Notable applications include temperature dynamics modeling in Proton Exchange Membrane (PEM) electrolysis systems [22] and the solution of boundary layer problems, where PINN results have been shown to closely match analytical solutions [18]. Motivated by these developments, this paper explores the application of PINNs to OTC problems by constructing tailored loss functions that encode system dynamics and tracking objectives.

Unlike classical numerical schemes that require explicit stability conditions—such as time-step constraints or Courant–Friedrichs–Lewy (CFL) conditions—PINN stability is governed by the formulation of the loss function, the selection of collocation points, and the optimization strategy employed during training. Recent studies suggest that, with appropriate configurations, PINNs can achieve asymptotic stability and convergence even for stiff systems [9]. Furthermore, structure-preserving PINN formulations enable the retention of fundamental system properties, such as Lyapunov stability and energy conservation, ensuring physically consistent solutions [6]. Nevertheless, a comprehensive theoretical framework for PINN stability and convergence remains an open research problem, particularly in the context of control and tracking applications.

Recent advancements in PINN research have introduced operator-learning frameworks, most notably Deep Operator Networks (DeepONets), which exhibit superior performance in learning high-dimensional operator mappings. When combined with Long Short-Term Memory (LSTM) architectures, DeepONets demonstrate enhanced accuracy in modeling time-dependent systems, even with limited training data [15]. Physics-Informed DeepONets have been successfully employed in practical applications, such as constructing digital twins capable of long-term prediction without explicit knowledge of evolving initial conditions [7]. These developments significantly extend the representational power of PINNs in capturing nonlinear dynamical behaviors.

In parallel, Adaptive Dynamic Programming (ADP) has been widely used to address OTC problems by approximating the solution of the HJB equation through value and policy iteration [16, 17, 21]. The conceptual similarity between ADP and PINN—both of which rely on minimizing an objective function—suggests promising opportunities for methodological integration. Recent studies have explored adaptive PINN variants, such as Adaptive-Weight PINNs and Adaptive-Loss PINNs, which enhance training stability and predictive accuracy [11, 23]. Moreover, the integration of PINNs into Model Predictive Control (MPC) frameworks has been shown to improve computational efficiency and tracking performance in nonlinear systems [14].

Given the analytical complexity of existing OTC solution methods, this study investigates the use of PINNs as an alternative framework for solving OTC problems. By eliminating the need for explicit analytical derivations, PINNs offer a more flexible and accessible approach to control design. The performance of PINNs with various activation functions is systematically evaluated based on tracking error metrics, providing insight into their effectiveness and robustness. This exploration aims to contribute practical and computationally efficient alternatives for solving OTC problems in real-world applications.

The remainder of this paper is organized as follows. Section 2 presents the classical solution of OTC problems using Hamiltonian-based methods. Section 3 introduces the PINN-based formulation and details the construction of the associated loss functions. Numerical examples and comparative results between traditional methods and PINNs are discussed in Section 4. Finally, conclusions and directions for future research are provided in Section 5.

2 Problem Formulation

An OTC problems can be formulated as follows.
Suppose the plant dynamic given as follows [13]:

$$\dot{x} = f(x, u), \tag{1}$$

where $x(t) = \begin{bmatrix} x_1(t) & x_2(t) & \dots & x_n(t) \end{bmatrix}^T \in \mathbb{R}^n$, x_0 is an initial given value, $u(t) \in \mathbb{R}^m$ is control function, and

$$f(x, u) = \begin{bmatrix} f_1(x, u) & f_2(x, u) & \dots & f_n(x, u) \end{bmatrix}^T \in \mathbb{R}^n.$$

The main objective is to determine the optimal control $u(t) \in \mathbb{R}^m$ such that the output

$$y(t) = Cx(t),$$

remain close to the pre-determined path $r(t)$ given on time interval $[t_0, T]$ and minimize the performance index (*cost function*) [13]:

$$\begin{aligned} J(x,u) = \; & \tfrac{1}{2}\left(Cx(T) - r(T)\right)^T P\left(Cx(T) - r(T)\right) + \\ & \tfrac{1}{2}\int_{t_0}^{T}\Big[\left(Cx(t) - r(t)\right)^T Q\left(Cx(t) - r(t)\right) + u(t)^T Ru(t)\Big]dt, \end{aligned} \tag{2}$$

where $P, Q \geq 0$, and $R > 0$ are symmetrics [13], $C, P, Q \in \mathbb{R}^{n\times n}$, $R \in \mathbb{R}^{m\times m}$ and $r(t) = \begin{bmatrix} r_1(t) & r_2(t) & \cdots & r_n(t) \end{bmatrix}^T$.

From dynamic plant in Equation (1) and performance index in Equation (2), we obtain the following Hamiltonian equation [13],

$$H = \frac{1}{2}\Big[\left(Cx(t) - r(t)\right)^T Q\left(Cx(t) - r(t)\right) + u(t)^T Ru(t)\Big] + \lambda^T f(x,u). \tag{3}$$

From Equation (3), we have state system, costate system, and stationary condition as follows,
State system:

$$\dot{x} = f(x,u).$$

Costate system:

$$-\dot{\lambda} = \left(\frac{\partial f}{\partial x}\right)^T \lambda + C^T QCx - C^T Qr.$$

Stationary condition:

$$0_m = \left(\frac{\partial f}{\partial u}\right)^T \lambda + Ru.$$

with boundary condition $x(t_0)$, $\lambda(T) = C^T P\left(Cx(T) - r(T)\right)$, and

$$\frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}, \qquad \frac{\partial f}{\partial u} = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} & \cdots & \frac{\partial f_1}{\partial u_m} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} & \cdots & \frac{\partial f_2}{\partial u_m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial u_1} & \frac{\partial f_n}{\partial u_2} & \cdots & \frac{\partial f_n}{\partial u_m} \end{bmatrix}.$$

Therefore, the optimal control is given by

$$u(t) = -R^{-1}\left(\frac{\partial f}{\partial u}\right)^T \lambda(t). \tag{4}$$

For nonlinear systems $f(x,u)$, the partial derivatives $\dfrac{\partial f}{\partial x}$ and $\dfrac{\partial f}{\partial u}$ produce a nonlinear system of ODEs that are paired with each other. Therefore, analytical solutions to nonlinear ODE systems cannot be found. Thus, the solution must be done entirely with numerical methods. The numerical approach is

sensitive to discretization and time step size, so alternatives such as PINN become relevant for directly solving nonlinear optimal tracking problems.

If the dynamic plant is of linear form, then the above problem is called the Linear Quadratic (LQ) Tracking Problem. In this case, Equation (1) can be written as

$$\dot{x} = Ax + Bu, \tag{5}$$

and costate equation can be written as:

$$-\dot{\lambda} = A^T\lambda + C^TQCx - C^TQr, \tag{6}$$

with $u = -R^{-1}B^T\lambda$ and $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, and $u \in \mathbb{R}^m$ [13]. To obtain the solution of LQ problem, the sweep method is applied by introducing additional term $v(t)$ so that we have:

$$\dot{x} = (A - BK(t))x + BR^{-1}B^Tv, \tag{7}$$

with,

$$\begin{aligned} K(t) &= R^{-1}B^TS(t), \\ -\dot{S} &= A^TS + SA - SBR^{-1}B^TS + C^TQC, \\ -\dot{v} &= (A - BK)^T v + C^TQr, \\ u &= -Kx + R^{-1}B^Tv. \end{aligned}$$

The above system of equations cannot be solved analytically so that it will be solved numerically with boundary conditions $S(T) = C^TPC$ and $v(T) = C^TPr(T)$ [13].

3 Tracking Problem Using PINNs

The PINNs extend traditional ANNs by embedding the governing differential equations directly into the network's loss function [2, 12]. This integration enables the network to respect the underlying system dynamics during training. To address the tracking problem, the system dynamics given in (1) are reformulated as a set of residual equations:

$$\begin{aligned} \dot{x}_1 - f_1(x,u) &= 0, \\ \dot{x}_2 - f_2(x,u) &= 0, \\ \dot{x}_3 - f_3(x,u) &= 0, \\ &\vdots \\ \dot{x}_n - f_n(x,u) &= 0, \end{aligned} \tag{8}$$

subject to the initial conditions $x_i(t_0) = x_0^i$ for $i = 1, 2, \ldots, n$.

The loss function associated with the system dynamics is defined using the mean squared error (MSE) of the differential equation residuals,

$$\mathcal{L}_{\text{DE}} = \frac{1}{N_t}\sum_{i=1}^{n}\sum_{j=1}^{N_t}\left(\frac{dx_i(t_j)}{dt} - f_i(t_j)\right)^2, \tag{9}$$

where N_t denotes the number of discrete time collocation points. Without loss of generality, the initial time is set to $t_0 = 0$.

In addition, the loss function corresponding to the initial conditions is expressed as:

$$\mathcal{L}_{\text{IC}} = \sum_{i=1}^{n} \left(x_i(0) - x_0^i\right)^2, \tag{10}$$

where $x_i(0)$ represents the initial value of the i-th state predicted by the PINN, and x_0^i denotes the prescribed initial condition. This term enforces consistency between the neural network solution and the known initial state of the dynamical system.

The control objective is to determine a tracking control input u such that system output the $y = Cx$ follows a desired reference trajectory $r(t)$ by minimizing a prescribed cost functional J. Using the PINN framework, the loss tracking function is formulated as

$$\mathcal{L}_{\text{tracking}} = \frac{1}{N_t} \sum_{i=1}^{n} \sum_{j=1}^{N_t} \left(\sum_{k=1}^{n} c_{ik} x_k(t_j) - r_i\left(t_j\right) \right)^2, \tag{11}$$

where N_t denotes the number of temporal collocation points, and c_{ik} are the entries of the output matrix C.

In addition, the cost-related loss term is defined as

$$\mathcal{L}_{\text{cost}} = \int_{t_0}^{T} g(x, u, t) dt, \tag{12}$$

with

$$g(x, u, t) = \left(Cx(t) - r(t)\right)^T Q \left(Cx(t) - r(t)\right) + u^T R u,$$

which corresponds to the performance index given in Equation (2).

Within the PINN implementation, the integral term in (12) is evaluated numerically at each training iteration. The optimization process proceeds iteratively until the total loss converges to a sufficiently small value, indicating that an optimal or near-optimal control solution has been obtained.

Finally, based on the loss components defined in Equations (9)–(12), the total loss function is expressed as

$$\mathcal{L}_{Total} = \mathcal{L}_{DE} + \mathcal{L}_{IC} + \mathcal{L}_{Tracking} + \mathcal{L}_{Cost}. \tag{13}$$

This total loss function is minimized during the training process to solve the optimal tracking control problems using the PINN framework.

4 Comparative Results

In this section, OTC problems are solved using both classical numerical methods and the PINN approach. Specifically, the Forward–Backward Runge–Kutta method [4] is employed as a traditional benchmark, and the results are compared with those obtained using PINNs. The OTC problems under consideration include linear systems, LQ tracking problems, and nonlinear dynamical systems. Within the PINN

framework, the temporal collocation points t_j are uniformly distributed over the prescribed time horizon. These points are used to evaluate the residuals of the governing differential equations. At each collocation point, the neural network is trained to satisfy the system dynamics by minimizing the differential equation loss term $\mathcal{L}_{\mathrm{DE}}$. In addition to enforcing the dynamic constraints, the training process simultaneously minimizes the loss terms associated with the initial conditions, tracking objectives, and performance indices. Time derivatives of the state variables, such as $\frac{dx_1}{dt}$ and $\frac{dx_2}{dt}$ are computed using automatic differentiation implemented through the *TensorFlow* library in the *Python* programming environment. This approach enables the accurate evaluation of derivatives of the network outputs without relying on finite-difference approximations. Consequently, the PINN training procedure enforces the initial conditions, system dynamics, and tracking objectives consistently over the entire time domain.

All numerical experiments were conducted on a computing platform with the following specifications: AMD Ryzen 5 6600A processor with Radeon Graphics (3.3 GHz), 12 CPU cores, and 6 GB of RAM.

4.1 Linear Quadratic Tracking Problem

Consider the following dynamic plant [13]:

$$\frac{dx_1}{dt} = x_2, \ \frac{dx_2}{dt} \ = -2x_2 + u, \tag{14}$$

with initial conditions $x_1(0) = 1$ and $x_2(0) = 0$. The objective in this subsection is to design a tracking control input $u(t)$ such that the state variable $x_1(t)$ follows a prescribed reference trajectory $r_1(t)$ over the time interval $[0, 10]$, while minimizing the quadratic performance index

$$J = \frac{1}{2}\int_0^{10} \left(2x_1^2 + x_2^2 + u^2\right)\, dt.$$

In this formulation, the control input influences the system dynamics so as to enforce tracking of the first state variable only. Accordingly, the reference signal is defined as

$$r(t) = \begin{bmatrix} r_1(t) \\ 0 \end{bmatrix}.$$

Two types of reference trajectories are considered for $r_1(t)$: a Heaviside step function and a sinusoidal function.

To illustrate the role of the tracking control input $u(t)$, the system defined in Equation (14) is first simulated without the control term. The uncontrolled system response is shown in Figure 1, which serves as a baseline for comparison with the controlled tracking results.

In Figure 1, it can be seen that the system without control variable has a constant solutions, namely $x_1(t) = 1$ and $x_2(t) = 0$. Based on simulation without control, it can be concluded that the control variable plays a significant role in tracking, especially in tracking nonlinear functions such as sinusoidal functions. The construction of the control variable in this tracking problem is solved using a general method and by employing PINN.

In the traditional method, the Hamiltonian equation is obtained as follows.

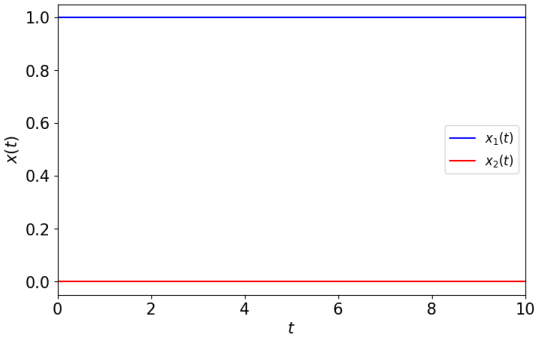

Figure 1: Simulation of system of Equations (14) without control variable.

$$H = \frac{1}{2}\left[(Cx(t) - r(t))^T Q\,(Cx(t) - r(t)) + Ru(t)^2\right] + \lambda^T(Ax + Bu),$$

such that from Equation (7) we have

$$\begin{aligned} \dot{x} &= (A - BK(t))x + BR^{-1}B^T v, \\ K(t) &= R^{-1}B^T S(t), \\ -\dot{S} &= A^T S + SA - SBR^{-1}B^T S + C^T QC, \\ -\dot{v} &= (A - BK)^T v + C^T Qr, \\ u &= -Kx + R^{-1}B^T v. \end{aligned} \tag{15}$$

with $A = \begin{bmatrix} 0 & 1 \\ 0 & -2 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, Q = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, R = \begin{bmatrix} 1 \end{bmatrix}$, and $r(t) = \begin{bmatrix} r_1(t) \\ 0 \end{bmatrix}$. The boundary conditions are specified as

$$S(T) = \begin{bmatrix} 10 & 0 \\ 0 & 0 \end{bmatrix}, \qquad v(T) = \begin{bmatrix} 10\,r_1(t) \\ 0 \end{bmatrix}.$$

The system of differential equations in (15) is solved numerically using a forward–backward Runge–Kutta scheme, where the state equation $\dot{x}$ is integrated forward in time, while the adjoint equations $\dot{S}$ and $\dot{v}$ are integrated backward.

Alternatively, the tracking problem associated with the system (14) is addressed using Physics-Informed Neural Networks (PINNs). Based on equations (9)–(12), the total loss function employed in the PINN framework is defined as:

$$\begin{aligned} \mathcal{L}_{\text{DE}} &= \frac{1}{N_t}\sum_{j=1}^{N_t}\left(\frac{dx_1(t_j)}{dt} - x_2(t_j)\right)^2 + \\ &\quad \frac{1}{N_t}\sum_{j=1}^{N_t}\left(\frac{dx_2(t_j)}{dt} + 2x_2(t_j) - u(t_j)\right)^2, \\ \mathcal{L}_{\text{IC}} &= (x_1(0) - 1)^2 + (x_2(0) - 0)^2, \\ \mathcal{L}_{\text{tracking}} &= \frac{1}{N_t}\sum_{j=1}^{N_t}(x_1(t_j) - r_1(t_j))^2, \\ \mathcal{L}_{\text{cost}} &= \int_0^{10}\left(2x_1^2(t_j) + x_2^2(t_j) + u^2(t_j)\right)dt. \end{aligned}$$

The results obtained using the conventional numerical method are compared with those generated by the PINN approach. Simulations are conducted with $N_t = 1000$, five hidden layers, 128 neurons per

layer, the Adam optimizer, a learning rate of 0.001, and 10,000 training epochs. The solution of the OTC problem in (14) is examined under two different scenarios determined by the reference signal $r_1(t)$.

Case 1: In this case, the reference trajectory is a Heaviside step function defined by

$$r_1(t) = \begin{cases} 1, & \text{for} \quad t > 1, \\ 1/2, & \text{for} \quad t = 1, \\ 0, & \text{for} \quad t < 1. \end{cases}$$

To assess the tracking performance, a quantitative error analysis is carried out by comparing the solutions obtained via the Runge–Kutta (RK) and PINN methods. The PINN is implemented using seven different activation functions to investigate the influence of activation choice on solution accuracy. Under the same hardware specifications, the resulting error metrics for each method are summarized in Table 1.

Table 1: Runge–Kutta and PINN tracking errors in Case 1

Method		$\mathcal{L}_{\text{tracking}}$	$\|e\|_2$	$\frac{\|e\|_2}{\|r\|_2}$	max $\|e\|$	$t_{0.05}$
PINN	tanh$(\cdot)$	0.02012	0.44879	0.14951	0.97782	1.89
	ReLU$(\cdot)$	0.04613	0.67958	0.22640	0.94986	233.32
	Sigmoid$(\cdot)$	0.02683	0.49707	0.16560	0.97239	523.50
	ELU$(\cdot)$	0.01914	0.43776	0.14583	0.99054	1.22
	SELU$(\cdot)$	0.03309	0.57553	0.19173	0.95645	1181.82
	Softplus$(\cdot)$	0.02164	0.46542	0.15505	0.97918	1304.86
	SiLU$(\cdot)$	0.01834	0.42850	0.14275	0.99555	304.33
Runge–Kutta		0.09768	31.28516	10.42259	1.00105	–

In Table 1, the tracking error is defined as the deviation between the numerical solution or model prediction and the desired reference signal, namely $e(t) = x_1(t) - r(t)$. The overall tracking performance is further quantified using the L^2-norm of the error,

$$\|e\|_2 = \left(\int_0^{10} e^2(t)\, dt \right)^{1/2}.$$

To evaluate the relative magnitude of the tracking error, the ratio $\|e\|_2/\|r\|_2$ is also reported. In addition, max $|e|$ denotes the maximum absolute error over the entire time interval, while $t_{0.05}$ represents the time (in seconds) required for the system to reduce the tracking loss $\mathcal{L}_{\text{tracking}}$ below 5%.

Based on the error metrics presented in Table 1, the smallest value of $\mathcal{L}_{\text{tracking}}$ is achieved using the SiLU activation function. Moreover, the PINN-based approach yields lower tracking loss values than the RK method, demonstrating the effectiveness of PINNs in solving linear optimal tracking control problems. The tracking responses obtained using the RK method and the PINN with the SiLU activation function are depicted in Figures 2 and 3.

Based on the simulation results obtained, it can be seen that both methods are capable of solving the OTC problem. From the absolute error in Figure 2, it is obtained that with the PINN method, $x_1(t)$ can

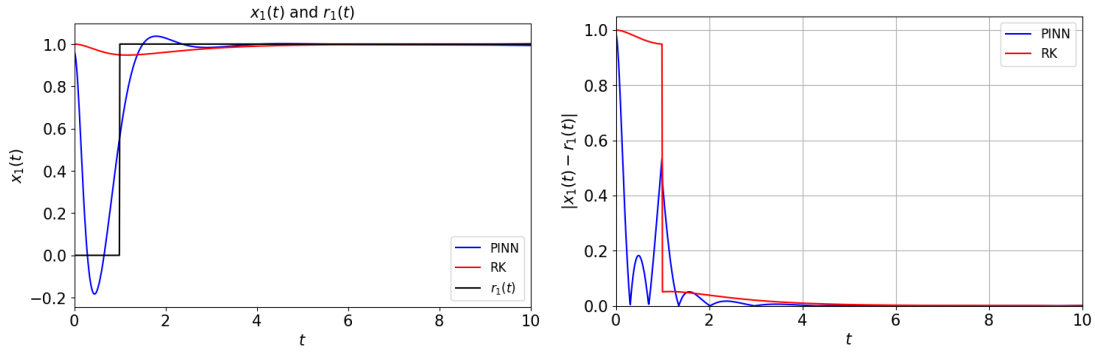

Figure 2: Tracking performance of $x_1(t)$ (left) and absolute error along the domain (right).

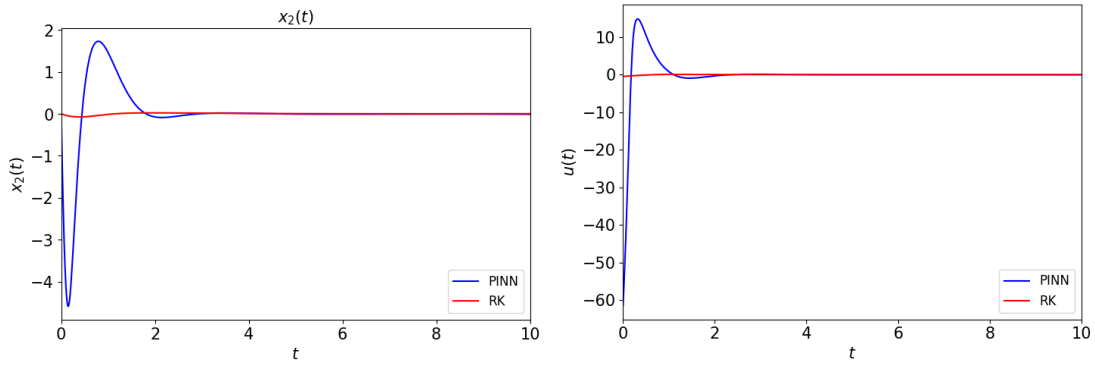

Figure 3: Simulation of $x_2(t)$ and $u(t)$ in Case 1.

approach or track $r_1(t)$ faster than using RK. In the RK method, $x_1(t)$ can track $r_1(t)$ with precision when $t > 2$, so that the effect of $u(t)$ in the tracking process is not very noticeable. Furthermore, in Figure 3, it is obtained that when $t > 2$, $x_2(t)$ and $u(t)$ tend to zero. This is because when $t > 0$, the tracking objective has been achieved, so that the control variable tends to zero, and $x_2(t)$ behaves the same as its analytical solution.

Case 2: In this case, the function being tracked is sinusoidal function $r_1(t) = \sin t$.

With the same hardware specifications as Case 1, the error comparison between the RK and PINN methods is presented in Table 2 below.

The error summary results in Table 2 show that PINN with the SiLU activation function has the lowest $\mathcal{L}_{\text{tracking}}$ value compared to the others. The results of tracking case 2 based on the RK method and the PINN method with the SiLU activation function are illustrated in Figures 4 and 5.

The simulation results in Figure 4 show that $x_1(t)$ using PINN is closer to $r_1(t) = \sin t$ than the RK method. The superiority of PINN is further demonstrated by the absolute error results obtained, where when using PINN the absolute error value approaches zero throughout the domain. We suspect that the shortcomings of the RK method are due to the forward-backward numerical scheme, which affects the stability or accuracy of the solution [4]. Furthermore, in Figure 5, it can be observed that $x_2(t)$ and $u(t)$ change over time. This is due to the fact that the values of $x_2(t)$ and $u(t)$ greatly contribute to the tracking process.

Table 2: Runge–Kutta and PINN tracking errors in Case 2

Method		$\mathcal{L}_{\text{tracking}}$	$\|e\|_2$	$\frac{\|e\|_2}{\|r\|_2}$	$\max\|e\|$	$t_{0.05}$
PINN	$\tanh(\cdot)$	0.01591	0.39907	0.18266	0.97396	101.64
	ReLU$(\cdot)$	0.02402	0.49035	0.22444	0.97158	48.06
	Sigmoid$(\cdot)$	0.01916	0.43801	0.20048	0.97298	163.36
	ELU$(\cdot)$	0.01448	0.38077	0.17428	0.97517	76.36
	SELU$(\cdot)$	0.02430	0.49321	0.22575	0.98943	171.46
	Softplus$(\cdot)$	0.01847	0.43001	0.19681	0.97464	116.39
	SiLU$(\cdot)$	0.01063	0.32623	0.14932	0.98633	38.16
Runge–Kutta		0.58736	76.71629	35.11401	1.99999	–

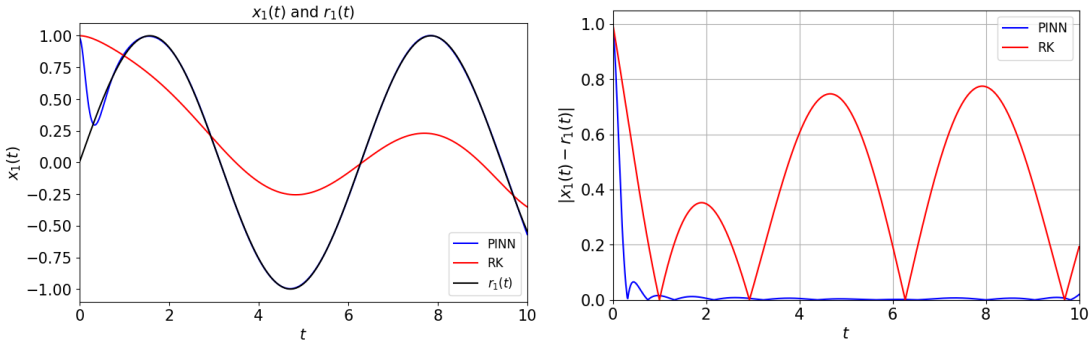

Figure 4: Tracking performance of $x_1(t)$ (left) and absolute error along the domain (right).

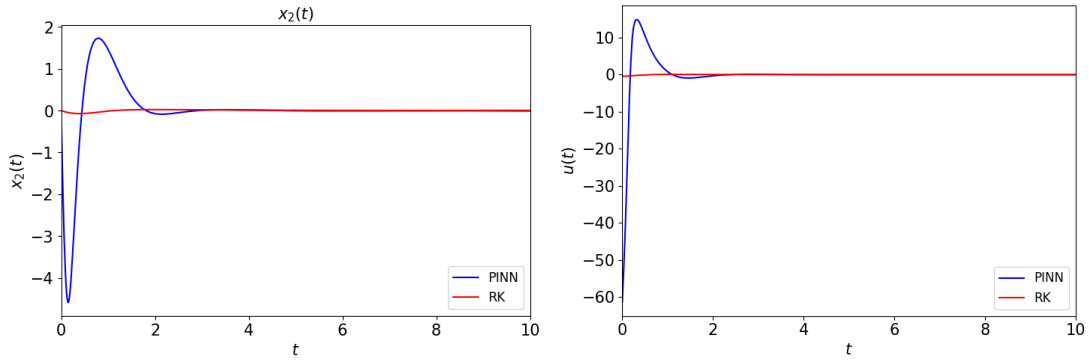

Figure 5: Simulation of $x_2(t)$ and $u(t)$ in Case 2.

4.2 LQ problem for tracking multiple functions

In this section, a modification of the tracking objective is made where a control variable is added, so that the dynamic plant in the System (14) becomes

$$\begin{aligned} \frac{dx_1}{dt} &= x_2 + u_1, \\ \frac{dx_2}{dt} &= -2x_2 + u_2. \end{aligned} \tag{16}$$

This problem aims to ensure that $x_1(t)$ should track $r_1(t)$ and $x_2(t)$ should track $r_2(t)$, subject to minimizing the cost function $J(x,u)$, where

$$r_1(t) = \begin{cases} 2, & \text{for} \quad t > 1 \\ 1, & \text{for} \quad t = 1 \\ 0, & \text{for} \quad t < 1 \end{cases}, \qquad r_2(t) = \begin{cases} 1/2, & \text{for} \quad t > 1 \\ 1/4, & \text{for} \quad t = 1 \\ 0, & \text{for} \quad t < 1 \end{cases},$$

and

$$J(x,u) = \frac{1}{2}\int_0^{10} (2x_1^2 + x_2^2 + u_1^2 + u_2^2)\, dt.$$

Therefore, in the general method we have

$$B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, S(T) = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}, dan\, v(T) = \begin{bmatrix} 10\, r_1(t) \\ 10\, r_2(t) \end{bmatrix}.$$

On the other hand, using the PINN, the loss function is obtained as follows:

$$\begin{aligned} \mathcal{L}_{\text{DE}} &= \frac{1}{N_t}\sum_{j=1}^{N_t}\left(\frac{dx_1(t_j)}{dt} - x_2(t_j) - u_1(t_j)\right)^2 + \\ &\quad \frac{1}{N_t}\sum_{j=1}^{N_t}\left(\frac{dx_2(t_j)}{dt} + 2x_2(t_j) - u_2(t_j)\right)^2, \\ \mathcal{L}_{\text{IC}} &= (x_1(0) - 1)^2 + (x_2(0) - 0)^2, \\ \mathcal{L}_{\text{tracking}} &= \frac{1}{N_t}\sum_{j=1}^{N_t}(x_1(t_j) - r_1(t_j))^2 + \frac{1}{N_t}\sum_{j=1}^{N_t}(x_2(t_j) - r_2(t_j))^2, \\ \mathcal{L}_{\text{cost}} &= \int_0^{10} \big(2x_1^2(t_j) + x_2^2(t_j) + u_1^2(t_j) + u_2^2(t_j)\big)dt. \end{aligned}$$

Similar to Subsection 4.1, a comparison of the RK and the PINN errors with various activation functions was also performed. A summary of the error calculation results is provided in Table 3.

In Table 3, the values in the L^2 error column are the sum of the L^2 tracking errors, namely $\|e_1\|_2 + \|e_2\|_2$, where $e_1(t) = x_1(t) - r_1(t)$ and $e_2(t) = x_2(t) - r_2(t)$. The RE column shows the combined relative error calculated with $\frac{\|e_1\|_2}{\|r_1\|_2} + \frac{\|e_2\|_2}{\|r_2\|_2}$. Then, the $\max_e$ column shows the maximum combined error calculated with $\max|e_1| + \max|e_2|$.

From the error summary results in Table 3, PINN with the SiLU activation function has the lowest $\mathcal{L}_{\text{tracking}}$ value compared to the others. Therefore, the tracking results obtained based on the RK method and the PINN method with the SiLU activation function are illustrated in Figures 6 and 7.

From the results of tracking simulations and absolute errors throughout the domain in Figures 6 and 7, it can be seen that tracking performance using PINN is better than traditional methods. This is also supported by the error summary results in Table 3, where almost all types of PINN activation functions have lower errors than the RK method. Although $x_1(t)$ using RK is able to track $r_1(t)$ for a certain time,

Table 3: Multi-function tracking errors for Runge–Kutta and PINN

Method		$\mathcal{L}_{\text{tracking}}$	L^2 error	RE	max $\lvert e\rvert$	$t_{0.05}$
PINN	tanh$(\cdot)$	0.00671	0.30705	0.07775	1.23010	144.75
	ReLU$(\cdot)$	0.01544	0.46383	0.11656	1.28542	42.35
	Sigmoid$(\cdot)$	0.00831	0.33620	0.08220	1.23113	734.63
	ELU$(\cdot)$	0.01493	0.45479	0.11350	1.26488	300.89
	SELU$(\cdot)$	0.02764	0.62826	0.16210	1.32551	106.34
	Softplus$(\cdot)$	0.01116	0.38884	0.09471	1.23514	82.20
	SiLU$(\cdot)$	**0.00458**	**0.25376**	**0.06430**	**1.21088**	85.45
Runge–Kutta		0.25041	2.23448	0.96626	1.55921	–

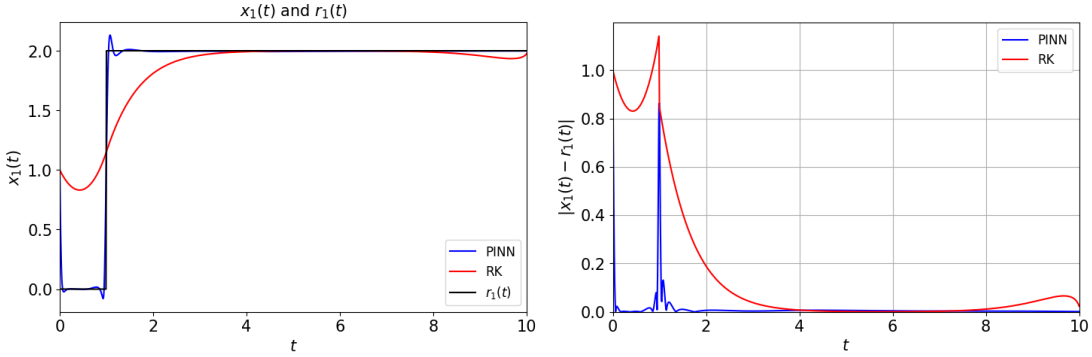

Figure 6: Tracking performance of $x_1(t)$ (left) and absolute error along the domain (right).

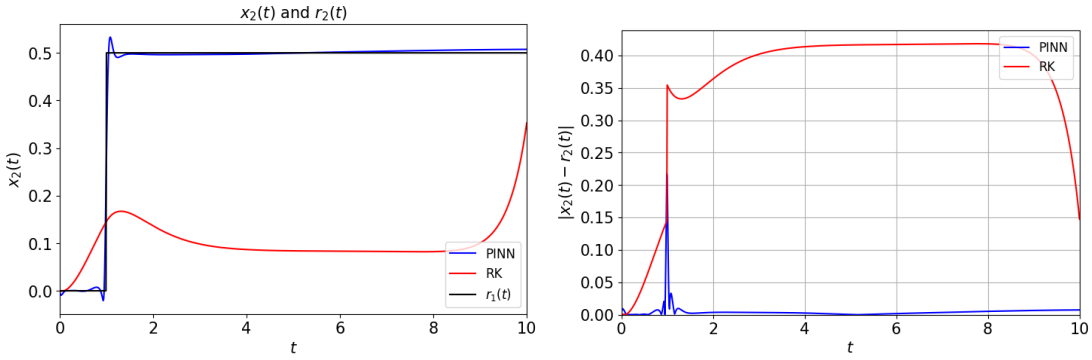

Figure 7: Tracking performance of $x_2(t)$ (left) and absolute error along the domain (right).

for larger values of t, $x_1(t)$ fails to follow the trajectory of $r_1(t)$. Meanwhile, in Figure 7, $x_2(t)$ using RK cannot track $r_2(t)$. The values of the control variables over time are given in Figure 8.

From Figure 8, it can be seen that when $t > 1$, the solution $u_2(t)$ approaches 1. Therefore, for $t > 1$, the second equation in the System (16) can be written as $\dot{x}_2 = 2x_2 + 1$, which has an analytical solution $x_2(t) = -\frac{1}{2}e^{-2t} + \frac{1}{2}$. As $t \to \infty$, $x_2(t)$ approaches $\frac{1}{2}$, which corresponds to the value of $r_2(t)$.

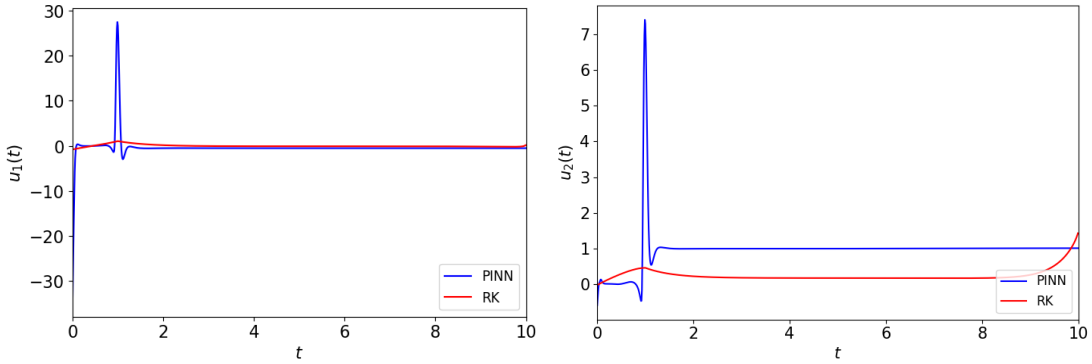

Figure 8: Simulation of $u_1(t)$ and $u_2(t)$ in the System (16).

Furthermore, for $t > 1$, $u_1(t) \approx 0.4$ based on the analytical solution $x_2(t)$, the first equation in the System (16) become $\dot{x}_1 \approx 0$. This indicates that for $t > 1$, $x_1(t)$ does not change over time.

4.3 Nonlinear Tracking Problem

Consider the nonlinear dynamical system

$$\begin{aligned} \frac{dx_1}{dt} &= x_2, \\ \frac{dx_2}{dt} &= -x_1 + x_1 x_2 + u, \end{aligned} \tag{17}$$

subject to the initial conditions $x_1(0) = 1$ and $x_2(0) = 0$. The objective is to design a control input $u(t)$ such that the state variable $x_1(t)$ tracks the reference signal $r_1(t) = \sin t$ over the time interval $[0, T]$. This is achieved by minimizing the performance index

$$J = \int_0^{10} \left(x_1^2(t) + x_2^2(t) + u^2(t) \right) \, dt.$$

As demonstrated in Subsections 4.1 and 4.2, the PINN framework exhibits superior performance in tracking problems when compared to the RK method, particularly in terms of accuracy and stability. In the present nonlinear case, closed-form analytical solutions for the state and co-state equations are not available. Moreover, the RK method relies heavily on time discretization and is sensitive to step-size selection. These limitations motivate the exclusive use of the PINN approach for solving the nonlinear optimal tracking control problem.

The nonlinear tracking problem is addressed using a PINN with the following loss components:

$$
\begin{aligned}
\mathcal{L}_{\text{DE}} &= \frac{1}{N_t}\sum_{j=1}^{N_t}\left(\frac{dx_1(t_j)}{dt} - x_2(t_j)\right)^2 + \\
&\quad \frac{1}{N_t}\sum_{j=1}^{N_t}\left(\frac{dx_2(t_j)}{dt} + x_1(t_j) - x_1(t_j)x_2(t_j) - u(t_j)\right)^2, \\
\mathcal{L}_{\text{IC}} &= (x_1(0)-1)^2 + (x_2(0)-0)^2, \\
\mathcal{L}_{\text{tracking}} &= \frac{1}{N_t}\sum_{j=1}^{N_t}(x_1(t_j) - r_1(t_j))^2, \\
\mathcal{L}_{\text{cost}} &= \int_0^{10}\left(x_1^2(t_j) + x_2^2(t_j) + u^2(t_j)\right)dt.
\end{aligned}
$$

To illustrate the role of the control input, the uncontrolled system dynamics corresponding to (17) are first simulated by setting $u(t) = 0$. The resulting behavior is shown in Figure 9, where it is evident that the state $x_1(t)$ fails to follow the reference signal in the absence of control.

Since this section focuses exclusively on the PINN-based solution, Table 4 summarizes the tracking error metrics obtained using different activation functions. Among all tested activations, the PINN with the SiLU activation function achieves the lowest tracking loss $\mathcal{L}_{\text{tracking}}$. Consequently, the corresponding tracking results are presented in Figures 10 and 11.

Table 4: Nonlinear system tracking error

Method		$\mathcal{L}_{\text{tracking}}$	$\|e\|_2$	$\frac{\|e\|_2}{\|r\|_2}$	$\max\|e\|$	$t_{0.05}$
PINN	tanh$(\cdot)$	0.01396	0.37386	0.17112	0.98046	27.14
	ReLU$(\cdot)$	0.02324	0.48239	0.22079	0.96565	49.68
	Sigmoid$(\cdot)$	0.01944	0.44120	0.20194	0.97321	191.33
	ELU$(\cdot)$	0.01474	0.38417	0.17584	0.97766	22.44
	SELU$(\cdot)$	0.02754	0.52509	0.24034	0.95613	120.22
	Softplus$(\cdot)$	0.02122	0.46096	0.21099	0.97013	29.90
	SiLU$(\cdot)$	**0.01277**	**0.35756**	**0.16366**	**0.98335**	32.77

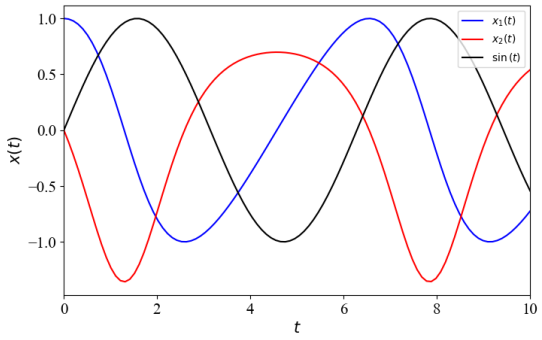

Figure 9: Simulation of system of Equations (17) without control variable.

As shown in Figure 10, the state $x_1(t)$ produced by the PINN accurately tracks the reference signal $r_1(t) = \sin t$. A comparison with the uncontrolled case in Figure 9 clearly demonstrates the essential

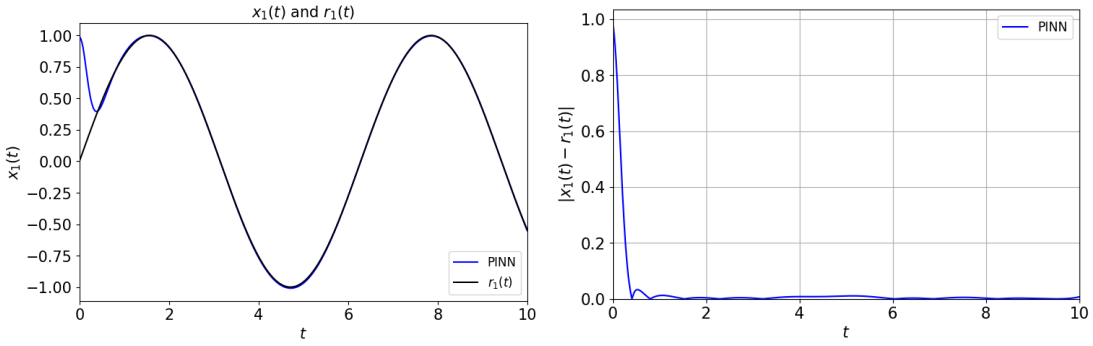

Figure 10: Tracking performance of $x_1(t)$ (left) and absolute error along the domain (right).

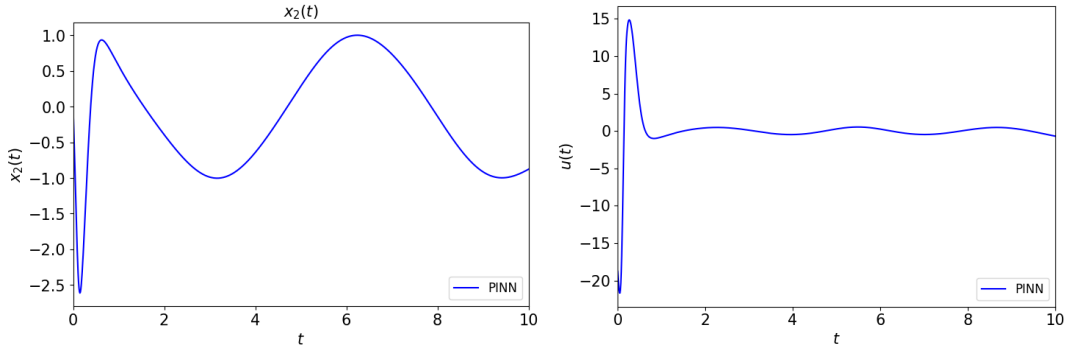

Figure 11: Simulation of $x_2(t)$ and $u(t)$ in nonlinear tracking.

role of the control input $u(t)$ in achieving the tracking objective. The effectiveness of the PINN approach is further confirmed by the absolute tracking error, which remains close to zero throughout the time domain. Figure 11 illustrates the evolution of $x_2(t)$ and the corresponding control input $u(t)$. When $x_2(t)$ is negative, the state $x_1(t)$ decreases, and when $x_2(t)$ is positive, $x_1(t)$ increases, consistent with the first equation of the system dynamics. Moreover, $x_2(t)$ exhibits oscillatory behavior resembling a sinusoidal waveform, reflecting its coupling with $x_1(t)$, which closely follows the sinusoidal reference trajectory.

5 Conclusion

This paper explores the application of Physics-Informed Neural Networks (PINNs) to the solution of optimal tracking control (OTC) problems. PINNs with various activation functions are examined, and the results demonstrate that all considered activation choices are capable of successfully solving tracking problems in both linear (Linear Quadratic) and nonlinear dynamical systems. Among them, the PINN employing the SiLU activation function consistently achieves the lowest tracking errors, indicating superior performance relative to other activation functions. In terms of accuracy, the PINN-based approach is shown to construct control inputs that track the desired reference signal with higher precision than the conventional Runge–Kutta (RK) method. Moreover, the PINN framework enables the tracking problem to be solved without requiring elaborate analytical derivations, which represents a practical advantage in

complex control settings. However, PINN-based simulations generally incur significantly higher computational costs compared to traditional numerical methods, resulting in longer execution times. These findings highlight the importance of hardware capabilities when employing PINNs, as advanced computational resources allow for larger network architectures, increased numbers of neurons per layer, and extended training epochs, thereby improving solution quality and efficiency. From a theoretical perspective, rigorous results on the stability and convergence of PINNs are not yet fully established, indicating a need for further analytical investigation. Although PINNs constitute a promising alternative for solving OTC problems, their practical applicability requires further validation, particularly in real-world industrial tracking scenarios. Future research may focus on hybrid frameworks that combine PINNs with established control methodologies, such as Model Predictive Control (MPC), the Theory of Functional Connections (TFC), or adaptive loss-balancing strategies, to enhance performance and efficiency in nonlinear control applications.

Declarations

Availability of Supporting Data

All data generated or analyzed during this study are included in this published paper.

Funding

The authors conducted this research without any funding, grants, or support.

Conflict of Interest

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

Author Contributions

Fidelis Nofertinus Zai contributed to the formal analysis, methodology, software, writing – original draft preparation. **Rian Kurnia** contributed to methodology, writing – original draft preparation, writing – review and editing. **Juan Prihanda Nainggolan** contributed to visualization, and validation of the results. All authors have read and approved the published version of the manuscript.

Artificial Intelligence Statement

The authors used AI-based tools solely as part of their writing and editing workflow. Specifically, AI-assisted capabilities were employed to improve language quality, clarity, grammar, and stylistic consistency. The authors did not rely on AI to generate original scientific content, data, analyses, or interpretations.

Publisher's Note

The publisher remains neutral regarding jurisdictional claims in published maps and institutional affiliations.

References

[1] Ahmadin, A., Naiborhu, J., Mu'tamar, K. (2023). "Control design on a non-minimum phase bilinear system by output redefinition and particle swarm optimization method". In: *2023 6th International Conference on Robotics, Control and Automation Engineering (RCAE), Suzhou, China* IEEE., 201–206. https://doi.org/10.1109/RCAE59706.2023.10398810

[2] Ali, I. (2025). "Data-driven machine learning approach based on physics-informed neural network for population balance model". *Advances in Continuous and Discrete Models*, 2025(12). https://doi.org/10.1186/s13662-025-03876-1

[3] Beard, R.W., Saridis, G.N., Wen, J.T. (1997). "Galerkin approximations of the generalized Hamilton-Jacobi-Bellman equation". *Automatica*, 33(12), 2159–2177. https://doi.org/10.1016/S0005-1098(97)00128-3

[4] Bryson, A. E., Ho, Y. C. (1975). *Applied optimal control*, Taylor and Francis Group.

[5] Chen, X., Wang, F. (2021). "Neural-network-based stochastic linear quadratic optimal tracking control scheme for unknown discrete-time systems using adaptive dynamic programming". *Control Theory and Technology*, 19(3), 315-327. https://doi.org/10.1007/s11768-021-00046-y

[6] Chu, H., Miyatake, Y., Cui, W., Wei, S., Furihata, D. (2024). "Structure-preserving physics-informed neural networks with energy or Lyapunov structure". *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24)*, 3872-3880. https://doi.org/10.24963/ijcai.2024/428

[7] Cong, A., Jin, Y., Lu, Z., Gao, Q., Ge, X., Li, Z., Lin, R., Hu, X., Hou, L. (2025). "Transfer learning-based physics-informed DeepONets for the adaptive evolution of digital twin models for dynamic systems". *Nonlinear Dynamics*, 113(15), 19075–19102. https://doi.org/10.1007/s11071-025-11158-4

[8] Devasia, S., Chen, D., Paden, B. (1996). "Nonlinear inversion-based output tracking". *IEEE Transactions on Automatic Control*, 41(7), 930-942. https://doi.org/10.1109/9.508898

[9] Fabiani, G., Bollt, E., Siettos, C., Yannacopoulos, A.N. (2025). "Linear stability analysis of physics-informed random projection neural networks for ODEs", *arXiv*. https://doi.org/10.48550/arXiv.2408.15393

[10] Feng, Y., Eun, J., Kim, S., Kim, Y.R. (2025). "Application of physics-informed neural networks (PINNs) solution to coupled thermal and hydraulic processes in silty sands". *International Journal of Geo-Engineering*, 16(3). https://doi.org/10.1186/s40703-025-00232-w

[11] Gao, B., Yao, R., Li, Y. (2025). "Physics-informed neural networks with adaptive loss weighting algorithm for solving partial differential equations". *Computers & Mathematics with Applications*, 181, 216-227. https://doi.org/10.1016/j.camwa.2025.01.007

[12] Karniadakis, G.E., Kevrekidis, I.G., Lu, L., Perdikaris, P., Wang, S., Yang, L. (2021). "Physics-informed machine learning". *Nature Reviews Physics*, 3, 422–440. https://doi.org/10.1038/s42254-021-00314-5

[13] Lewis, F.L., Vrabie, D.L., Syrmos, V.L. (2012). "Optimal control". 3rd Edition, *John Wiley & Sons, INC.*

[14] Li, Y., Liu, L. (2024). "Physics-informed neural network-based nonlinear model predictive control for automated guided vehicle trajectory tracking". *World Electric Vehicle Journal*, 15(10), 460. https://doi.org/10.3390/wevj15100460

[15] Michałowska, K., Goswami, S., Karniadakis, G.E., Riemer-Sørensen, S. (2025). "Multi-resolution learning with DeepONets and long short-term memory neural networks". *Neurocomputing*, 653, 131154. https://doi.org/10.1016/j.neucom.2025.131154

[16] Mu, C., Ni, Zh., Sun, C., He, H. (2017). "Air-breathing hypersonic vehicle tracking control based on adaptive dynamic programming". *IEEE Transactions on Neural Networks and Learning Systems*, 28(3), 584-598. https://doi.org/10.1109/TNNLS.2016.2516948

[17] Mu'tamar, K., Naiborhu, J., Saragih, R., Handayani, D. (2022). "Tracking control for planar nonminimum-phase bilinear control system with disturbance using backstepping". *Indonesian Journal of Electrical Engineering and Computer Science*, 26(3), 1315–1327. https://doi.org/10.11591/ijeecs.v26.i3.pp1315-1327

[18] Pratama, M. H. Y., Gunawan, A. Y. (2023). "Exploring physics-informed neural networks for solving boundary layer problems". *Journal of Fundamental Mathematics and Applications*, 6(2), 101-116. https://doi.org/10.14710/jfma.v6i2.20084

[19] Tang, G., Fan, M. (2008). "Series-based approximate approach of optimal tracking control for nonlinear systems with time-delay". *Progress in Natural Science*, 18. https://doi.org/10.1016/j.pnsc.2008.03.033

[20] Wang, D., Liu, D., Wei, Q. (2012). "Finite-horizon neuro-optimal tracking control for a class of discrete-time nonlinear systems using adaptive dynamic programming approach". *Neurocomputing*, 78(1), 14-22. https://doi.org/10.1016/j.neucom.2011.03.058

[21] Wei, Q., Liu, D. (2014). "Adaptive dynamic programming for optimal tracking control of unknown nonlinear systems with application to coal gasification". *IEEE Transactions on Automation Science and Engineering*, 11, 1020-1036. https://doi.org/10.1109/TASE.2013.2284545

[22] Zerrougui, I., Li, Z., Hissel, D. (2025). "Physics-informed neural network for modeling and predicting temperature fluctuations in proton exchange membrane electrolysis". *Energy and AI*, 20, 100474. https://doi.org/10.1016/j.egyai.2025.100474

[23] Zhu, P., Liu, Z., Xu, Z., Lv, J. (2025). "An adaptive weight physics-informed neural network for vortex-induced vibration problems". *Buildings*, 15(9), 1533. https://doi.org/10.3390/buildings15091533

Authors Bio-sketches

Fidelis Nofertinus Zai is a lecturer in the Mathematics Undergraduate Program at the University of North Sumatra. He completed his undergraduate studies at the State University of Medan and his master's studies at the Bandung Institute of Technology. His research interests include control theory, optimization, differential equations, and dynamic systems. Corresponding author: Email: fidelis@usu.ac.id

Rian Kurnia is a lecturer in the Data Science Undergraduate Program at the Sumatra Institute of Technology. He completed his undergraduate studies at the Bogor Agricultural Institute and his master's studies at the Bandung Institute of Technology. His research interests include algebra, optimization, biostatistics, and data analysis.

Juan Prihanda Nainggolan is a senior student in the Mathematics Undergraduate Program at the University of North Sumatra. He is currently pursuing research in differential equations through his final project.