Integral Error-Based Adaptive Neural Identifier for a Class of Uncertain Nonlinear Systems

Donghwa Hong[1] and Kyunghwan Choi[2]*

[1]Department of Mechanical Robotics Engineering, GIST,
Seoul, 13391, Korea (first@hankook.ac.kr)
[2]Department of Mechanical Engineering, Cho,
Seoul, 13391, Korea (second@hankook.ac.kr) * Corresponding author

Abstract: This paper presents a robust online neural network (NN) identifier for uncertain nonlinear systems, focusing on online system identification and function approximation. The proposed method employs a cumulative error cost function with a forgetting factor to enable real-time learning of unknown nonlinear dynamics. A rigorous Lyapunov-based stability analysis guarantees Uniform Ultimate Boundedness (UUB) of the identification error. Comparative analysis with state error-based gradient descent methods highlights the improved stability and convergence properties of the proposed approach. The results demonstrate that the method enables effective online identification and function approximation for a wide range of nonlinear systems.

Keywords: Online System Identification, Neural Networks, Adaptive Control, Stability Analysis, Nonlinear Dynamics

1. INTRODUCTION

System identification is the process of estimating the dynamic model of an unknown plant from input-output data, which is essential for controller design and state estimation. In particular, when the system dynamics are nonlinear or time-varying, the importance of online identification becomes more pronounced. Offline identification techniques assume large batches of data and lack real-time adaptability, making them unsuitable for dynamic environments. In contrast, online identification updates the model in real time, enabling the tracking of changing system characteristics, and is thus indispensable in dynamic control applications.

Traditional online system identification methods mainly employ Recursive Least Squares (RLS)-based algorithms. RLS updates parameters by weighting past data and guarantees convergence when the system is static. However, the forgetting factor used in RLS presents a trade-off between convergence speed and tracking ability, and the choice of an appropriate forgetting factor greatly affects real-time estimation performance. To address this issue, variable forgetting factor (VFF) techniques have been proposed, which adaptively adjust the forgetting factor.

Neural network-based identification methods have been actively studied due to their flexible modeling capabilities for nonlinear systems. Neural networks possess universal approximation properties, meaning that with appropriate size, structure, and weights, they can approximate any continuous nonlinear function arbitrarily well over a compact set, thus effectively learning complex nonlinearities. For example, multilayer perceptrons or recurrent neural networks can be used as identification models to learn input-output relationships. The conventional backpropagation algorithm updates neural network weights by minimizing an instantaneous squared error cost function, but this approach inherently focuses only on instantaneous errors, making it difficult to guarantee long-term parameter convergence. To enhance robustness, stabilization techniques such as σ-modification and composite adaptation have been proposed.

The limitations of existing methods are as follows. First, most algorithms use cost functions that minimize instantaneous errors, causing model parameters to focus only on momentary discrepancies and making it difficult to achieve accurate function approximation over the entire time interval. Second, persistent excitation (PE) conditions are often required for parameter convergence in neural network identification, but PE may be lost once the control objective is achieved and the system reaches steady state. Third, in nonlinear system identification and control, stability proofs are often lacking, making practical application to real systems difficult.

This paper proposes a robust online neural network identification method that applies a cumulative error cost function with a forgetting factor to overcome the limitations of previous studies. Specifically, by defining a new cost function that accumulates identification errors at all time points, the method aims to minimize the overall error in a balanced manner. In addition, a variable forgetting factor is introduced to maintain adaptability to recent data while gradually reducing the influence of past information. The proposed method is rigorously analyzed using Lyapunov-based stability analysis to guarantee the Uniform Ultimate Boundedness (UUB) of the identification error. Comparative analysis with state error-based gradient descent methods demonstrates the improved stability and convergence properties of the proposed approach. The results of this study show that the method enables effective online identification and function approximation for a wide range of nonlinear systems.

2. PROBLEM FORMULATION

2.1 Model Dynamics

Consider the nonlinear system

$$\dot{\boldsymbol{x}}(t) = \underbrace{\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})}_{known} + \underbrace{\boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u})}_{unknown} \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^{\ltimes}$ is the state vector and $\boldsymbol{u} \in \mathbb{R}^{\gtrdot}$ is the control input vector, and $\boldsymbol{f}(\cdot)$ represents the known part of the system dynamics, and $\boldsymbol{h}(\cdot)$ denotes the unknown nonlinear dynamics.

2.2 Neural Network Identifier

To design a identifier, some assumptions are needed as follows :

Assumption 1: The ideal weights $\mathbf{W}, \mathbf{V}$ are bounded. The disturbance $w(t)$ is bounded by $\|w(t)\| \leq w_M$. The activation function σ and its derivatives are bounded.

Assumption 2 (Open-Loop Stability) The open-loop system (1) is stable, which implies that the state vector $\boldsymbol{x}(t)$ is bounded in L_∞.

By adding and subtracting $\mathbf{A}\boldsymbol{x}$ from . So, the system is described by:

$$\dot{\boldsymbol{x}}(t) = \mathbf{A}\mathbf{x}(\mathbf{t}) + \mathbf{g}(\mathbf{x}, \mathbf{u}) + \mathbf{h}(\mathbf{x}, \mathbf{u}) \tag{2}$$

where $\mathbf{A} \in \mathbb{R}^{\ltimes \times \ltimes}$ is an arbitrary Hurwitz matrix, $\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) - \mathbf{A}\boldsymbol{x}$, which is the known nonlinear function.

Then, the identifier model can be selected as:

$$\dot{\hat{\boldsymbol{x}}}(t) = \mathbf{A}\hat{\boldsymbol{x}}(t) + \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) + \hat{\boldsymbol{h}}(\hat{\bar{\boldsymbol{x}}}) \tag{3}$$

where the NN output is $\hat{\boldsymbol{h}}(\hat{\boldsymbol{x}}, \boldsymbol{u}) = \hat{\mathbf{W}}^T\sigma(\hat{\mathbf{V}}^T\hat{\bar{\boldsymbol{x}}})$. Here, $\hat{\mathbf{W}} \in \mathbb{R}^{\eqsim \times \ltimes}$ and $\hat{\mathbf{V}} \in \mathbb{R}^{\times \eqsim}$ are the estimated weight matrices. The NN input $\hat{\bar{\boldsymbol{x}}}$ is constructed from the estimated states $\hat{\boldsymbol{x}}$ and $\boldsymbol{u}$. Defining the errors $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \hat{\boldsymbol{x}}$, $\tilde{\mathbf{W}} = \mathbf{W} - \hat{\mathbf{W}}$, and $\tilde{\mathbf{V}} = \mathbf{V} - \hat{\mathbf{V}}$, the error dynamics are given by:

$$\dot{\tilde{\boldsymbol{x}}} = \mathbf{A}\tilde{\boldsymbol{x}} + \tilde{\mathbf{W}}^T\sigma(\hat{\mathbf{V}}^T\hat{\bar{\boldsymbol{x}}}) + w(t) \tag{4}$$

where $w(t) = \mathbf{W}^T\left(\sigma(\mathbf{V}^T\bar{\boldsymbol{x}}) - \sigma(\hat{\mathbf{V}}^T\hat{\bar{\boldsymbol{x}}})\right) + \epsilon(x)$ is the lumped disturbance term.

2.3 Conventional Adpative Law

A common approach to update the weights of a neural network is gradient descent method, which minimizes the cost function defined as the squared error.

Consider the plant model (2) and the identifier model (3). Given Assumption 2, if the weights of the NLPNN are updated according to

$$\dot{\hat{\mathbf{W}}} = -\eta_1\left(\frac{\partial J}{\partial \hat{\mathbf{W}}}\right) - \rho_1\|\tilde{\boldsymbol{x}}\|\hat{\mathbf{W}} \tag{5}$$

$$\dot{\hat{\mathbf{V}}} = -\eta_2\left(\frac{\partial J}{\partial \hat{\mathbf{V}}}\right) - \rho_2\|\tilde{\boldsymbol{x}}\|\hat{\mathbf{V}}, \tag{6}$$

where $\eta > 0$ is the learning rate, $\rho > 0$ is the leakage rate, and $\frac{\partial J}{\partial \hat{\mathbf{W}}}$ and $\frac{\partial J}{\partial \hat{\mathbf{V}}}$ are the gradients of the cost function with respect to the weights. And the cost function is defined as

$$J = \frac{1}{2}\tilde{\boldsymbol{x}}(t)^T\tilde{\boldsymbol{x}}(t) \tag{7}$$

where $\tilde{\boldsymbol{x}}(t) = \boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)$ is the state error. The cost function J is minimized by updating the weights of the neural network using the gradient descent method.

However, since the cost function is based solely on the instantaneous error, the model parameters tend to focus on minimizing short-term discrepancies rather than ensuring consistent function approximation over the entire time interval. Consequently, it becomes challenging to guarantee accurate and uniform approximation performance throughout the whole time horizon, particularly when the system dynamics are time-varying. Therefore, it is necessary to consider a cost function that captures the overall behavior of the system in the time domain.

2.4 Integral Adaptive Law based Gradient Descent

To address the limitations of the instantaneous error cost function, we define an integrated sqaured error cost functional as follows:

$$J = \frac{1}{2}\int_0^t e^{-\lambda(t-\tau)}\tilde{\boldsymbol{x}}(\tau)^T\tilde{\boldsymbol{x}}(\tau)\,\mathrm{d}\tau \tag{8}$$

where $\lambda > 0$ is called forgetting factor. This function accumulates the squared error over time, with a forgetting factor that gradually reduces the influence of past errors.

Since the cost functional is of an integral form, we first introduce the filtered error signal $\boldsymbol{z}(t)$ to construct the update laws.

$$\boldsymbol{z}(t) = \int_0^t e^{-\lambda(t-\tau)}\tilde{\boldsymbol{x}}(\tau)\,\mathrm{d}\tau \tag{9}$$

$$\dot{\boldsymbol{z}} = -\lambda\boldsymbol{z} + \tilde{\boldsymbol{x}} \tag{10}$$

Let us define

$$\mathrm{net}_{\hat{\mathbf{V}}} = \hat{\mathbf{V}}\hat{\boldsymbol{x}}$$
$$\mathrm{net}_{\hat{\mathbf{W}}} = \hat{\mathbf{W}}\sigma(\hat{\mathbf{V}}\hat{\boldsymbol{x}}).$$

Therefore, by using the chain rule $\frac{\partial J}{\partial \hat{\mathbf{W}}}$ and $\frac{\partial J}{\partial \hat{\mathbf{V}}}$ can be computed according to

$$\frac{\partial J}{\partial \hat{\mathbf{W}}} = \frac{\partial J}{\partial \mathrm{net}_{\hat{\mathbf{W}}}} \cdot \frac{\partial \mathrm{net}_{\hat{\mathbf{W}}}}{\partial \hat{\mathbf{W}}}$$
$$\frac{\partial J}{\partial \hat{\mathbf{V}}} = \frac{\partial J}{\partial \mathrm{net}_{\hat{\mathbf{V}}}} \cdot \frac{\partial \mathrm{net}_{\hat{\mathbf{V}}}}{\partial \hat{\mathbf{V}}},$$

where

$$\frac{\partial J}{\partial \mathrm{net}_{\hat{\mathbf{W}}}} = \frac{\partial J}{\partial \tilde{\boldsymbol{x}}}\frac{\partial \tilde{\boldsymbol{z}}}{\partial \hat{\boldsymbol{x}}}\frac{\partial \hat{\boldsymbol{x}}}{\partial \mathrm{net}_{\hat{\mathbf{W}}}} = -\boldsymbol{z}^T\frac{\partial \hat{\boldsymbol{x}}}{\partial \mathrm{net}_{\hat{\mathbf{W}}}},$$
$$\frac{\partial J}{\partial \mathrm{net}_{\hat{\mathbf{V}}}} = \frac{\partial J}{\partial \tilde{\boldsymbol{x}}}\frac{\partial \tilde{\boldsymbol{x}}}{\partial \hat{\boldsymbol{x}}}\frac{\partial \hat{\boldsymbol{x}}}{\partial \mathrm{net}_{\hat{\mathbf{V}}}} = -\boldsymbol{z}^T\frac{\partial \hat{\boldsymbol{x}}}{\partial \mathrm{net}_{\hat{\mathbf{V}}}}$$

and

$$\frac{\partial \text{net}_{\hat{\mathbf{W}}}}{\partial \hat{\mathbf{W}}} = \sigma(\hat{\mathbf{V}}\hat{\boldsymbol{x}})$$
$$\frac{\partial \text{net}_{\hat{\mathbf{V}}}}{\partial \hat{\mathbf{V}}} = \hat{\boldsymbol{x}}.$$

We modify the original BP algorithm such that the static approximations of $\frac{\partial \hat{\boldsymbol{x}}}{\partial \text{net}_{\hat{\mathbf{W}}}}$ and $\frac{\partial \hat{\boldsymbol{x}}}{\partial \text{net}_{\hat{\mathbf{V}}}}$ ($\dot{\hat{\boldsymbol{x}}} = 0$) can be used.

$$\frac{\partial \hat{\boldsymbol{x}}}{\partial \text{net}_{\hat{\mathbf{W}}}} \approx -\mathbf{A}^{-\mathbf{1}}$$
$$\frac{\partial \hat{\boldsymbol{x}}}{\partial \text{net}_{\hat{\mathbf{V}}}} \approx -\mathbf{A}^{-\mathbf{1}}\hat{\mathbf{W}}(\mathbf{I} - \mathbf{\Lambda}(\hat{\mathbf{V}}_i\hat{\boldsymbol{x}})),$$

where

$$\mathbf{\Lambda}(\hat{\mathbf{V}}_i\hat{\boldsymbol{x}}) = \text{diag}\{\sigma_i^2(\hat{\mathbf{V}}_i\hat{\boldsymbol{x}})\}, \quad i = 1, 2, \ldots, m.$$

Then, the update laws for the weights $\hat{\mathbf{W}}$ and $\hat{\mathbf{V}}$ can be expressed as:

$$\dot{\hat{\mathbf{W}}} = -\eta_1 \left(\boldsymbol{z}^T \boldsymbol{A}^{-1}\right)^T \sigma(\hat{\mathbf{V}}^T\hat{\boldsymbol{x}})^T - \rho_1\|\tilde{\boldsymbol{x}}\|\hat{\mathbf{W}} \tag{11}$$
$$\dot{\hat{\mathbf{V}}} = -\eta_2\hat{\boldsymbol{x}} \left(\boldsymbol{z}^T \boldsymbol{A}^{-1}\hat{\mathbf{W}}(\mathbf{I} - \mathbf{\Lambda}(\hat{\mathbf{V}}_i\hat{\boldsymbol{x}}))\right)^T - \rho_2\|\tilde{\boldsymbol{x}}\|\hat{\mathbf{V}} \tag{12}$$

2.5 Stability Analysis

To analyze the stability of the system described by (4) with the update laws (11)-(12), we will use Lyapunov's direct method. The goal is to show that the errors $\tilde{\boldsymbol{x}}$, $\tilde{\mathbf{W}}$, and $\tilde{\mathbf{V}}$ are Uniformly Ultimately Bounded (UUB).

Theorem 1: For the system given by (4) with the update laws (11)-(12), all signals in the system $(\tilde{\boldsymbol{x}}, \tilde{\mathbf{W}}, \tilde{\mathbf{V}})$ are Uniformly Ultimately Bounded.

Proof: The stability proof is conducted in two steps using a cascaded system approach.

We first prove the boundedness of the state error $\tilde{\boldsymbol{x}}$ and the output layer weight error $\tilde{\mathbf{W}}$. This is possible because the term $\sigma_v = \sigma(\hat{\mathbf{V}}^T\hat{\boldsymbol{x}})$ in the error dynamics (4) is always bounded, regardless of the value of $\hat{\mathbf{V}}$, due to the bounded nature of the activation function σ.

Consider the Lyapunov function candidate for the first subsystem:

$$L = \frac{1}{2}\tilde{\boldsymbol{x}}^T\mathbf{P_1}\tilde{\boldsymbol{x}} + \frac{1}{2}\,\text{tr}(\tilde{\mathbf{W}}^T\rho_1^{-1}\tilde{\mathbf{W}}) + \frac{1}{2}\int_0^t e^{-\lambda(t-\tau)}\tilde{\boldsymbol{x}}(\tau)^T\mathbf{P_2}\tilde{\boldsymbol{x}}(\tau)\,\mathrm{d}\tau \tag{13}$$

Its time derivative, after substituting the error dynamics, is:

$$\dot{L} = -\frac{1}{2}\tilde{\boldsymbol{x}}^T(\mathbf{Q_1} - \mathbf{P_2})\tilde{\boldsymbol{x}} + \tilde{\boldsymbol{x}}^T\mathbf{P_1}(\tilde{\mathbf{W}}^{\mathbf{T}}\sigma_{\mathbf{v}} + \mathbf{w}) + \text{tr}(\dot{\tilde{\mathbf{W}}}^{\mathbf{T}}\rho_{\mathbf{1}}^{-\mathbf{1}}\tilde{\mathbf{W}}) - \lambda\mathbf{L}_{\text{int}}$$

where $\sigma_v = \sigma(\hat{\mathbf{V}}^T\hat{\boldsymbol{x}})$, $\mathbf{Q} = \mathbf{Q_1} - \mathbf{P_2} > \mathbf{0}$, and $\boldsymbol{L}_{\text{int}} = \frac{1}{2}\int_0^t e^{-\lambda(t-\tau)}\tilde{\boldsymbol{x}}(\tau)^T\mathbf{P_2}\tilde{\boldsymbol{x}}(\tau)\,\mathrm{d}\tau$.

And we substitute the update law (11) using $\dot{\tilde{\mathbf{W}}} = -\dot{\hat{\mathbf{W}}}$.

$$\begin{aligned}\text{tr}(\dot{\tilde{\mathbf{W}}}^{\mathbf{T}}\rho_{\mathbf{1}}^{-\mathbf{1}}\tilde{\mathbf{W}}) &= \text{tr}\left(\left(\eta_W\mathbf{A}^{-T}\boldsymbol{z}\sigma_v^T + \rho\|\tilde{\boldsymbol{x}}\|\hat{\mathbf{W}}\right)^T\rho_1^{-1}\tilde{\mathbf{W}}\right)\\ &= \eta_W\,\text{tr}(\sigma_v\boldsymbol{z}^Tl_1\tilde{\mathbf{W}})\\ &+ \|\tilde{\boldsymbol{x}}\|\,\text{tr}(\hat{\mathbf{W}}^T\tilde{\mathbf{W}})\end{aligned}$$

We expand the leakage term by substituting $\hat{\mathbf{W}} = \mathbf{W} - \tilde{\mathbf{W}}$:

$$\begin{aligned}\|\tilde{\boldsymbol{x}}\|\,\text{tr}(\hat{\mathbf{W}}^T\tilde{\mathbf{W}}) &= \|\tilde{\boldsymbol{x}}\|\,\text{tr}((\mathbf{W} - \tilde{\mathbf{W}})^{\mathbf{T}}\tilde{\mathbf{W}})\\ &= -\|\tilde{\boldsymbol{x}}\|\|\tilde{\mathbf{W}}\|^2 + \|\tilde{\boldsymbol{x}}\|\|\mathbf{W}\|\|\tilde{\mathbf{W}}\|\end{aligned}$$

Substituting this back into the $\dot{L}$ expression:

$$\begin{aligned}\dot{L} \leq &-\frac{1}{2}\tilde{\boldsymbol{x}}^T\mathbf{Q}\tilde{\boldsymbol{x}} - \|\tilde{\boldsymbol{x}}\|\|\tilde{\mathbf{W}}\|^2 - \lambda\boldsymbol{L}_{\text{int}}\\ &+ \tilde{\boldsymbol{x}}^T\mathbf{P_1}(\tilde{\mathbf{W}}^T\sigma_v + w) + \eta_W\,\text{tr}(\sigma_v\boldsymbol{z}^T\|\mathbf{A}^{-\mathbf{1}}\rho_1^{-1}\|\tilde{\mathbf{W}})\\ &+ \|\tilde{\boldsymbol{x}}\|\|\mathbf{W}\|\|\tilde{\mathbf{W}}\|\end{aligned}$$

Moreover, we have

$$\begin{aligned}|\tilde{\boldsymbol{x}}^T\mathbf{P_1}\tilde{\mathbf{W}}^{\mathbf{T}}\sigma_{\mathbf{v}}| &\leq \|\tilde{\boldsymbol{x}}\|\|\mathbf{P_1}\|(\|\tilde{\mathbf{W}}\|\sigma_{\mathbf{M}} + \bar{\mathbf{w}})\\ \|\tilde{\boldsymbol{x}}\|\|\mathbf{W}\|\|\tilde{\mathbf{W}}\| &\leq \|\tilde{\boldsymbol{x}}\|\|\tilde{\mathbf{W}}\|W_M\\ |\eta_W\,\text{tr}(\sigma_v\boldsymbol{z}^Tl_1\tilde{\mathbf{W}})| &\leq \eta_W\|\sigma_v\|\|\boldsymbol{z}\|\|\mathbf{A}^{-\mathbf{1}}\rho_1^{-1}\|\|\tilde{\mathbf{W}}\|\\ &\leq \eta_W\sigma_M\frac{\sqrt{n}}{\lambda}\|\tilde{\boldsymbol{x}}\|\|\mathbf{A}^{-\mathbf{1}}\rho_1^{-1}\|\|\tilde{\mathbf{W}}\|.\end{aligned}$$

where $\|W\| \leq W_M$, $\|\sigma(\hat{\boldsymbol{x}})\| \leq \sigma_M$, and because $\boldsymbol{z}(t)$ is the state of the first-order filter (9) driven by $\tilde{\boldsymbol{x}}(t)$, its 2-norm satisfies

$$\begin{aligned}\|\boldsymbol{z}(t)\| &= \left\|\int_0^t e^{-\lambda(t-\tau)}\tilde{\boldsymbol{x}}(\tau)\,\mathrm{d}\tau\right\|\\ &\leq \int_0^t e^{-\lambda(t-\tau)}\|\tilde{\boldsymbol{x}}(\tau)\|\,\mathrm{d}\tau\\ &\leq \|\tilde{\boldsymbol{x}}\|_\infty\int_0^t e^{-\lambda(t-\tau)}\,\mathrm{d}\tau\\ &= \frac{1 - e^{-\lambda t}}{\lambda}\|\tilde{\boldsymbol{x}}\|_\infty\\ &\leq \frac{\sqrt{n}}{\lambda}\|\tilde{\boldsymbol{x}}\|_\infty\\ &\leq \frac{\sqrt{n}}{\lambda}\|\tilde{\boldsymbol{x}}\| \quad (\text{since } \|\tilde{\boldsymbol{x}}\|_\infty \leq \|\tilde{\boldsymbol{x}}\|).\end{aligned}$$

with n denoting the state dimension. Then, the inequality becomes:

$$\begin{aligned}\dot{L} \leq &-\frac{1}{2}\lambda_{\min}(\mathbf{Q})|\tilde{\boldsymbol{x}}\|^2 - \|\tilde{\boldsymbol{x}}\|\|\tilde{\mathbf{W}}\|^2 - \lambda\boldsymbol{L}_{\text{int}}\\ &+ \|\tilde{\boldsymbol{x}}\|\|\mathbf{P_1}\|(\|\tilde{\mathbf{W}}\|\sigma_{\mathbf{M}} + \bar{\mathbf{w}})\\ &+ \|\tilde{\boldsymbol{x}}\|W_M\|\tilde{\mathbf{W}}\| + \eta_W\sigma_M\frac{\sqrt{n}}{\lambda}\|\tilde{\boldsymbol{x}}\|\|\mathbf{A}^{-\mathbf{1}}\rho_1^{-1}\|\|\tilde{\mathbf{W}}\|\end{aligned}$$

By completing the squares for the terms involving $\|\tilde{W}\|$, we look for conditions on $\|x\|$ which are independent of the neural network weights error and also make the time derivative of the Lyapunov candidate negative.

$$\begin{aligned}\dot{L} \leq & -\|\tilde{\boldsymbol{x}}\|\|\tilde{\mathbf{W}}\|^2 + k_b\|\tilde{\boldsymbol{x}}\|\|\tilde{\mathbf{W}}\| \\ & -\frac{1}{2}\lambda_{\min}(\mathbf{Q})\|\tilde{\boldsymbol{x}}\|^2 + \|\tilde{\boldsymbol{x}}\|\|\mathbf{P_1}\|w_M - \lambda \boldsymbol{L}_{\text{int}}\end{aligned} \quad (14)$$

where $k_b = \|\mathbf{P_1}\|\sigma_M + W_M + \eta_W \sigma_M \frac{\sqrt{n}}{\lambda}\|\mathbf{A^{-1}}\rho_1^{-1}\|$. The terms involving $\tilde{\mathbf{W}}$ are of the form $-(\|\tilde{\boldsymbol{x}}\|)\|\tilde{\mathbf{W}}\|^2 + (k_b\|\tilde{\boldsymbol{x}}\|)\|\tilde{\mathbf{W}}\|$.
By completing the square, this is bounded above by $\frac{(k_b\|\tilde{\boldsymbol{x}}\|)^2}{4|\tilde{\boldsymbol{x}}\|} = \frac{k_b^2}{4}\|\tilde{\boldsymbol{x}}\|$. The final inequality for $\dot{L}$ is:

$$\dot{L} \leq -\frac{1}{2}\lambda_{min}(\mathbf{Q})\|\tilde{\boldsymbol{x}}\|^2 - \lambda \boldsymbol{L}_{\text{int}} + \left(\|\mathbf{P_1}\|\bar{w} + \frac{k_b^2}{4}\right)\|\tilde{\boldsymbol{x}}\| \quad (15)$$

To find a sufficient condition that guarantees $\dot{L} \leq 0$ and subsequently derive the ultimate bound, we can analyze a simpler upper bound. Since the term $-\lambda L_{int}$ is always non-positive, it can be omitted from the right-hand side while the inequality still holds. The analysis thus proceeds with the remaining terms :

$$\|\tilde{\boldsymbol{x}}\| \geq \frac{2\left(\|\mathbf{P_1}\|\bar{\mathbf{w}} + \mathbf{k_b^2}\right)}{\lambda_{min}(\mathbf{Q})} = b \quad (16)$$

Furthermore, the above condition on $\|\tilde{\boldsymbol{x}}\|$ guarantees the negative semi-definiteness of $\dot{L}$ and therefore, ultimate boundedness of $\tilde{\boldsymbol{x}}$. In fact, $\dot{L}$ is negative definite outside the ball with radius b.

■

2.6 Parameter tuning guidelines

From the estimation error bound (16), several parameter tuning guidelines can be derived. First, k_b is proportional to the learning rates η_1, as well as inversely proportional to the forgetting factor λ. That is,

$$k_b \propto \eta_1, \frac{1}{\lambda}, \frac{1}{\rho_1}$$

Therefore, increasing η_1 increases k_b, which in turn increases the ultimate bound b on the estimation error. Conversely, increasing λ reduces k_b and thus tightens the error bound. This proportionality relationship provides a guideline for selecting parameters to achieve a desired trade-off between convergence speed and steady-state error. Although larger learning rates can lead to faster convergence, care must be taken to avoid overshoot or instability.

The Hurwitz matrix $\mathbf{A}$, which is primarily used for stable integration in the estimator, has a significant effect on both the convergence and accuracy of the state estimation. A more stable (i.e., with eigenvalues further left in the complex plane) matrix $\mathbf{A}$ helps the estimator states track the actual system states with better accuracy. However, a more stable $\mathbf{A}$ may slow down the convergence of the weights, since $\mathbf{A^{-1}}$ appears in the weight update laws. One practical solution is to use a more stable $\mathbf{A}$ for improved estimation accuracy, while compensating for slower weight adaptation by increasing the learning rates.

The following guidelines can be used for parameter selection:
- **Learning Rates** (η_1, η_2): Choose small enough values to ensure stability, but large enough for fast learning. Start with small values (e.g., 0.01) and adjust based on observed convergence speed and overshoot in simulations.
- **Forgetting Factor** (λ): Controls the influence of past errors. Typical values are around 0.1, but should be tuned based on system dynamics and desired responsiveness.
- **Hurwitz Matrix** ($\mathbf{A}$): Select $\mathbf{A}$ to be sufficiently stable for accurate state tracking. If convergence of the weights is too slow, increase the learning rates to compensate.

3. SIMULATION

3.1 Robot Dynamics

To demonstrate the effectiveness of the proposed online neural network identifier, we consider a robot manipulator system as an example. The dynamics of the robot manipulator are described by a nonlinear state-space model, which is suitable for applying the proposed identification method. The dynamics of an n-degree-of-freedom robot manipulator can be described by the following nonlinear state-space model:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{\tau_d} = \boldsymbol{\tau} \quad (17)$$

where $\boldsymbol{q} \in \mathbb{R}^{\ltimes}$ is the joint position vector, $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$ are the joint velocity and acceleration vectors, $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{\ltimes\times\ltimes}$ is the positive definite inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{\ltimes\times\ltimes}$ is the Coriolis/centrifugal matrix, $\boldsymbol{G}(\boldsymbol{q}) \in \mathbb{R}^{\ltimes}$ is the gravity vector, $\boldsymbol{\tau} \in \mathbb{R}^{\ltimes}$ is the control input, and $\boldsymbol{\tau_d}$ is the unknown friction/damping vector. To generalize the system for identification of an unknown nonlinear function, we consider a Hamiltonian structure where the dynamics are partitioned into known and unknown components in state-space form.

$$\begin{bmatrix}\dot{\boldsymbol{q}} \\ \dot{\boldsymbol{p}}\end{bmatrix} = \underbrace{\begin{bmatrix}\mathbf{0} & \mathbf{I} \\ -\mathbf{I} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\frac{\partial H}{\partial p} \\ \frac{\partial H}{\partial q}\end{bmatrix} + \begin{bmatrix}\mathbf{0} \\ \mathbf{I}\end{bmatrix}\boldsymbol{\tau}}_{\text{known}} - \underbrace{\begin{bmatrix}\mathbf{0} \\ \mathbf{I}\end{bmatrix}\boldsymbol{\tau_d}}_{\text{unknown}} \quad (18)$$

where $\mathbf{I} \in \mathbb{R}^{\ltimes\times\ltimes}$ is the identity matrix, $\boldsymbol{p} = \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}}$ is the generalized momentum vector, and $H(\boldsymbol{q}, \boldsymbol{p})$ is the Hamiltonian of the system. The first term on the right-hand side represents the intrinsic system dynamics derived from the Hamiltonian structure, while the second and third terms correspond to the control input and dissipative torque(friction), respectively. And let us define the state vector as $\boldsymbol{x} = [\boldsymbol{q}^T, \boldsymbol{p}^T]^T \in \mathbb{R}^{\not\ltimes\ltimes}$, and the control input as $\boldsymbol{u} = \boldsymbol{\tau} \in \mathbb{R}^{\ltimes}$. Then, the system dynamics can be expressed in the form of (1).

3.2 Figures and tables

All figures and tables should be placed after their first mention in the text. Large figures and tables may span across both columns. Scanned images (e.g., line art, photos) can be used if the output resolution is at least 600 dpi.

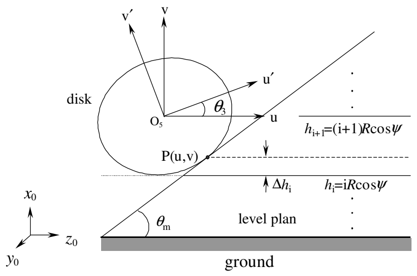

Fig. 1. The caption should be placed after the figure.

Figure captions should be below the figures; table captions should be above the tables. They should be referred to in the text as, for example, Fig. 1, or Figs. 1∼3.

	A	B	C
(1)	150%	16.3%	18.2%

Table 1. Your table caption here

3.3 Equations

Equation numbers should be Arabic numerals enclosed in parentheses on the right-hand margin. They should be cited in the text as, for example, Eq. (1), or Eqs. (1)∼(3). Equations are located in the middle and equation numbers are located at the end. Punctuate equations with commas or periods when they are part of a sentence. For example,

$$\dot{x} = Ax + Bu, \tag{19}$$
$$y = Cx + Du. \tag{20}$$

3.4 References

References should appear in a separate bibliography at the end of the paper, with items referred to by numerals in square brackets [1, 4-5]. Times New Roman 10pt is used for references.

4. PAGE NUMBERS

Do not put a page number in the manuscript PDF.

REFERENCES

[1] J. H. Bong, “Controlling the parasite,” in *Proc. of International Conference on Control, Automation and Systems*, pp. 0209-0210, 2020

[2] A. Alice and B. Bob, “Nonlinear unstable systems,” *International Journal of Control, Automation, and Systems*, vol. 23, no. 4, pp. 123–145, 1989.

[3] M. Young, *The Technical Writer’s Handbook*, Mill Valley, Seoul, 1989.