University of Pavia

Department of Electrical, Computer and Biomedical Engineering

Master's Degree in Industrial Automation Engineering

Deep Neural Network-based Sliding Mode Observers

Supervisor:
Prof. Antonella Ferrara

Candidate:
Ethan Peri

Co-Supervisors:
Nikolas Sacchi, Ph.D.
Edoardo Vacchini, M.Sc.

A.Y. 2025/2026

Abstract

This thesis presents and analyzes a novel observer-based control framework integrating Deep Neural Networks (DNNs) with Sliding Mode Observers (SMOs). This approach aims to achieve accurate approximation of the uncertain dynamics of the system under control in presence of disturbances, dealt with Integral Sliding Mode (ISM) to ensure robustness.

Drawing inspiration from adaptive control principles, the DNNs are trained online via Lyapunov-based adaptation laws enhanced with Barrier Lyapunov Functions (BLFs) to impose prescribed approximation error bound.

The framework is then applied to control scenarios in which the system must satisfy states and input constraints, exploiting DNN-estimates in a Model Predictive Control (MPC).

The DNN-SMO methodology has been theoretically analyzed and its validity verified through numerical simulations on two benchmark examples: a Duffing oscillator and a 3-DoF robotic manipulator, demonstrating practical convergence and constraint satisfaction.

Sommario

Questa tesi presenta e analizza un innovativo framework di controllo basato su osservatori, integrando Reti Neurali Profonde (DNNs) con Osservatori a Modo Scorrevole (SMOs). Tale approccio mira a un'accurata approssimazione della dinamica incerta del sistema sotto controllo in presenza di disturbi, gestiti tramite Integral Sliding Mode (ISM) per garantire robustezza.

Traendo ispirazione dai principi di controllo adattivo, le DNN vengono addestrate online tramite leggi di adattamento basate sull'analisi di Lyapunov, potenziate con Funzioni a Barriera di Lyapunov (BLFs) per imporre vincoli predefiniti sull'errore di stima.

Il framework è poi applicato a sistemi soggetti a vincoli su stati e ingressi, sfruttando le stime delle DNN all'interno di un Controllore Predittivo (MPC).

La metodologia DNN-SMO è stata analizzata teoricamente e validata in simulazione su: un oscillatore Duffing e un manipolatore robotico a 3 giunti, dimostrando convergenza pratica e soddisfazione dei vincoli.

Contents

List of Figures

Acronyms

List of acronyms used in this thesis reported in alphabetic order.

ANN Artificial Neural Network

BLF Barrier Lyapunov Function

DNN Deep Neural Network

DNN-ISM Deep Neural Network-based Integral Sliding Mode

DNN-SMO Deep Neural Network-based Sliding Mode Observer

DoF Degrees of Freedom

EL Euler-Lagrange

FHOCP Finite-Horizon Optimal Control Problem

ISM Integral Sliding Mode

MIMO Multi Input Multi Output

ML Machine Learning

MLP Multi-Layer Perceptron

MPC Model Predictive Control

NAC Neural-Adaptive Control

SMC Sliding Mode Control

SMO Sliding Mode Observer

Introduction

Modern control systems must operate under significant uncertainty, including disturbances, modeling mismatches, and sensor noise. Traditional robust control approaches [1, 2] address these via worst-case designs, often sacrificing performance or increasing actuator stress. Moreover, when system models are entirely unavailable, classical synthesis methods become inapplicable. Over recent decades, the surge in data availability and computational power has driven widespread adoption of Machine Learning (ML), particularly Artificial Neural Networks (ANNs), in engineering, including control theory. Although ML roots trace to early least squares methods, modern ANNs emerged with the Perceptron model [3] and gained prominence via the Universal Approximation Theorem [4], proving that a single-hidden-layer network can approximate any continuous function on compact sets to arbitrary accuracy. Deeper architectures, called Deep Neural Networks (DNNs), further enhance expressiveness [5, 6]. Control theory has embraced ML, positioning DNNs as ideal for data-driven control without parametric models due to their approximation capabilities. However, despite theoretical promise, practical ANN control implementations reveal critical limitations: the main issue is that the training phase is done offline and the approximation error, which is dependent on the network structure and by the quality of the gathered data, is not properly dealt with, preventing an effective control design. Moreover, the fact that the ANNs are trained offline does not give any guarantees regarding approximation fidelity during system transients, when operating conditions deviate significantly from training data distributions. A viable solution to these limitations lies in Neural-Adaptive Control NAC. Works such as [7, 8, 9] demonstrates the effectiveness of online weight adaptation for DNN-based controllers on perturbed nonlinear systems with fully unknown dynamics, even in the presence of state and input constraints. However, online adaptation is a double-edged sword: analysis reveals poor approximation during initial transients, particularly evident with Model Predictive Control (MPC) integration, requiring an auxiliary controller used in place of MPC until DNNs achieve sufficient accuracy.

To address these aspects and enrich the existing literature, this thesis covers the following topics:

- design of a novel DNN-based Sliding Mode Observer (DNN-SMO) framework with online parameter adaptation to estimate perturbed nonlinear system with fully or partially unknown dynamics, built upon SMO-driven neural adaptation principles [10], separating estimation from control;
- integration of Barrier Lyapunov Function (BLF) to explicitly constrain observer sliding variables during transients, ensuring robust estimation performance;
- extension of the aforementioned framework to state and input constrained systems via MPC integration.

The thesis structured into two parts:

(I) **Preliminaries on Sliding Mode and Neural Networks**: Introduces all preliminary concepts behind what is presented. Specifically, Chapter 1 covers Sliding Mode Control Theory fundamentals including Sliding Mode Control (SMC), Integral Sliding Mode (ISM), and Sliding Mode Observer (SMO); Chapter 2 presents Neural Networks foundations, from Deep Neural Network architectures to Universal Approximation Theorem.

(II) **Deep Neural Network-based Sliding Mode Observers**: Details the novel contributions. In particular, Chapter 3 presents the core DNN-SMO architecture; Chapter 4 employs the DNN-SMO strategy to address the control design problem; while Chapter 5 demonstrates the effectiveness of the presented methodology by simulations, including a sensitivity analysis to key design parameters.

Finally, Chapter 6 summarizes the main theoretical contributions, simulation validations, and future research directions.

Part I

Preliminaries on Sliding Mode and Neural Networks

Chapter 1

Preliminaries on Sliding Mode Control Theory

This chapter introduces the main concepts of Sliding Mode Control, Integral Sliding Mode and Sliding Mode Observer, providing the theoretical foundation essential for the DNN-based approach presented later in Chapter 3.

1.1 Sliding Mode Control

In any practical control problem, discrepancies inevitably arise between the actual plant and its mathematical model used for controller design. These mismatches stem from unknown external disturbances, parametric uncertainties, and unmodeled dynamics. Designing control laws that guarantee desired closed-loop performance despite such uncertainties poses a significant challenge for control engineers. This has motivated extensive research into the so-called robust control methodologies which are supposed to solve this problem. Among these, Sliding Mode Control has emerged. Consider a system characterized by a state vector $x \in \mathcal{X} \subset \mathbb{R}^n$, with $\mathcal{X}$ being a compact set containing the origin. Let $\sigma_i : \mathbb{R}^n \to \mathbb{R}$, where $i \in \{1, 2, \dots, \kappa\}$, with $\kappa > 0$, and consider the relative surfaces given by $\sigma_i(x) = 0$, each representing a sub-manifold of the state space. Their intersection defines the so-called sliding manifold (see Figure 1.1). If the system state is forced onto $\sigma_i(x) = 0$, a sliding motion is generated along that switching surface.
The idea behind SMC is to design a controller which, starting from $x(t_0) = x_0 \in \mathcal{X}$, with t_0 assumed to be zero without loss of generality throughout the thesis, steers the state onto the sliding manifold in finite time, maintaining sliding motion on all switching surfaces simultaneously. Once attained, the system enters sliding mode,

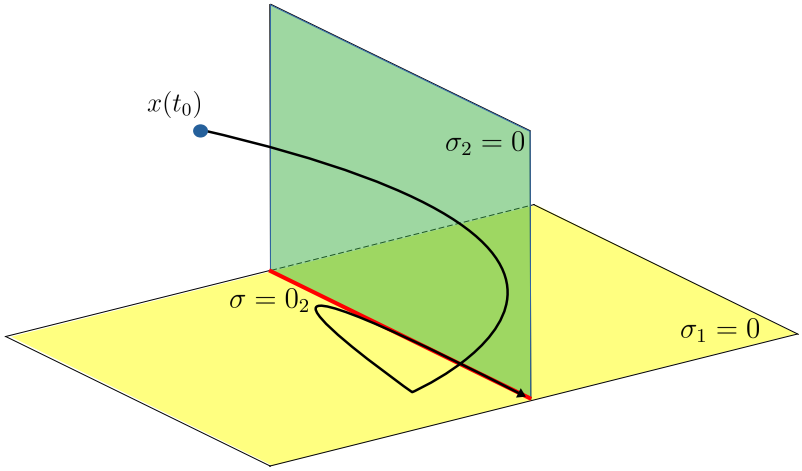

Figure 1.1: Example of the sliding manifold $\sigma = 0_2$ given by the intersection of $\kappa = 2$ switching surfaces.

exhibiting reduced-order dynamics described by the so-called equivalent system, which is implicitly defined by the sliding manifold, becoming insensitive to a significant class of parameter uncertainties and disturbances.

1.1.1 Control-Affine Systems

In classical SMC theory, the focus lies on control-affine systems, which are nonlinear with respect to the state but linear with respect to the control input. As outlined in [11], these systems take the form

$$\dot{x}(t) = f(x(t), t) + B(x(t), t)u(t), \tag{1.1}$$

where $x \in \mathcal{X} \subset \mathbb{R}^n$ denotes the state vector with $\mathcal{X}$ being a compact set containing the origin, $u \in \mathbb{R}^m$ is the control input vector, $t \in \mathbb{R}_{\geq 0}$ is time, $f : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ is the so-called drift dynamics, while $B : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^{n \times m}$ is the so-called control effectiveness matrix. Within the SMC framework, it is common to assume that both f and B are smooth and bounded functions [12, 13].

Assumption 1.1. *The drift dynamics $f(x(t), t)$ is a function of class $C^0(\mathcal{X})$. Moreover, there exist a known constant $\bar{f} \in \mathbb{R}_{>0}$ such that*

$$\sup_{x \in \mathcal{X}} \| f(x(t), t) \| \leq \bar{f},$$

for all $t \in \mathbb{R}_{\geq 0}$.

Assumption 1.2. *The control effectiveness matrix $B(x(t),t)$ is a function of class $C^0(\mathcal{X})$. Moreover, there exist known constants $\underline{b}, \bar{b} \in \mathbb{R}_{>0}$ such that*

$$\underline{b} \leq \|B(x(t),t)\| \leq \bar{b},$$

for all $x \in \mathcal{X}$ and $t \in \mathbb{R}_{\geq 0}$.

1.1.2 Canonical Forms

Analysis of SMC is simplified when the considered nonlinear system is expressed in one of the following canonical forms [11, 14].

Reduced Form Consider a control-affine system as in (1.1). If the control effectiveness matrix has a structure

$$B(x(t),t) = \begin{bmatrix} 0_{(n-m)\times m} \\ \bar{B}(x(t),t) \end{bmatrix},$$

with $\bar{B} : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^{m\times m}$, then the state vector can be partitioned into two components, $x_1 \in \mathcal{X}_1 \subset \mathbb{R}^{n-m}$ and $x_2 \in \mathcal{X}_2 \subset \mathbb{R}^m$, yielding the equivalent dynamics

$$\dot{x}(t) = \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} f_1(x(t),t) \\ f_2(x(t),t) + \bar{B}(x(t),t)u(t) \end{bmatrix}, \tag{1.2}$$

where $f_1 : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^{n-m}$ and $f_2 : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^m$ are the two components of the drift dynamics.

Controllability Form In this form, the control-affine system (1.1) can be decomposed into m subsystems, each of them being a perturbed chain of integrators. Represent the state as $x(t) = \begin{bmatrix} x_1^\top(t) & x_2^\top(t) & \dots & x_m^\top(t) \end{bmatrix}^\top$, where $x_i \in \mathbb{R}^{n_i}$, for $i \in \{1, 2, \dots, m\}$, with $n_i \in \mathbb{N}$ being the dimension of the i-th state vector such that $\sum_{i=1}^m n_i = n$. Each subsystem is characterized by the dynamics

$$\dot{x}_i = A_i x_i(t) + f_i(x(t),t) + b_i(x(t),t)u(t),$$

where $A_i \in \mathbb{R}^{n_i\times n_i}$, $f_i(x(t),t) \in \mathbb{R}^{n_i}$, and $b_i(x(t),t) \in \mathbb{R}^{n_i\times m}$ are defined as

$$f_i(x(t),t) = \begin{bmatrix} 0_{n_i-1} \\ f_{i,0}(x(t),t) \end{bmatrix}, \; b_i(x(t),t) = \begin{bmatrix} 0_{(n_i-1)\times m} \\ b_{i,0}^\top(x(t),t) \end{bmatrix}, \; A_i = \begin{bmatrix} 0_{n_i-1} & I_{n_i-1} \\ 0 & 0_{n_i-1}^\top \end{bmatrix},$$

with $f_{i,0}(x(t),t) \in \mathbb{R}$ and $b_{i,0}(x(t),t) \in \mathbb{R}^m$.

Normal Form Given the following MIMO control-affine system

$$\begin{cases} \dot{x}(t) = f(x(t), t) + B(x(t), t)u(t) \\ \\ y(t) = \psi(x(t), t) \end{cases}$$

where $y \in \mathbb{R}^m$ is the output vector, while $\psi : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^m$ is a sufficiently smooth output map, with relative degree vector $\bar{r} = [r_1 \quad r_2 \quad \cdots \quad r_m]^\top$ and total relative degree $r \leq n$, defined below.

Definition 1.1 (Relative degree.). *Define $r_i \in \mathbb{N}$ as the minimum order of time-derivative of the output element, namely, $y_i^{(r_i)} = \frac{d^r y_i}{dt^{r_i}}$, in which any component of the control vector explicitly appears. The result is the so-called relative degree vector $\bar{r} = [r_1 \quad r_2 \quad \cdots \quad r_m]^\top \in \mathbb{N}^m$ and its 1-norm defines the total relative degree of the system, i.e., $r = \|\bar{r}\|_1 = \sum_{i=1}^m |r_i|$.*

The system can then be decomposed into m subsystems. For the i-th subsystem, a set of so-called external variables $z_{i,j}$ for $(i, j) \in \{1, 2, \ldots, m\} \times \{1, 2, \ldots, r_i - 1\}$ are defined including the output y_i and its derivatives up to order $r_i - 1$. The remaining $n - r$ variables, called internal variables, complete the transformation. Then, the nominal canonical form of the system is

$$\begin{cases} \dot{z}_{i,j} = z_{i,j+1} & \text{if } j \in \{1, 2, \ldots, r_i - 1\} \\ \dot{z}_{i,r_i} = \alpha_i(z, \eta) + \sum_{k=1}^m \beta_{i,k}(z, \eta)\, u_k(t) & \text{if } j = r_i \\ \dot{\eta} = \gamma(z, \eta) \end{cases} \tag{1.3}$$

where $z \in \mathbb{R}^{m \times (r-1)}$ is the matrix collecting the external variables and $\eta \in \mathbb{R}^{n-r}$ is the vector collecting the internal variables.

1.1.3 The Sliding Manifold

As introduced in Section 1.1, the sliding manifold represents the target sub-manifold of the state space toward which the systems states must be steered by the SMC. This section offers a more precise definition, alongside some design examples. For the control-affine system (1.1) and let $\sigma_i : \mathcal{X} \to \mathbb{R}$, with $i \in \{1, 2, \ldots, m\}$, be the sliding variables. Then, for each σ_i, it is possible to define the corresponding sliding surface as the set $\{x \in \mathcal{X} : \sigma_i(x(t)) = 0\}$. Since the sliding manifold is the intersection of all the m sliding surfaces, it is possible to define a new sliding variable $\sigma : \mathcal{X} \to \mathbb{R}^m$ that collects the sliding variables associated with the sliding surfaces, i.e.,

$$\sigma(x(t)) = \Big[\sigma_1(x(t)) \quad \sigma_2(x(t)) \quad \cdots \quad \sigma_m(x(t))\Big]^\top, \tag{1.4}$$

and then define the sliding manifold as $\{x \in \mathcal{X} : \sigma_i(x(t)) = 0, \forall i \in \{1, 2, \ldots, m\}\}$. The sliding manifold may be arbitrary nonlinear functions of the states [15]. Nevertheless, it is common to define it as linear combination of the system states $\{x \in \mathcal{X} : \sigma(x(t)) = Cx(t) = 0\}$, with $C \in \mathbb{R}^{m \times n}$ a suitable design matrix. When the system is in one of the canonical forms, some specific structures for the sliding variable are known, as detailed next.

Reduced Form Given the state partition $x_1 \in \mathcal{X}_1 \subset \mathbb{R}^{n-m}$ and $x_2 \in \mathcal{X}_2 \subset \mathbb{R}^m$, a natural sliding manifold design is the linear combination

$$\sigma(x(t)) = \sigma(x_1(t), x_2(t)) = C_1 x_1(t) + C_2 x_2(t),$$

where $C_1 \in \mathbb{R}^{m \times (n-m)}$ and $C_2 \in \mathbb{R}^{m \times m}$ being non-singular. When the system is in sliding mode, it exhibits the reduced order dynamics

$$\begin{bmatrix} x_2(t) \\ \dot{x}_1(t) \end{bmatrix} = \begin{bmatrix} -C_2^{-1} C_1 x_1(t) \\ f_1 \left(\begin{bmatrix} x_1^\top(t) & -(C_2^{-1} C_1 x_1(t))^\top \end{bmatrix}^\top, t \right) \end{bmatrix},$$

where f_1 is defined in (1.2).

Controllability Form With the system decomposed into m different subsystems, one can design a sliding surface for each subsystem $\sigma_i(x_i(t)) = c_i^\top x_i(t)$, where $c_i \in \mathbb{R}^{n_i}$ is a design vector. On the i-th surface, $\sigma_i(x_i) = 0$, the associated subsystem reduces to

$$\begin{cases} \dot{x}_{i,j} = x_{i,j+1}, & \text{if } j \in \{1, 2, \ldots, n_i - 1\} \\ x_{i,n_i} = -\frac{1}{c_{i,n_i}} \sum_{j=1}^{n_i - 1} c_{i,j} x_{i,j} & \text{if } j = n_i, \end{cases}$$

for $i \in \{1, 2, \ldots, m\}$. Thus, the vectors c_i must be chosen so that the characteristic polynomial defining x_{i,n_i} is Hurwitz.

Normal Form System stability in this representation is related on the so-called zero dynamics, obtained by posing equal to zero the outputs and their derivatives, i.e. the external variables. Hence, it is convenient to design m different sliding variables $\sigma_i : \mathbb{R}^{r-1} \to \mathbb{R}$ as linear combination of the external variables from (1.3) as $\sigma_i(z_i(t)) = c_i^\top z_i$, for all $i \in \{1, 2, \ldots, m\}$, with design vector $c_i \in \mathbb{R}^{r-1}$ chosen as in the controllability form.

1.1.4 The Control Law

The next step involves designing the SMC control law, characterized by high frequency switching behavior. Literature offers different control laws.

Relay Control This controller sets each i-th component of the control input vector to one of two distinct values, based on the sign of the corresponding sliding variable $\sigma_i(x(t))$ from the sliding variable vector $\sigma : \mathcal{X} \to \mathbb{R}^m$ defined as in (1.4), then the control law is

$$u_i(t) = \begin{cases} u_i^+(x(t), t) & \text{if } \sigma_i(x(t)) > 0, \\ u_i^-(x(t), t) & \text{if } \sigma_i(x(t)) < 0, \end{cases}$$

with $i \in \{1, 2, \ldots, m\}$. The values $u_i^+, u_i^- \in \mathbb{R}$ must be designed during the design phase. The standard discontinuous law based on $-K \operatorname{sign}(\sigma)$ is a particular case of the Relay Control Law.

Unit Vector Control Ideal for MIMO system, this method [15] directs control along the unit vector which indicates the direction toward the sliding manifold. Specifically, with sliding variable vector $\sigma : \mathcal{X} \to \mathbb{R}^m$ from (1.4), the control law $u \in \mathbb{R}^m$ is given by

$$u(t) = -K(x(t), t) \frac{\sigma(x(t))}{\|\sigma(x(t))\|}, \tag{1.5}$$

where $K : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^{m \times m}$ is a positive definite design gain matrix.

State Feedback Control with Switching Gain The control law is given by

$$u(t) = \Gamma(x(t))x(t),$$

with state-dependent design gain matrix $\Gamma : \mathcal{X} \to \mathbb{R}^{m \times n}$, whose entries rely on the sliding variable vector $\sigma : \mathcal{X} \to \mathbb{R}^m$, defined as in (1.4). In particular

$$\Gamma_{i,j}(x(t)) = \begin{cases} \gamma_{i,j}^+ & \text{if } \sigma_i(x(t))x_j(t) > 0 \\ \gamma_{i,j}^- & \text{if } \sigma_i(x(t))x_j(t) < 0 \end{cases}$$

for $i \in \{1, 2, \ldots, m\}$ and $j \in \{1, 2, \ldots, n\}$.

1.1.5 Existence and Reaching conditions

A core requirement for SMC is the existence of the sliding motion. Let $t_r \in \mathbb{R}_{\geq 0}$ denote the so-called reaching time, marking when states from some initial condition $x(0) = x_0 \in \mathcal{X}$ reach the sliding manifold. Then, if the condition $\sigma(x(t)) = 0_m$ is satisfied for $t \geq t_r$, the sliding mode is said to be ideal [12]. This allows to distinguish two phases of the system motion: the so-called reaching phase, during which the system states are approaching the manifold, and the sliding phase, in which the

system evolves on the sliding manifold. From a theoretical point of view, the sliding mode existence problem can be treated as a stability problem. In particular, it must be ensured the attractiveness of the sliding manifold for some initial conditions $x(0) = x_0 \in \mathcal{X}$. To prove it, one can rely on Lyapunov's second method [14]. The standard choice for the Lyapunov-like Function to prove the existence of a sliding mode is

$$v(x(t)) = \frac{1}{2}\sigma^\top(x(t))\sigma(x(t)), \tag{1.6}$$

for which the aim is to obtain that the first time-derivative satisfies the so-called reaching condition [15]

$$\dot{v}(x(t)) = \sigma^\top(x(t))\dot{\sigma}(x(t)) < 0.$$

Achieving a sliding mode in a finite time t_r, hinges crucially on satisfying the γ-reaching condition

$$\dot{v}(x(t)) \leq -\gamma\sqrt{2v(x(t))}, \tag{1.7}$$

which translates into

$$\sigma^\top(x(t))\dot{\sigma}(x(t)) \leq -\gamma\|\sigma(x(t))\|, \tag{1.8}$$

with $\gamma \in \mathbb{R}_{>0}$.

Theorem 1.1 (Finite time reaching). *Consider a control-affine system* (1.1) *with initial conditions* $x(0) = x_0 \in \mathcal{X}$ *and sliding variable* $\sigma : \mathcal{X} \to \mathbb{R}^m$. *If the condition* (1.8) *holds, then the reaching time* $t_r \geq 0$ *is bounded above as*

$$t_r \leq \frac{\|\sigma(x_0)\|}{\gamma}.$$

Proof. See Appendix A.1. □

1.1.6 Robustness Property

Another key aspect of SMC is its applications to systems affected by uncertainties and external disturbances. Consider the control-affine system affected by uncertainties given by

$$\dot{x}(t) = f(x(t),t) + \Delta f(x(t),\theta(t),t) + (B(x(t),t) + \Delta B(x(t),\theta(t),t)u(t), \tag{1.9}$$

where $x \in \mathcal{X} \subset \mathbb{R}^n$ and $u \in \mathbb{R}^m$ represent the state and the input of the system, $f : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ is the nominal drift dynamics, and $B : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^{n\times m}$ is the nominal control effectiveness matrix. As for $\Delta f : \mathcal{X} \times \Theta \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ and $\Delta B : \mathcal{X} \times \Theta \times \mathbb{R}_{\geq 0} \to \mathbb{R}^{n\times m}$, representing the model uncertainties dependent on a

time-varying unknown parameter vector $\theta \in \Theta \subset \mathbb{R}^p$, with Θ being a compact set, and satisfying the so-called matching condition.

Definition 1.2 (Matching condition). *A vector $v \in \mathbb{R}^n$ is said to be matched if $v \in \text{span}\{B(x(t),t)\}$, for all $x \in \mathcal{X}$ and $t \in \mathbb{R}_{\geq 0}$.*

The uncertainties can be then rewritten as

$$\Delta f(x(t),\theta(t),t) + \Delta B(x(t),\theta(t),t) = B(x(t),t)w(x(t),\theta(t),t),$$

with $w \in \mathbb{R}^m$ being the vector of matched uncertainties, then, it is possible to rewrite (1.9) as if the uncertainties act as a disturbance acting on the input channel

$$\dot{x}(t) = f(x(t),t) + B(x(t),t)\left(u(t) + w(x(t),\theta(t),t)\right). \tag{1.10}$$

Moreover, the perturbation vector $w(x(t),\theta(t),t)$ satisfies the following assumption.

Assumption 1.3 (Bounded perturbation). *There exists a known scalar function $\kappa : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}$ such that the norm of the perturbation vector $w(x(t),\theta(t),t)$ is bounded above as $\|w(x(t),\theta(t),t)\| \leq \kappa(x(t),t)$, for all $x \in \mathcal{X}$ and $t \in \mathbb{R}_{\geq 0}$.*

It has been proven that, when the states lie on the sliding manifold, the system is robust to disturbances that enters the system in the same channel of the input [16].

1.2 Integral Sliding Mode

The order reduction and robustness against matched perturbations hold only during sliding mode, leaving the system vulnerable to disturbances in the reaching phase. One remedy involves high-gain switching control to shorten t_r, though this increases the chattering phenomena risking actuators damage [12, 17, 18]. In [19], Utkin and Shi address this problem by introducing the Integral Sliding Mode (ISM), proposing a modified sliding variable that removes the reaching phase entirely. This ensures robust dynamics across the full state space, viewed as the sliding manifold. Consider the nonlinear system

$$\dot{x}(t) = f(x(t),t) + B(x(t),t)u(t) + h(x(t),t) \tag{1.11}$$

where $x \in \mathcal{X} \subset \mathbb{R}^n$, with $\mathcal{X}$ being a compact set containing the origin, represent the system states, $u \in \mathbb{R}^m$ is the control input vector, $f : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ is the drift dynamics, $B : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^{n\times m}$ is the control effectiveness matrix, while $h : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ is the vector of disturbances, bounded according to the following assumption

Assumption 1.4. *There exists a known positive scalar function* $\bar{h} : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{>0}$ *that bounds the disturbance term as*

$$h(x(t), t) \in \mathcal{H}, \quad \mathcal{H} \coloneqq \left\{ v(x(t), t) \in \mathbb{R}^n : \|v(x(t), t)\| \leq \bar{h}(x(t), t) \right\},$$

for all $x \in \mathcal{X}$ *and* $t \in \mathbb{R}_{\geq 0}$.

In the ISM control framework, the control law is designed as follows

$$u(t) = u_n(t) + u_r(t), \tag{1.12}$$

where $u_r \in \mathbb{R}^m$ is a switching controller designed to ensure robustness against h, while $u_n \in \mathbb{R}^m$ is the nominal controller, designed to stabilize the nominal system (1.11) in the absence of h. The integral sliding variable $\sigma : \mathcal{X} \to \mathbb{R}^m$ is then defined as the sum of two components

$$\sigma(x(t)) = \sigma_0(x(t)) - z(x(t)), \tag{1.13}$$

where $\sigma_0 : \mathcal{X} \to \mathbb{R}^m$ is the conventional SMC sliding variable, chosen, for example, according to the methods outlined in Section 1.1.3 and fulfilling the following assumption.

Assumption 1.5. *The conventional sliding variable* $\sigma_0 : \mathcal{X} \to \mathbb{R}^m$ *is chosen so that the matrix*

$$\frac{\partial \sigma_0(x(t))}{\partial x} B(x(t), t) \in \mathbb{R}^{m \times m},$$

with $\frac{\partial \sigma_0(x(t))}{\partial x} \in \mathbb{R}^{m \times n}$ *being the Jacobian of* σ_0 *with respect to* x*, is positive definite for all* $x \in \mathcal{X}$ *and* $t \in \mathbb{R}_{\geq 0}$.

As for $z :\in \mathcal{X} \to \mathbb{R}^m$, it is the so-called transient function and it is designed on the system controlled by the nominal controller as

$$z(x(t)) = \sigma_0(x_0) + \int_0^t \frac{\partial \sigma_0(x(\tau))}{\partial x} \left(f(x(\tau), \tau) + B(x(\tau), \tau) u_n(\tau) \right) d\tau.$$

This design guarantees that at $t = 0$ it holds $\sigma(x(0)) = \sigma_0(x_0) - z(x_0) = \sigma_0(x_0) - \sigma_0(x_0) = 0_m$. Placing the system on the sliding manifold from the initial time instant.

1.2.1 Existence conditions

The following theorem presents the existence conditions of an ISM for $t \geq 0$.

Theorem 1.2 (ISM Existence). *Given the nonlinear perturbed multi-input system* (1.11) *with matched uncertainty* $h(x(t),t)$ *satisfying Assumption 1.4, integral sliding variable* $\sigma(x(t))$ *defined as in* (1.13)*, and controlled via ISM controller* $u(t)$ *in* (1.12)*, with discontinuous controller* $u_r(t)$ *chosen according to the unit vector approach in* (1.5)*, i.e.,*

$$u_r(t) = -\rho(x(t),t)\frac{\sigma(x(t),t)}{\|\sigma(x(t),t)\|}.$$

Then, if the control gain $\rho : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{>0}$ *is chosen so that it satisfies*

$$\rho(x(t),t) > \frac{\|\frac{\partial \sigma_0(x(\tau))}{\partial x}\|\bar{h}(x(t),t) + \bar{\eta}}{\underline{\lambda}\left(\frac{\partial \sigma_0(x(\tau))}{\partial x}B(x(t),t)\right)},$$

where $\bar{h}(x(t),t)$ *is the function introduced in Assumption 1.4,* $\bar{\eta} \in \mathbb{R}_{>0}$ *is a design constant, while* $\lambda(\cdot)$ *denotes the smallest in norm eigenvalue of its argument, then a sliding mode* $\sigma(x(t)) = 0_m$ *is enforced for* $t \geq 0$*.*

Proof. The result follows straight-forwardly from Theorem 1.1. □

1.3 Sliding Mode Observers

The aim of an observer is to reconstruct unmeasurable states of a system using only measured inputs and outputs, mimicking the system dynamics. It is essentially a mathematical replica of the system, driven by the system input together with a correction signal based on the output mismatch between plant and model. The classic Luenberger observer design applies linear feedback of this error [20]. However, under model uncertainties or unknown disturbances, it typically fails to drive output estimation error to zero and prevents observer states from tracking true states. Sliding Mode Observers (SMOs) address these problems by applying SMC principles, providing an attractive solution by replacing the linear feedback with a discontinuous injection term based on the sign function of the output error $y - \hat{y}$. Moreover, under bounded disturbances (Assumption 1.4), SMOs guarantee finite-time convergence of the estimation error to zero, with observer states converging asymptotically to the system states [13]. Consider the MIMO system

$$\begin{cases} \dot{x}(t) = f(x(t),t) + B(x(t),t)u(t) + h(x(t),t) \\ \\ y(t) = \psi(x(t),t) \end{cases} \tag{1.14}$$

where $x \in \mathcal{X} \subset \mathbb{R}^n$, with $\mathcal{X}$ being a compact set containing the origin, represent the system states, $u \in \mathbb{R}^m$ is the control input vector, $f : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ is the drift dynamics, $B : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^{n \times m}$ is the control effectiveness matrix, $h : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ is the vector disturbances, while $y \in \mathbb{R}^p$ is the output vector, with $\psi : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^p$ be a smooth vector field satisfying the following assumption.

Assumption 1.6. *Given the observable system in* (1.14)*, the output dynamics* $\psi(x(t), t)$ *is a function of class of class* $C^1(\mathcal{X})$*. Moreover, there exist a known constant* $\bar{\psi} \in \mathbb{R}_{>0}$ *such that*

$$\sup_{x \in \mathcal{X}} \left\| \frac{\partial \psi(x(t), t)}{\partial x} \right\| \leq \bar{\psi},$$

for all $t \in \mathbb{R}_{\geq 0}$*, with* $\frac{\partial \psi}{\partial x} \in \mathbb{R}^{p \times n}$ *being the Jacobian of* ψ *with respect to* x*.*

The corresponding SMO takes the form

$$\begin{cases} \dot{\hat{x}}(t) = f(\hat{x}(t), t) + B(\hat{x}(t), t)u(t) + l(y(t), \hat{y}(t)) \\ \\ \hat{y}(t) = \psi(\hat{x}(t), t) \end{cases} \tag{1.15}$$

where $\hat{x} \in \mathcal{X} \subset \mathbb{R}^n$ is the estimated state vector and $l : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^n$ is the correction term aimed to steer the estimated states toward the real ones, given by

$$l(y(t), \hat{y}(t)) = \hat{\rho} \frac{\hat{\sigma}(y(t), \hat{y}(t))}{\|\hat{\sigma}(y(t), \hat{y}(t))\|}, \tag{1.16}$$

with $\hat{\rho} \in \mathbb{R}_{>0}$ being the observer gain and observer sliding variable $\hat{\sigma} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^n$ defined as

$$\hat{\sigma}(x(t), \hat{x}(t)) = \hat{C}(y(t) - \hat{y}(t)) = \hat{C}(\psi(x(t), t) - \psi(\hat{x}(t), t)),$$

where $\hat{C} \in \mathbb{R}^{n \times p}$ is a design matrix chosen according to the next assumption.

Assumption 1.7. *The matrix* $\hat{C} \in \mathbb{R}^{n \times p}$ *is chosen so that the matrix*

$$\hat{C} \frac{\partial \psi(x(t), t)}{\partial x} \in \mathbb{R}^{n \times n},$$

with $\frac{\partial \psi}{\partial x} \in \mathbb{R}^{p \times n}$ *being the Jacobian of* ψ *with respect to* x*, is positive definite for all* $x \in \mathcal{X}$ *and* $t \in \mathbb{R}_{\geq 0}$*.*

Consider the plant-model mismatch

$$\begin{aligned} \Delta(x(t), \hat{x}(t), t) = & \frac{\partial \psi(x(t), t)}{\partial x}(f(x(t), t) + B(x(t), t)u(t) + h(x(t), t)) + \\ & - \frac{\partial \psi(\hat{x}(t), t)}{\partial \hat{x}}(f(\hat{x}(t), t) + B(\hat{x}(t), t)u(t)), \end{aligned}$$

to be compensated by the correction term the following assumption is required.

Assumption 1.8. *Let Assumptions 1.1, 1.2, 1.4 and 1.6 hold, then, there exists a known constant* $\bar{\Delta} \in \mathbb{R}_{>0}$ *such that*

$$\sup_{x,\hat{x}\in\mathcal{X}} \|\Delta(x(t),\hat{x}(t),t)\| \leq \bar{\Delta},$$

for all $t \in \mathbb{R}_{\geq 0}$.

1.3.1 Existence conditions

The following theorem presents the existence conditions of a SMO for $t \geq 0$.

Theorem 1.3. *Consider the system in* (1.14) *and the SMO in* (1.15). *If Assumption 1.6-1.8 hold, and*

$$\hat{\rho} > \frac{\|\hat{C}\|\bar{\Delta} + \bar{\eta}}{\underline{\lambda}\left(\hat{C}\frac{\partial\psi}{\partial\hat{x}}\right)}, \tag{1.17}$$

with $\bar{\eta} \in \mathbb{R}_{>0}$ *being a design parameter, then,* $\hat{\sigma}(y(t),\hat{y}(t)) \to 0_n$ *for* $t \to \infty$. *Moreover, since the system is observable the estimated states converge asymptotically to the true states, i.e.,* $\hat{x}(t) \to x(t)$ *for* $t \to \infty$.

Proof. See Appendix A.2. □

Chapter 2

Preliminaries on Neural Networks

This chapter introduces the foundational concepts of Artificial Neural Networks and Machine Learning essential for the development of the strategies presented in this thesis. In particular, it covers the mathematical structure of the Multi-Layer Perceptron (MLP) and the universal approximation theorem.

2.1 Multi-Layer Perceptron

MLP belongs to the class of Artificial Neural Networks (ANNs). These models were originally designed to replicate the operations and information-processing abilities of the nervous system, which functions as a highly interconnected web of neurons. A biological neuron, as shown in Figure 2.1, consists of three main parts: the soma, the dendrites, and the axon. Dendrites capture incoming signals from neighboring neurons, whereas the axon, linked to the dendrites of other cells, carries outgoing signals from the neuron. The behavior of a neuron can be summarized as follows: dendrites adjust signals received from other neurons using weighting factors before passing them to the soma that form an aggregated signal. Once this exceeds a specific threshold, it travels along the axon. Notably, cellular nonlinearities make the soma's output signal a nonlinear function of the incoming weighted inputs.

2.1.1 The Perceptron model

Inspired from the biological neuron dynamics, the so-called perceptron model, proposed in [3] and illustrated in Figure 2.2, is now described. Consider an input vector $x = [x_1 \quad x_2 \quad \cdots \quad x_n]^\top \in \mathbb{R}^n$, a weight vector $w = [w_1 \quad w_2 \quad \cdots \quad w_n]^\top \in \mathbb{R}^n$, a

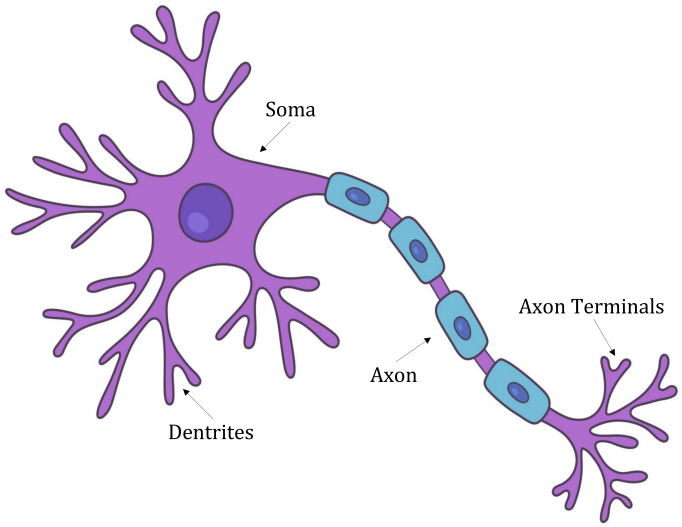

Figure 2.1: Graphical representation of a biological neuron.

bias term $b \in \mathbb{R}$, and an activation function $\varsigma : \mathbb{R} \to \mathbb{R}$. The neuron's output $y \in \mathbb{R}$ is then computed as

$$y = \varsigma(w^\top x + b) = \varsigma(b + \sum_{i=1}^{n} x_i w_i).$$

By extending the input vector to include the weights and the bias in a single vector,

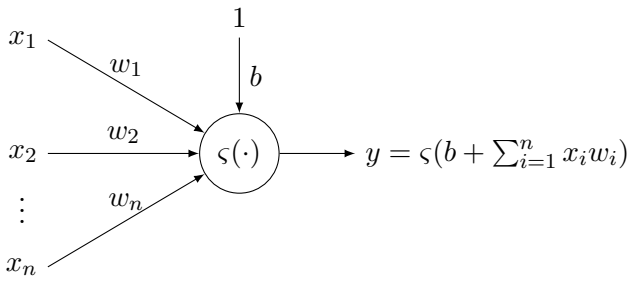

Figure 2.2: Mathematical model of the neuron.

augmented vectors $x_h \in \mathbb{R}^{n+1}$ and $v \in \mathbb{R}^{n+1}$ emerge

$$x_h = \begin{bmatrix} x^\top & 1 \end{bmatrix}^\top, \quad v = \begin{bmatrix} w^\top & b \end{bmatrix}. \tag{2.1}$$

This yields a streamlined output expression $y = \varsigma(v^\top x)$. While a lone perceptron suffices for basic tasks [3], tackling greater complexity demands multiple neurons arranged in layers-sets of neurons that share a common input but lack internal connections, with their collective outputs feeding the next layer. This architecture define the Multi-Layer Perceptron (MLP) [4], with an example shown in Figure 2.3.

Their mathematical formulation is now provided. A MLP features an input layer

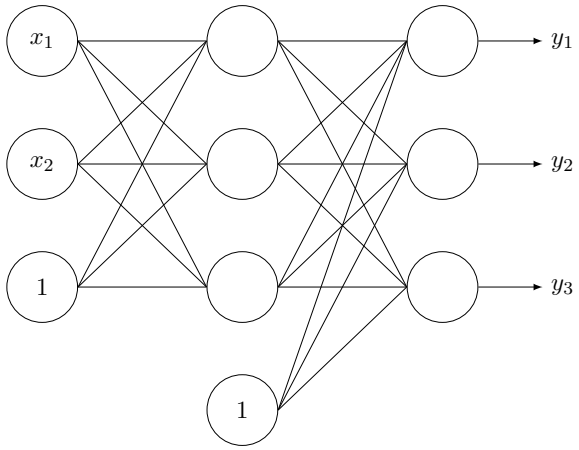

Figure 2.3: Example of a MLP with one hidden layer.

followed by $k \in \mathbb{N}_{\geq 1}$ hidden layers and final output layer. Each j-th layer employs $L_j - 1$ activation functions alongside a linear identity mapping to incorporate bias terms for the subsequent layer, where, $L_0 - 1$ and $L_{k+1} - 1$ represent the respective input and output dimensions. Moreover, each layer includes a weight matrix $W_j \in \mathbb{R}^{(L_j-1)\times(L_{j+1}-1)}$ and a bias vector $b_j \in \mathbb{R}^{L_{j+1}-1}$, structured as follows

$$W_j^\top \coloneqq \begin{bmatrix} w_{j,1}^\top \\ w_{j,2}^\top \\ \vdots \\ w_{j,L_{j+1}-1}^\top \end{bmatrix}, \quad b_j \coloneqq \begin{bmatrix} b_{j,1} & b_{j,2} & \cdots & b_{j,L_{j+1}-1} \end{bmatrix},$$

where $w_{j,i} \in \mathbb{R}^{L_j-1}$ and $b_{j,i} \in \mathbb{R}$ denote, respectively, the weight vector and bias for the i-th neuron in the j-th layer, with $i \in \{1, 2, \ldots, L_{j+1} - 1\}$. Thus, the output of the j-th layer can be expressed through the vector $\Phi_j \in \mathbb{R}^{L_{j+1}-1}$, given by

$$\Phi_j \coloneqq \begin{cases} W_j^\top \varsigma_j(\Phi_j) + b_j & \text{for } j \in \{1, 2, \ldots, k\} \\ W_0^\top x + b_0 & \text{for } j = 0, \end{cases} \tag{2.2}$$

where $\varsigma_j : \mathbb{R}^{L_j-1} \to \mathbb{R}^{L_j-1}$ denotes the vector of the activation functions of the j-th layer, expressed as

$$\varsigma_j(\Phi_{j-1}) = \begin{bmatrix} \varsigma_{j,1}(\Phi_{j-1,1}) & \varsigma_{j,2}(\Phi_{j-1,2}) & \cdots & \varsigma_{j,L_j-1}(\Phi_{j-1,L_j-1}) \end{bmatrix},$$

with $\varsigma_{j,i} : \mathbb{R} \to \mathbb{R}$ representing the activation function applied to the i-th neuron in the j-th layer, and $\Phi_{j-1,i} \in \mathbb{R}$ indicating the i-th element of the input vector $\Phi_{j-1} \in$

$\mathbb{R}^{L_j-1}$. Building an augmented state vector x_h from (2.1), the ANN formulation simplifies by defining, for each layer j, the function vector $\phi_j : \mathbb{R}^{L_j} \to \mathbb{R}^{L_j}$ and the matrix $V_j \in \mathbb{R}^{L_j \times L_{j+1}}$. Specifically, for any input $\alpha \in \mathbb{R}^{L_j}$, the function becomes

$$\phi(\alpha) \coloneqq \begin{bmatrix} \varsigma_j(\alpha_{[1:L_j-1]}) \\ \mathrm{id}(\alpha_{L_j}) \end{bmatrix}, \tag{2.3}$$

where $\mathrm{id} : \mathbb{R} \to \mathbb{R}$ is an identity function $\mathrm{id}(\alpha) = \alpha, \forall \alpha \in \mathbb{R}$. The matrix V_j integrates both the weight matrix W_j and the bias vector b_j as

$$V_j^\top \coloneqq \begin{bmatrix} W_j^\top & b_j \\ 0_{L_j-1}^\top & 1 \end{bmatrix},$$

for layers $j \in \{0, 1, \ldots, k-1\}$. For the final layer $j = k$, V_k excludes bias augmentation since no further bias term is needed. In fact, this last one is defined as

$$V_k^\top \coloneqq \begin{bmatrix} W_k^\top & b_k \end{bmatrix} \in \mathbb{R}^{(L_{k+1}-1)\times L_k},$$

with $L_{k+1} - 1$ being the number of outputs of the ANN. This allows recasting the j-th layer output, expressed originally as in (2.2), more compactly as

$$\Phi_j = \begin{cases} V_j^\top \phi_j(\Phi_{j-1}) & \text{for } j \in \{1, 2, \ldots, k\} \\ V_0^\top x_h & \text{for } j = 0, \end{cases} \tag{2.4}$$

where $\phi_j(\Phi_{j-1})$ is computes as in (2.3) with $\alpha = \Phi_{j-1}$. Note that, now $\Phi_j \in \mathbb{R}^{L_{j+1}}$.

2.2 Universal approximation capabilities of ANNs

A primary factor behind the broad adoption of ANNs stems from their ability to approximate a wide class of functions. Known as the universal approximation property, this feature has been rigorously analyzed in key works such as [4, 6, 21, 22] and is outlined below.

Theorem 2.1 (Universal Approximation [21]). *Let $f : \Omega \to \mathbb{R}^p$ be a function of class $C^0(\Omega)$ defined over a compact set $\Omega \subset \mathbb{R}^n$ and $\Phi : \Omega \to \mathbb{R}^p$ an* ANN *with $k \in \mathbb{N}_{\geq 1}$ hidden layers whose output is computed as in* (2.4)*. Then, if the activation functions in the vector ς_j, with $j \in \{0, 1, \ldots, k\}$ are not polynomial, it holds that*

$$f(x) = \Phi(x) + \varepsilon_\Phi(x), \tag{2.5}$$

where $\varepsilon_\Phi : \Omega \to \mathbb{R}^p$ being the approximation error. Moreover, there exists a constant $\bar{\varepsilon}_\Phi \in \mathbb{R}_{>0}$ such that

$$\sup_{x\in\Omega} \|\varepsilon_\Phi(x)\| \leq \bar{\varepsilon}_\Phi.$$

Note that $\Phi(x)$ in (2.5) corresponds to the final layer's output of the DNN. This result extends to matrix case as well. Let $B : \Omega \to \mathbb{R}^{p\times q}$ be a continuous function, and $\Phi : \Omega \to \mathbb{R}^{pq}$. Then, it holds that

$$\mathrm{vec}(B(x)) = \Phi(x) + \varepsilon_\Phi(x), \tag{2.6}$$

with $\varepsilon_\Phi : \Omega \to \mathbb{R}^{pq}$ and $\mathrm{vec}(\cdot)$ the vectorization operator defined as follows.

Definition 2.1 (Vectorization). *Given a matrix $A \in \mathbb{R}^{p\times q}$ and having $A^{(i)} \in \mathbb{R}^p$, with $i \in \{1, 2, \ldots, q\}$, denoting its i-th column, then $\mathrm{vec}(A) \in \mathbb{R}^{pq}$ is a column vector defined as*

$$\mathrm{vec}(A) = \begin{bmatrix} A^{(1)} \\ A^{(2)} \\ \vdots \\ A^{(q)} \end{bmatrix}.$$

Moreover, if $a \in \mathbb{R}^n$ is a vector, then $a \equiv \mathrm{vec}(a)$.

The expression in (2.6) can be equivalently expressed as

$$B(x) = \mathrm{vec}^{-1}(\Phi(x) + \varepsilon_\Phi(x)), \tag{2.7}$$

with $\mathrm{vec}^{-1}(\cdot)$ denotes the inverse of the vectorization operator. Before to detailing this last one, the concept of the Kronecker product must first be introduced.

Definition 2.2 (Kronecker product). *Given two matrices $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{p\times q}$, the Kronecker product, denoted with $A \otimes B$, is defined as*

$$A \otimes B = \begin{bmatrix} a_{1,1}B & a_{1,2}B & \cdots & a_{1,m-1}B & a_{1,m}B \\ a_{2,1}B & a_{2,2}B & \cdots & a_{2,m-1}B & a_{2,m}B \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{n-1,1}B & a_{n-1,2}B & \cdots & a_{n-1,m-1}B & a_{n-1,m}B \\ a_{n,1}B & a_{n,2}B & \cdots & a_{n,m-1}B & a_{n,m}B \end{bmatrix} \in \mathbb{R}^{np\times mq}$$

Definition 2.3 (Vectorization inverse). *Given a vector $a \in \mathbb{R}^{pq}$, then, $\mathrm{vec}^{-1}(a) \in \mathbb{R}^{p\times q}$ is the matrix obtained performing the inverse of the vectorization operation, defined as*

$$\mathrm{vec}^{-1}(a) = (\mathrm{vec}(I_q)^\top \otimes I_p)(I_p \otimes a) \in \mathbb{R}^{p\times q}.$$

Moreover, it holds that $\mathrm{vec}(\mathrm{vec}^{-1}(a)) = a$.

Leveraging the representation in (2.7), the estimate for any specific column $B^{(i)}(x) \in \mathbb{R}^p$ of $B(x)$ takes the form

$$B^{(i)}(x) = \text{vec}^{-1}(\Phi(x))^{(i)} + \text{vec}^{-1}(\varepsilon_\Phi(x))^{(i)},$$

for $i \in \{1, 2, \ldots, q\}$. Recalling that $V_k \in \mathbb{R}^{(L_k-1)\times(L_{k+1}-1)}$, with $L_{k+1} - 1 = pq$ matches the output size of Φ, V_k can be decomposed into different sub-matrices $V_k^{[i]} \in \mathbb{R}^{L_k-1} \times p$

$$V_k = \begin{bmatrix} V_k^{[1]} & V_k^{[2]} & \cdots & V_k^{[q]} \end{bmatrix}.$$

Consequently, the expression of $\text{vec}^{-1}(\Phi(x))^{(i)} \in \mathbb{R}^p$ is given by

$$\text{vec}^{-1}(\Phi(x))^{(i)} = \text{vec}^{-1}(\Phi_k)^{(i)} = V_k^{[i]}\phi_k(\Phi_{k-1}).$$

2.3 Approximating the Dynamics using DNNs

A key design choice for ANNs involves selecting the number of hidden layers k, called depth, and the neuron count per layer, known as width. Over recent decades, research has shown that adding layers yields greater approximation gains than expanding width alone [5, 23]. From Theorem 2.1, approximation precision scales with neuron count. In shallow network, a ANN with $k = 1$, neuron numbers dictate slope variations linearly, achieving polynomial growth in this capacity with width, whereas, in DNNs, ANN with $k \in \mathbb{N}_{\geq 2}$ exhibits exponential gains tied to depth. Thus, DNNs can approximate arbitrary continuous functions, making them ideal for capturing complex nonlinear dynamics such as those of the control-affine system in (1.1). In this case, the dynamics is modeled by two ideal DNNs $\Phi : \mathcal{X} \to \mathbb{R}^n$ and $\Psi : \mathcal{X} \to \mathbb{R}^{nm}$ as

$$f(x(t), t) = \Phi(x(t)) + \varepsilon_\Phi(x(t)), \tag{2.8a}$$

$$B(x(t), t) = \text{vec}^{-1}\left(\Psi(x(t))\right) + \varepsilon_\Psi(x(t)), \tag{2.8b}$$

where $\varepsilon_\Phi : \mathcal{X} \to \mathbb{R}^n$ and $\varepsilon_\Psi : \mathcal{X} \to \mathbb{R}^{n\times m}$ are the approximation errors. The DNN Φ is characterized by $k_\Phi \in \mathbb{N}_{\geq 2}$ hidden layers and each layer $j \in \{0, 1, \ldots, k_\Phi\}$ contains L_{Φ_j} neurons and is activated using a nonlinear function $\phi_j : \mathbb{R}^{L_{\Phi_j}} \to \mathbb{R}^{L_{\Phi_j}}$. Similarly, Ψ is characterized by $k_\Psi \in \mathbb{N}_{\geq 2}$ hidden layers and each layer $j \in \{0, 1, \ldots, k_\Psi\}$ contains L_{Ψ_j} neurons and is activated using a nonlinear function $\psi_j : \mathbb{R}^{L_{\Psi_j}} \to \mathbb{R}^{L_{\Psi_j}}$. The following regularity condition on the activation functions is required.

Assumption 2.1. *The activation functions $\phi_j : \mathbb{R}^{L_{\Phi_j}} \to \mathbb{R}^{L_{\Phi_j}}$ and $\psi_p : \mathbb{R}^{L_{\Psi_p}} \to \mathbb{R}^{L_{\Psi_p}}$, with $j \in \{0, 1, \ldots, k_\Phi\}$ and $p \in \{0, 1, \ldots, k_\Psi\}$, are of class C^1 and Lipschitz continuous.*

Common choices fulfilling this assumption include the sigmoid and the hyperbolic tangent, which are widely adopted in nonlinear system identification. It is possible to express the output of each layer of Φ and Ψ in compact form (2.4). In particular, for the first DNN, one has

$$\Phi_j = \begin{cases} V_j^\top \phi_j(\Phi_{j-1}) & \text{for } j \in \{1, 2, \dots, k_\Phi\} \\ V_0^\top x_h & \text{for } j = 0, \end{cases}$$

where $V_j \in \mathbb{R}^{L_{\Phi_j} \times L_{\Phi_{j+1}}}$ is the matrix that contains ideal weights and biases associated with the j-th layer and $x_h = [x^\top \quad 1]^\top \in \mathbb{R}^{n+1}$. As for the DNN Ψ a similar expression can be derived

$$\Psi_j = \begin{cases} U_j^\top \psi_j(\Psi_{j-1}) & \text{for } j \in \{1, 2, \dots, k_\Psi\} \\ U_0^\top x_h & \text{for } j = 0, \end{cases}$$

where, similarly to the previous case, $U_j \in \mathbb{R}^{L_{\Psi_j} \times L_{\Psi_{j+1}}}$ is the matrix containing ideal weights and biases associated with the j-th layer of the DNN. Moreover, since the overall output of a DNN coincides with the output of the last layer, it holds that $\Phi(x(t)) \equiv \Phi_{k_\Phi}$ and $\Psi(x(t)) \equiv \Psi_{k_\Psi}$, so the representation in (2.8) can be rewritten

$$\begin{aligned} f(x(t), t) &= \Phi_{k_\Phi} + \varepsilon_\Phi(x(t)), \\ B(x(t), t) &= \operatorname{vec}^{-1}(\Psi_{k_\Psi}) + \varepsilon_\Psi(x(t)). \end{aligned}$$

Then, it is convenient to provide an expression for f and of each i-th column of B, with $i \in \{1, 2, \dots, m\}$. In particular, the matrix $U_{k_\Psi} \in \mathbb{R}^{L_{k_\Psi} \times nm}$ can be seen as the composition of different sub-matrices, i.e.,

$$U_{k_\Psi} = \begin{bmatrix} U_{k_\Psi}^{[1]} & U_{k_\Psi}^{[2]} & \dots & U_{k_\Psi}^{[m]} \end{bmatrix},$$

where $U_{k_\Psi}^{[i]} \in \mathbb{R}^{L_{k_\Psi} \times n}$. Then, one has that

$$f(x(t), t) = (V_{k_\Phi})^\top \phi_{k_\Phi}(\Phi_{k_\Phi - 1}) + \varepsilon_\Phi(x(t)) = \Phi_{k_\Phi} + \varepsilon_\Phi(x(t)), \tag{2.9a}$$

$$B^{(i)}(x(t), t) = \left(U_{k_\Psi}^{[i]}\right)^\top \psi_{k_\Psi}(\Psi_{k_\Psi - 1}) + \varepsilon_\Psi^{(i)}(x(t)) = \Psi_{k_\Psi}^{[i]} + \varepsilon_\Psi^{(i)}(x(t)), \tag{2.9b}$$

where $\varepsilon_\Psi^{(i)} : \mathcal{X} \to \mathbb{R}^n$ is the i-th column of ε_Ψ. By virtue of the universal approximation property of Φ and Ψ and the boundedness of f and B (see Assumption 1.1 and 1.2), the following assumption about ideal DNNs can be introduced.

Assumption 2.2. *There exist some known constants $\bar{V}, \bar{U}, \bar{\varepsilon}_{\Phi_1}, \bar{\varepsilon}_{\Phi_2}, \bar{\varepsilon}_\Psi \in \mathbb{R}_{>0}$ such that*

$$\begin{aligned} \sup_{j \in \{0,1,\dots,k_\Phi\}} \|V_j\| \leq \bar{V}, \quad \sup_{x \in \mathcal{X}} \|\varepsilon_\Phi(x(t))\| \leq \bar{\varepsilon}_\Phi, \\ \sup_{j \in \{0,1,\dots,k_\Psi\}} \|U_j\| \leq \bar{U}, \quad \sup_{x \in \mathcal{X}} \|\varepsilon_\Psi(x(t))\| \leq \bar{\varepsilon}_\Psi. \end{aligned}$$

Note that the knowledge of the matrices V_j and U_j, is not available. Hence, the ideal estimation in (2.8) cannot be computed. For this reason, an additional pair of DNNs, namely $\hat{\Phi} : \mathcal{X} \to \mathbb{R}^n$ and $\hat{\Psi} : \mathcal{X} \to \mathbb{R}^{nm}$, is introduced, sharing the same structure and activation functions of the ideal ones, but characterized by an approximation of the ideal weights and biases. The resulting approximations of the drift term and of the i-th column of the control effectiveness matrix are given by

$$\hat{f}(x(t),t) = \left(\hat{V}_{k_\Phi}\right)^\top \phi_{k_\Phi}\left(\hat{\Phi}_{k_\Phi - 1}\right) = \hat{\Phi}_{k_\Phi}, \tag{2.10a}$$

$$\hat{B}^{(i)}(x(t),t) = \left(\hat{U}^{[i]}_{k_\Psi}\right)^\top \psi_{k_\Psi}\left(\hat{\Psi}_{k_\Psi - 1}\right) = \hat{\Psi}^{[i]}_{k_\Psi}, \tag{2.10b}$$

with $i \in \{1, \dots, m\}$, where the matrices $\hat{V}_{k_\Phi}$ and $\hat{U}^{[i]}_{k_\Psi}$ contain the estimated weights and biases of the last layer. Of particular interest is the expression of the approximation error of the DNNs, which will be crucial for the design of the adaptation laws and the stability analysis later in this thesis. To this end, the following section derives, for each layer j, an explicit formulation of the error between the optimal DNNs and the estimated counterparts, first considering Φ, and then Ψ.

2.3.1 Approximation error of the Drift Dynamics

Consider the error of $\hat{\Phi}$ with respect to Φ for each layer $j \in \{1, 2, \dots, k_\Phi\}$, given by

$$\begin{aligned}
\tilde{\Phi}_j &= \Phi_j - \hat{\Phi}_j \in \mathbb{R}^{L_{\Phi_{j+1}}} \\
&= V_j^\top \phi_j(\Phi_{j-1}) - \hat{V}_j^\top \phi_j(\hat{\Phi}_{j-1}) \\
&= V_j^\top \phi_j(\Phi_{j-1}) - \hat{V}_j^\top \phi_j(\hat{\Phi}_{j-1}) + V_j^\top \phi_j(\hat{\Phi}_{j-1}) - V_j^\top \phi_j(\hat{\Phi}_{j-1}) \\
&= \tilde{V}_j^\top \phi_j(\hat{\Phi}_{j-1}) + V_j^\top \left(\phi_j(\Phi_{j-1}) - \phi_j(\hat{\Phi}_{j-1})\right),
\end{aligned} \tag{2.11}$$

where $\tilde{V}_j = V_j - \hat{V}_j$ and $\tilde{\Phi}_0 = \tilde{V}_0^\top x_h$. Since the ideal activation $\phi_j(\Phi_{j-1})$ is unknown, the expression (2.11) cannot be computed directly. However, ϕ_j can be approximated using first order Taylor expansion centered around $\hat{\Phi}_{j-1}$, obtaining

$$\begin{aligned}
\phi_j(\Phi_{j-1}) &= \phi_j(\hat{\Phi}_{j-1}) + \left.\frac{\partial \phi_j}{\partial \Phi}\right|_{\Phi = \hat{\Phi}_{j-1}} (\Phi_{j-1} - \hat{\Phi}_{j-1}) + \mathrm{O}^2(\tilde{\Phi}_{j-1}) \\
&= \phi_j(\hat{\Phi}_{j-1}) + \hat{\phi}'_j \tilde{\Phi}_{j-1} + \mathrm{O}^2(\tilde{\Phi}_{j-1}),
\end{aligned} \tag{2.12}$$

where $\hat{\phi}'_j = \phi'_j(\hat{\Phi}_{j-1}) \in \mathbb{R}^{L_{\Phi_j} \times L_{\Phi_j}}$ is the Jacobian matrix of the activation function vector $\phi_j(\cdot)$ with respect to its argument, computed in the estimated output of the previous layer $\hat{\Phi}_{j-1}$, while $\mathrm{O}^2(\tilde{\Phi}_{j-1}) \in \mathbb{R}^{L_{\Phi_j}}$ denotes the lumped terms of order higher than one. Substituting (2.12) in (2.11), and having $\phi_j = \phi_j(\Phi_{j-1}), \hat{\phi}_j =$

$\phi_j(\hat{\Phi}_{j-1})$ for sake of readability, one has

$$\begin{aligned}\tilde{\Phi}_j &= \tilde{V}_j^\top \hat{\phi}_j + V_j^\top(\phi_j - \hat{\phi}_j)\\ &= \tilde{V}_j^\top \hat{\phi}_j + V_j^\top(\hat{\phi}_j + \hat{\phi}_j'\tilde{\Phi}_{j-1} + \mathrm{O}^2(\tilde{\Phi}_{j-1}) - \hat{\phi}_j)\\ &= \tilde{V}_j^\top \hat{\phi}_j + V_j^\top\left(\hat{\phi}_j'\tilde{\Phi}_{j-1} + \mathrm{O}^2(\tilde{\Phi}_{j-1})\right)\\ &= \tilde{V}_j^\top \hat{\phi}_j + V_j^\top \hat{\phi}_j'\tilde{\Phi}_{j-1} + V_j^\top \mathrm{O}^2(\tilde{\Phi}_{j-1}).\end{aligned}$$

Exploiting the fact that $V_j = \tilde{V}_j + \hat{V}_j$, it is possible to write

$$\begin{aligned}\tilde{\Phi}_j &= \tilde{V}_j^\top \hat{\phi}_j + V_j^\top \hat{\phi}_j'\tilde{\Phi}_{j-1} + V_j^\top \mathrm{O}^2(\tilde{\Phi}_{j-1})\\ &= \tilde{V}_j^\top \hat{\phi}_j + (\tilde{V}_j^\top + \hat{V}_j^\top)\hat{\phi}_j'\tilde{\Phi}_{j-1} + V_j^\top \mathrm{O}^2(\tilde{\Phi}_{j-1})\\ &= \tilde{V}_j^\top \hat{\phi}_j + \tilde{V}_j^\top \hat{\phi}_j'\tilde{\Phi}_{j-1} + \hat{V}_j^\top \hat{\phi}_j'\tilde{\Phi}_{j-1} + V_j^\top \mathrm{O}^2(\tilde{\Phi}_{j-1})\\ &= \tilde{V}_j^\top \hat{\phi}_j + \hat{V}_j^\top \hat{\phi}_j'\tilde{\Phi}_{j-1} + \Delta_{\Phi_j},\end{aligned}$$

where $\Delta_{\Phi_j} \coloneqq \tilde{V}_j^\top \hat{\phi}_j'\tilde{\Phi}_{j-1} + V_j^\top \mathrm{O}^2(\tilde{\Phi}_{j-1}) \in \mathbb{R}^{L_{\Phi_{j+1}}}$. Before going further with the analysis, the following property of the Kronecker product is introduced.

Lemma 2.1 ([24], Proposition 7.1.9). *Given three matrices $A \in \mathbb{R}^{n\times m}, B \in \mathbb{R}^{m\times q}$ and $C \in \mathbb{R}^{q\times p}$ and letting* $\mathrm{vec}(\cdot)$ *be the vectorization operation, it holds that*

$$\mathrm{vec}(ABC) = (C^\top \otimes A)\mathrm{vec}(B) \in \mathbb{R}^{np}.$$

Since $\tilde{V}_j^\top \hat{\phi}_j \in \mathbb{R}^{L_{\Phi_{j+1}}}$, it holds that

$$\tilde{V}_j^\top \hat{\phi}_j = \mathrm{vec}\left(\tilde{V}_j^\top \hat{\phi}_j\right) = \mathrm{vec}\left(\hat{\phi}_j^\top \tilde{V}_j\right) = \mathrm{vec}\left(\hat{\phi}_j^\top \tilde{V}_j I_{L_{\Phi_{j+1}}}\right),$$

and, applying Lemma 2.1 with $A \equiv \hat{\phi}_j^\top, B \equiv \tilde{V}_j$ and $C \equiv I_{L_{\Phi_{j+1}}}$ it obtains

$$\tilde{V}_j^\top \hat{\phi}_j = (I_{L_{\Phi_{j+1}}}^\top \otimes \hat{\phi}_j^\top)\mathrm{vec}(\tilde{V}_j) = (I_{L_{\Phi_{j+1}}} \otimes \hat{\phi}_j^\top)\mathrm{vec}(\tilde{V}_j).$$

Hence, the error associated with the j-th layer can be then reformulated as

$$\begin{aligned}\tilde{\Phi}_j &= \tilde{V}_j^\top \hat{\phi}_j + \hat{V}_j^\top \hat{\phi}_j'\tilde{\Phi}_{j-1} + \Delta_{\Phi_j}\\ &= (I_{L_{\Phi_{j+1}}} \otimes \hat{\phi}_j^\top)\mathrm{vec}(\tilde{V}_j) + \hat{V}_j^\top \hat{\phi}_j'\tilde{\Phi}_{j-1} + \Delta_{\Phi_j},\end{aligned} \tag{2.13}$$

for $j \in \{1, 2, \dots, k_\Phi\}$, with $\tilde{\Phi}_0 = (I_{L_{\Phi_1}} \otimes x_h^\top)\mathrm{vec}(\tilde{V}_0)$. Before going any further, it is convenient to introduce the oriented product operator.

Definition 2.4. *Given some matrices A_i, with $i \in \{1, 2, \dots, N\}$, characterized by compatible dimensions, then $\prod_{1=1}^{\curvearrowleft N} A_i = A_N A_{N-1} \dots A_1$, represents the oriented product. Moreover, it holds that $\prod_{p=1}^{\curvearrowleft p-1} A_i \coloneqq 1$.*

The overall estimation error of the DNN coincides with (2.13) with $j = k_\Phi$, having

$$\tilde{\Phi}_{k_\Phi} = (I_n \otimes \hat{\phi}_{k_\Phi}^\top)\text{vec}(\tilde{V}_{k_\Phi}) + \hat{V}_{k_\Phi}^\top \hat{\phi}'_{k_\Phi} \tilde{\Phi}_{k_\Phi - 1} + \Delta_{\Phi_{k_\Phi}}.$$

In general, the expression of $\tilde{\Phi}_j$ depends on $\tilde{\Phi}_{j-1}$. So, the above equation has an inherently recursive nature. If one repeatedly substitutes this expression backward down to $\tilde{\Phi}_0$ and groups the similar terms, the overall approximation error can be conveniently expressed as

$$\tilde{\Phi}_{k_\Phi} = \sum_{j=0}^{k_\Phi} \Lambda_{\Phi_j} \text{vec}(\tilde{V}_j) + \sum_{j=1}^{k_\Phi} \Xi_{\Phi_j} \Delta_{\Phi_j}, \tag{2.14}$$

where $\Xi_{\Phi_j} \in \mathbb{R}^{n \times L_{\Phi_{j+1}}}$ and $\Lambda_{\Phi_j} \in \mathbb{R}^{n \times (L_{\Phi_j} L_{\Phi_{j+1}})}$ are given by

$$\Xi_{\Phi_j} \coloneqq \prod_{p=j+1}^{\overset{\frown}{k_\Phi}} \hat{V}_p^\top \hat{\phi}'_p, \tag{2.15}$$

$$\Lambda_{\Phi_j} \coloneqq \Xi_{\Phi_j} \left(I_{L_{\Phi_{j+1}}} \otimes \hat{\phi}_j^\top \right), \tag{2.16}$$

with $\Lambda_{\Phi_0} = \Xi_{\Phi_0} \left(I_{L_{\Phi_1}} \otimes x_h \right)$ and $\Xi_{\Phi_{k_\Phi}} \coloneqq 1$.

2.3.2 Approximation error of the Control Effectiveness Matrix

As detailed in Section 2.2, when a network is used for approximating a matrix, it can be treated as a multi-head DNN, with each head sharing the input, all the hidden layers, and being the approximation of a column. The layer-wise approximation error between Ψ and $\hat{\Psi}$, with $j \in \{0, 1, \ldots, k_\Psi\}$, follows the same reasoning of the previous section. To take into account the fact that the last layer ($j = k_\Psi$) approximates multiple columns, the error associated to this last one is expressed for each column $i \in \{1, 2, \ldots, m\}$. In particular, for $j = \{0, 1, \ldots, k_\Psi - 1\}$, it is possible to express the approximation errors as

$$\begin{aligned}
\tilde{\Psi}_j &= \Psi_j - \hat{\Psi}_j \\
&= U_j^\top \psi_j(\Psi_{j-1}) - \hat{U}_j^\top \psi_j(\hat{\Psi}_{j-1}) \\
&= U_j^\top \psi_j(\Psi_{j-1}) - \hat{U}_j^\top \psi_j(\hat{\Psi}_{j-1}) + U_j^\top \psi_j(\hat{\Psi}_{j-1}) - U_j^\top \psi_j(\hat{\Psi}_{j-1}) \\
&= \left(U_j^\top - \hat{U}_j^\top \right) \psi_j(\hat{\Psi}_{j-1}) + U_j^\top \left(\psi_j(\Psi_{j-1}) - \psi_j(\hat{\Psi}_{j-1}) \right) \\
&= \tilde{U}_j^\top \psi_j(\hat{\Psi}_{j-1}) + U_j^\top \left(\psi_j(\Psi_{j-1}) - \psi_j(\hat{\Psi}_{j-1}) \right),
\end{aligned} \tag{2.17}$$

where $\tilde{U}_j = U_j - \hat{U}_j$ and $\tilde{\Psi}_0 = \tilde{U}_0^\top x_h$. Since the ideal activation $\psi_j(\Psi_{j-1})$ is unknown, it is approximated using first order Taylor expansion centered around $\hat{\Psi}_{j-1}$,

obtaining

$$\begin{aligned}\psi_j(\Psi_{j-1}) &= \psi_j(\hat{\Psi}_{j-1}) + \left.\frac{\partial \psi_j}{\partial \Psi}\right|_{\Psi=\hat{\Psi}_{j-1}} (\Psi_{j-1} - \hat{\Psi}_{j-1}) + \mathrm{O}^2(\tilde{\Psi}_{j-1}) \\ &= \psi_j(\hat{\Psi}_{j-1}) + \hat{\psi}'_j \tilde{\Psi}_{j-1} + \mathrm{O}^2(\tilde{\Psi}_{j-1}),\end{aligned} \tag{2.18}$$

where $\hat{\psi}'_j = \psi'_j(\hat{\Psi}_{j-1}) \in \mathbb{R}^{L_{\Psi_j} \times L_{\Psi_j}}$ is the Jacobian matrix of the activation function vector $\psi_j(\cdot)$ with respect to its argument, computed in the estimated output of the previous layer $\hat{\Psi}_{j-1}$, while $\mathrm{O}^2(\tilde{\Psi}_{j-1}) \in \mathbb{R}^{L_{\Psi_j}}$ denotes the lumped terms of order higher than one. Substituting (2.18) in (2.17), and having $\psi_j = \psi_j(\Psi_{j-1}), \hat{\psi}_j = \hat{\psi}_j(\hat{\Psi}_{j-1})$ for sake of readability, one has

$$\begin{aligned}\tilde{\Psi}_j &= \tilde{U}_j^\top \hat{\psi}_j + U_j^\top \left(\psi_j - \hat{\psi}_j\right) \\ &= \tilde{U}_j^\top \hat{\psi}_j + U_j^\top \left(\hat{\psi}_j + \hat{\psi}'_j \tilde{\Psi}_{j-1} + \mathrm{O}^2(\tilde{\Psi}_{j-1}) - \hat{\psi}_j\right) \\ &= \tilde{U}_j^\top \hat{\psi}_j + U_j^\top \left(\hat{\psi}'_j \tilde{\Psi}_{j-1} + \mathrm{O}^2(\tilde{\Psi}_{j-1})\right) \\ &= \tilde{U}_j^\top \hat{\psi}_j + U_j^\top \psi'_j \tilde{\Psi}_{j-1} + U_j^\top \mathrm{O}^2(\tilde{\Psi}_{j-1}) \\ &= \tilde{U}_j^\top \hat{\psi}_j + (\tilde{U}_j^\top + \hat{U}_j^\top)\hat{\psi}'_j \tilde{\Psi}_{j-1} + U_j^\top \mathrm{O}^2(\tilde{\Psi}_{j-1}) \\ &= \tilde{U}_j^\top \hat{\psi}_j + \tilde{U}_j^\top \hat{\psi}'_j \tilde{\Psi}_{j-1} + \hat{U}_j^\top \hat{\psi}'_j \tilde{\Psi}_{j-1} + U_j^\top \mathrm{O}^2(\tilde{\Psi}_{j-1}) \\ &= \tilde{U}_j^\top \hat{\psi}_j + \hat{U}_j^\top \hat{\psi}'_j \tilde{\Psi}_{j-1} + \Delta_{\Psi_j},\end{aligned}$$

where $\Delta_{\Psi_j} \coloneqq \tilde{U}_j^\top \hat{\psi}'_j \tilde{\Psi}_{j-1} + U_j^\top \mathrm{O}^2\left(\tilde{\Psi}_{j-1}\right) \in \mathbb{R}^{L_{\Psi_{j+1}}}$. Since $\tilde{U}_j^\top \hat{\psi}_j \in \mathbb{R}^{L_{\Psi_j} L_{\Psi_{j+1}}}$, it holds that

$$\tilde{U}_j^\top \hat{\psi}_j = \mathrm{vec}\left(\tilde{U}^\top \hat{\psi}_j\right) = \mathrm{vec}\left(\hat{\psi}_j^\top \tilde{U}_j\right) = \mathrm{vec}\left(\hat{\psi}_j^\top \tilde{U}_j I_{L_{\Psi_{j+1}}}\right),$$

and, applying Lemma 2.1 with $A \equiv \hat{\psi}_j^\top, B \equiv \tilde{U}_j$, and $C \equiv I_{L_{\Psi_{j+1}}}$ it obtains

$$\tilde{U}_j^\top \hat{\psi}_j = \left(I_{L_{\Psi_{j+1}}}^\top \otimes \hat{\psi}_j^\top\right) \mathrm{vec}\left(\tilde{U}_j\right) = \left(I_{L_{\Psi_{j+1}}} \otimes \hat{\psi}_j^\top\right) \mathrm{vec}\left(\tilde{U}_j\right).$$

Hence, the error associated with the j-th layer can be reformulated as

$$\begin{aligned}\tilde{\Psi}_j &= \tilde{U}_j^\top \hat{\psi}_j + \hat{U}_j^\top \hat{\psi}'_j \tilde{\Psi}_{j-1} + \Delta_{\Psi_j} \\ &= \left(I_{L_{\Psi_{j+1}}} \otimes \hat{\psi}_j^\top\right) \mathrm{vec}\left(\tilde{U}_j\right) + \hat{U}_j^\top \hat{\psi}'_j \tilde{\Psi}_{j-1} + \Delta_{\Psi_j},\end{aligned}$$

for $j \in \{0, 1, \ldots, k_\Psi\}$, with $\hat{\Psi}_0 = (I_{L_{\Psi_1}} \otimes x_h^\top)\mathrm{vec}(\tilde{U}_0)$. As anticipated, for the last layer, the error is expressed for each column. In particular, having $\Psi_{k_\Psi}^{[i]}$ and $\hat{\Psi}_{k_\Psi}^{[i]}$ defined as in (2.9b) and (2.10b), respectively, the error associated with the i-th

column is given by

$$\begin{aligned}
\tilde{\Psi}_{k_\Psi}^{[i]} &= \Psi_{k_\Psi}^{[i]} - \hat{\Psi}_{k_\Psi}^{[i]} \\
&= (U_{k_\Psi}^{[i]})^\top \psi_{k_\Psi} - (\hat{U}_{k_\Psi}^{[i]})^\top \hat{\psi}_{k_\Psi} \\
&= (U_{k_\Psi}^{[i]})^\top \psi_j - (\hat{U}_{k_\Psi}^{[i]})^\top \hat{\psi}_j + (U_{k_\Psi}^{[i]})^\top \hat{\psi}_j - (U_{k_\Psi}^{[i]})^\top \hat{\psi}_j \\
&= \left((U_{k_\Psi}^{[i]})^\top - (\hat{U}_{k_\Psi}^{[i]})^\top\right) \hat{\psi}_{k_\Psi} + (U_{k_\Psi}^{[i]})^\top \left(\psi_{k_\Psi} - \hat{\psi}_{k_\Psi}\right) \\
&= \tilde{U}_{k_\Psi}^{[i]^\top} \hat{\psi}_{k_\Psi} + (U_{k_\Psi}^{[i]})^\top \left(\psi_{k_\Psi} - \hat{\psi}_{k_\Psi}\right) \\
&= \tilde{U}_{k_\Psi}^{[i]^\top} \hat{\psi}_{k_\Psi} + (\hat{U}_{k_\Psi}^{[i]})^\top \hat{\psi}'_{k_\Psi} \tilde{\Psi}_{k_\Psi - 1} + \Delta_{\Psi_{k_\Psi}}^{[i]} \\
&= \left(I_m \otimes \hat{\psi}_{k_\Psi}^\top\right) \mathrm{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right) + (\hat{U}_{k_\Psi}^{[i]})^\top \hat{\psi}'_{k_\Psi} \tilde{\Psi}_{k_\Psi - 1} + \Delta_{\Psi_{k_\Psi}}^{[i]}.
\end{aligned}$$

Exploiting its recursive nature, it can be written as

$$\tilde{\Psi}_{k_\Psi}^{[i]} = \Lambda_{\Psi_{k_\Psi}}^{[i]} \mathrm{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right) + \Delta_{\Psi_{k_\Psi}}^{[i]} + \sum_{j=0}^{k_\Psi - 1} \Lambda_{\Psi_j}^{[i]} \mathrm{vec}\left(\tilde{U}_j\right) + \sum_{j=1}^{k_\Psi - 1} \Xi_{\Psi_j}^{[i]} \Delta_{\Psi_j}, \tag{2.19}$$

where

$$\Xi_{\Psi_j}^{[i]} = (\hat{U}_{k_\Psi}^{[i]})^\top \hat{\psi}'_{k_\Psi} \overset{\curvearrowleft}{\prod}_{l=j+1}^{k_\Psi - 1} \hat{U}_l^\top \hat{\psi}'_l \in \mathbb{R}^{n \times L_{\Psi_{j+1}}}, \tag{2.20a}$$

$$\Lambda_{\Psi_j}^{[i]} = \Xi_{\Psi_j}^{[i]} (I_{L_{\Psi_{j+1}}} \otimes \psi_{L_{\Psi_j}}^\top) \in \mathbb{R}^{n \times L_{\Psi_j} L_{\Psi_{j+1}}}, \tag{2.20b}$$

with $\Xi_{\Psi_{k_\Psi}}^{[i]} := 1, \Xi_{\Psi_{k_\Psi - 1}}^{[i]} := \left(\hat{U}_{k_\Psi}^{[i]}\right)^\top \hat{\psi}'_{k_\Psi}$, $\Lambda_{\Psi_0}^{[i]} = \Xi_{\Psi_0}^{[i]} (I_{L_{\Psi_1}} \otimes x_h^\top)$ and $\Lambda_{\Psi_{k_\Psi}}^{[i]} = (I_m \otimes \hat{\psi}_{k_\Psi}^\top)$.

2.3.3 Weights initialization

Intuitively, the approximation accuracy of $\hat{\Phi}$ and $\hat{\Psi}$ depends critically on their initial weight estimates. In particular, better initializations lead to faster convergence towards the ideal weights, reducing transient errors and improving overall function approximation quality. Classical random initializations may lead to saturation of hidden units and vanishing or exploding gradients in deep architectures, slowing down or even preventing weights adaptation convergence. Motivated by the analysis in [25], the weights of all layers of $\hat{\Phi}$ and $\hat{\Psi}$ are initialized using the Xavier-Glorot scheme

$$W_j \sim U\left[-\frac{\sqrt{6}}{\sqrt{n_j + n_{j+1}}}, \frac{\sqrt{6}}{\sqrt{n_j + n_{j+1}}}\right],$$

where $U[-a, a]$ denotes the uniform distribution over $(-a, a)$, and n_j and n_{j+1} are the input and output dimensions of layer j, respectively. This variance-preserving initialization maintains the singular values of the layer Jacobians close to one [25], ensuring more stable activation and gradient statistics across layers and thus facilitating both offline and online training.

Part II

Deep Neural Network-based Sliding Mode Observers

Chapter 3

DNN-SMO Strategy

Controller synthesis often relies on the knowledge of the system dynamics to ensure stability and performance guarantees. However, precise dynamic models are often unavailable in practical applications, motivating different control strategies to deal with uncertain dynamics. Traditional approaches rely on either robust designs, bounding uncertainty [1, 2, 26], or adaptive schemes, requiring parameter identifiability [27, 28]. Recent ANN-based methods leverage their universal approximation property to directly estimate plant dynamics from data, bypassing explicit model identification. However, offline training, required to adjust network weights, introduces significant limitations. Once trained, these networks operate as static maps within control loops. Moreover, ANNs suffer from approximation errors requiring robust control augmentation, while, measurement noise degrades data quality, compromising both training effectiveness and control performance. Online adaptation overcomes these limitations by continuously adjusting ANN weights via Lyapunov-derived adaptation laws during system operation. A remarkable example is the DNN-ISM approach [7, 8], demonstrating online adaptation effectiveness. However, closer analysis reveals poor approximation accuracy during transients phases. The proposed DNN-SMO strategy overcomes this limitation through a dual-DNN approximation of the unknown dynamics, with SMO-driven neural adaptation principles [10] ensured via Lyapunov stability analysis.

3.1 Problem Formulation

Consider a nonlinear control-affine system affected by disturbance

$$\dot{x}(t) = f(x(t), t) + B(x(t), t)u(t) + h(x(t), t), \tag{3.1}$$

where $x \in \mathcal{X} \subset \mathbb{R}^n$ denotes the full state vector assumed measurable, meaning that the output vector coincides with the state vector, with $\mathcal{X}$ being a compact set containing the origin, $u \in \mathbb{R}^m$ is the control input vector, $h : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ is the perturbation vector satisfying Assumption 1.4, while $f : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ and $B : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^{n \times m}$ denote the drift dynamics and the control effectiveness matrix assumed unknown and satisfying Assumption 1.1 and 1.2, respectively. Thus, direct control synthesis is infeasible. To overcome this, the idea is to approximate these dynamics using a dual-DNN architecture, leveraging the universal approximation results of Section 2.3, via SMO-based online adaptation. This decouples the adaptation laws from the controller and enables controller design using the DNN approximations.

3.2 The DNN-SMO scheme

In the following, the DNN-SMO scheme, depicted in Figure 3.1, is presented and analyzed. Considering the problem formulation in Section 3.1, and taking advantage

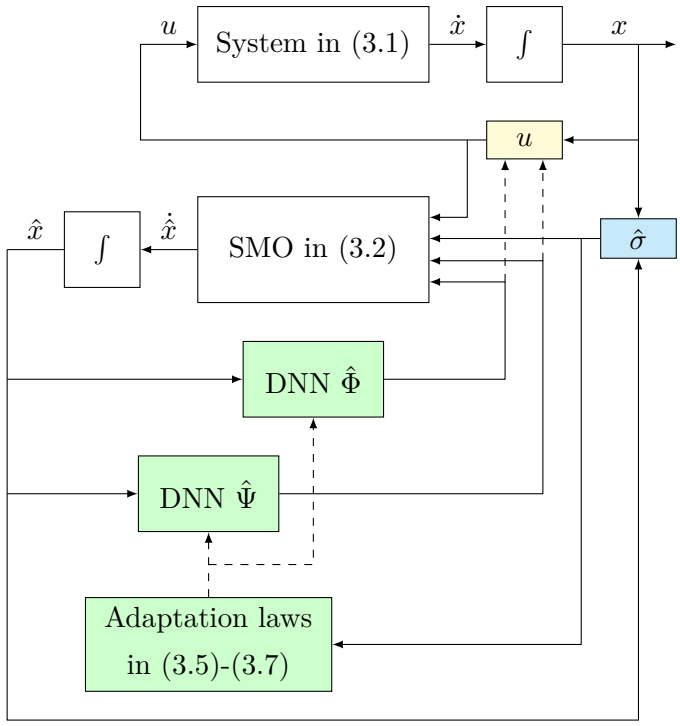

Figure 3.1: Block diagram of the DNN-SMO scheme. The blocks associated with the DNNs, the observer sliding variable and the control law are highlighted in green, blue and yellow, respectively.

of the approximation property of the DNN exploited in Section 2.3, a DNN-based SMO model can be defined as

$$\begin{aligned}\dot{\hat{x}}(t) &= \hat{f}(\hat{x}(t),t) + \hat{B}(\hat{x}(t),t)u(t) + l(x(t),\hat{x}(t)) \\ &= \hat{f}(\hat{x}(t),t) + \sum_{i=1}^{m} \hat{B}^{(i)}(\hat{x}(t),t)u_i(t) + l(x(t),\hat{x}(t)),\end{aligned}$$

Substituting $\hat{f}$ and $\hat{B}^{(i)}$ with their definitions in (2.10) the above system becomes

$$\dot{\hat{x}}(t) = \hat{\Phi}_{k_\Phi} + \sum_{i=1}^{m} \hat{\Psi}_{k_\Psi}^{[i]} u_i(t) + l(x(t),\hat{x}(t)), \tag{3.2}$$

with initial condition $\hat{x}(0) = x(0) \in \mathcal{X}$ and where $l : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^n$ is the correction signal, defined in (1.16), used to drive the estimated states toward the measured ones, hereafter recalled for reader's convenience.

$$l(x(t),\hat{x}(t)) = \hat{\rho}\frac{\hat{\sigma}(x(t),\hat{x}(t))}{\|\hat{\sigma}(x(t),\hat{x}(t))\|}, \tag{3.3}$$

with $\hat{\rho} \in \mathbb{R}_{>0}$ being the observer gains, while $\hat{\sigma} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^n$ is the observer sliding variable defined as

$$\hat{\sigma}(x(t),\hat{x}(t)) = \hat{C}(x(t) - \hat{x}(t)), \tag{3.4}$$

with $\hat{C} \in \mathbb{R}^{n\times n}$ satisfying the following assumption.

Assumption 3.1. *The design matrix $\hat{C} \in \mathbb{R}^{n\times n}$ is chosen symmetric and positive definite.*

The proposed update rules depend explicitly on the observer sliding variables $\hat{\sigma}$ and derive from the Lyapunov-based analysis. For what concerns the DNN $\hat{\Phi}$, the adaptation of all layers $j \in \{0, 1, \dots, k_\Phi\}$, is performed according to

$$\operatorname{vec}\left(\dot{\hat{V}}_j\right) = \underset{\mathcal{B}_{\Phi_j}}{\operatorname{proj}}\left(\Gamma_{\Phi_j}\Lambda_{\Phi_j}^\top \hat{C}^\top \hat{\sigma}\right) \in \mathbb{R}^{L_{\Phi_j}L_{\Phi_{j+1}}}, \tag{3.5}$$

where $\Lambda_{\Phi_j} \in \mathbb{R}^{n\times L_{\Phi_j}L_{\Phi_{j+1}}}$ is the one defined in (2.16) and $\Gamma_{\Phi_j} \in \mathbb{R}^{L_{\Phi_j}L_{\Phi_{j+1}}\times L_{\Phi_j}L_{\Phi_{j+1}}}$ being the adaptation rate matrix, defined as diagonal with positive entries, while $\operatorname{proj}(\cdot)$ is the projection operator which ensures that the estimated weights remain bounded in an admissible sets depending on the bounds introduced in Assumption 2.2. As for the DNN $\hat{\Psi}$, for $j \in \{0, 1, \dots, k_\Psi - 1\}$, one has that

$$\operatorname{vec}\left(\dot{\hat{U}}_j\right) = \underset{\mathcal{B}_{\Psi_j}}{\operatorname{proj}}\left(\Gamma_{\Psi_j}\left(\sum_{i=1}^{m} u_i(\Lambda_{\Psi_j}^{[i]})^\top\right)\hat{C}^\top\hat{\sigma}\right) \in \mathbb{R}^{L_{\Psi_j}L_{\Psi_{j+1}}}, \tag{3.6}$$

where $\Lambda_{\Psi_j}^{[i]} \in \mathbb{R}^{n\times L_{\Psi_j}L_{\Psi_{j+1}}}$ is defined as in (2.20b), $u_i \in \mathbb{R}$ is the i-th component of the control law, while $\Gamma_{\Psi_j} \in \mathbb{R}^{L_{\Psi_j}L_{\Psi_{j+1}}\times L_{\Psi_j}L_{\Psi_{j+1}}}$ is the adaptation rate matrix,

chose as diagonal with positive entries. The weight sub-matrices of last layer of $\hat{\Psi}$ are adapted according to

$$\operatorname{vec}\left(\dot{\hat{U}}_{k_\Psi}^{[i]}\right) = \underset{\mathcal{B}_{\Psi_{k_\Psi}}}{\operatorname{proj}}\left(\Gamma_{\Psi_{k_\Psi}} u_i (\Lambda_{\Psi_{k_\Psi}}^{[i]})^\top \hat{C}^\top \hat{\sigma}\right) \in \mathbb{R}^{L_{k_\Psi} n}, \tag{3.7}$$

with $i \in \{1, 2, \ldots, m\}$. As mentioned above, the aim of the projection operator, defined in [28, Appendix E], is to maintain the weights in the admissible sets $\mathcal{B}_{\Phi_j} \subset \mathbb{R}^{L_{\Phi_j} L_{\Phi_{j+1}}}$, with $j \in \{0, 1, \ldots, k_\Phi\}$, and, $\mathcal{B}_{\Psi_j} \subset \mathbb{R}^{L_{\Psi_j} L_{\Psi_{j+1}}}$, conveniently defined as

$$\begin{aligned}
\mathcal{B}_{\Phi_j} &\coloneqq \left\{\operatorname{vec}\left(\hat{V}_j\right) \in \mathbb{R}^{L_{\Phi_j} L_{\Phi_{j+1}}} : \left\|\operatorname{vec}\left(\hat{V}_j\right)\right\| - \bar{V} \leq 0\right\}, \\
\mathcal{B}_{\Psi_j} &\coloneqq \left\{\operatorname{vec}\left(\hat{U}_j\right) \in \mathbb{R}^{L_{\Psi_j} L_{\Psi_{j+1}}} : \left\|\operatorname{vec}\left(\hat{U}_j\right)\right\| - \bar{U} \leq 0\right\}.
\end{aligned}$$

Lemma 3.1 (Boundedness of the DNN weights and biases). *Let $\mathcal{B}_{\Phi_j}$ and $\mathcal{B}_{\Psi_j}$ be the admissible sets, while* $\operatorname{proj}(\cdot)$ *is the smooth projection operator. Moreover, consider the weight adaptation laws in* (3.5), (3.6) *and* (3.7). *Then, one has that*

1. *$V_j(t) \in \mathcal{B}_{\Phi_j}^\alpha$, for all $t \geq 0$ and $j \in \{0, 1, \ldots, k_\Phi\}$;*
2. *$U_j(t) \in \mathcal{B}_{\Psi_j}^\alpha$, for all $t \geq 0$ and $j \in \{0, 1, \ldots, k_\Psi\}$;*
3. *$-\operatorname{vec}\left(V_j - \hat{V}_j\right)^\top \Gamma_{\Phi_j}^{-1} \underset{\mathcal{B}_{\Phi_j}}{\operatorname{proj}}(\tau) \leq -\operatorname{vec}\left(V_j - \hat{V}_j\right)^\top \Gamma_{\Phi_j}^{-1} \tau$, $\forall j \in \{0, 1, \ldots, k_\Phi\}$;*
4. *$-\operatorname{vec}\left(U_j - \hat{U}_j\right)^\top \Gamma_{\Psi_j}^{-1} \underset{\mathcal{B}_{\Psi_j}}{\operatorname{proj}}(\tau) \leq -\operatorname{vec}\left(U_j - \hat{U}_j\right)^\top \Gamma_{\Psi_j}^{-1} \tau$, $\forall j \in \{0, 1, \ldots, k_\Psi\}$.*

Since the activation functions of the network $\Phi, \hat{\Phi}, \Psi$ and $\hat{\Psi}$ are chosen differentiable and Lipschitz continuous, their Jacobian matrices are norm-bounded. Combined with Lemma 3.1 and the classical result that, for standard activation functions, the higher order of the Taylor's expansion are bounded over a compact domain [29], the approximation residuals Δ_{Φ_j} and Δ_{Ψ_j} introduced in Section 2.3.1 and Section 2.3.2, respectively, are bounded as well.

Proposition 3.1. *There exist some constants $\bar{c}_\Phi, \bar{c}_\Psi \in \mathbb{R}_{>0}$ such that the residual terms in* (2.14) *and* (2.19) *as*

$$\left\|\Delta_{\Phi_{k_\Phi}} + \sum_{j=1}^{k_\Phi - 1} \Xi_{\Phi_j} \Delta_{\Phi_j}\right\| \leq \bar{c}_\Phi, \quad \left\|\Delta_{\Psi_{k_\Psi}}^{[i]} + \sum_{j=1}^{k_\Psi - 1} \Xi_{\Psi_j}^{[i]} \Delta_{\Psi_j}\right\| \leq \bar{c}_\Psi.$$

3.2.1 Sliding Mode Existence

The main theoretical results related to DNN-SMO architecture are presented in the following.

Theorem 3.1. *Consider the system in* (3.1) *and the DNN-based SMO in* (3.2)*, with* l *and* $\hat{\sigma}$ *defined as in* (3.3) *and* (3.4)*, respectively. Moreover, the DNNs weights are adapted according to the adaptation laws* (3.5) - (3.7)*. If Assumption 1.1, 1.2, 3.1, and Proposition 3.1 hold, and*

$$\hat{\rho} > \frac{\bar{\lambda}(\hat{C})\{\bar{c}_\Phi + \bar{\varepsilon}_\Phi + m(\bar{c}_\Psi + \bar{\varepsilon}_\Psi)\|u\| + \bar{h}\} + \bar{\eta}}{\underline{\lambda}(\hat{C})},$$

with $\bar{\eta} \in \mathbb{R}_{>0}$ *being design parameter, then,* $\hat{\sigma}(x(t), \hat{x}(t)) \to 0_n$ *for* $t \to \infty$.

Proof. See Appendix A.3. □

In other words, choosing the observer gain according to Theorem 3.1 enforces sliding mode on $\hat{\sigma}(x(t), \hat{x}(t)) = 0_n$ asymptotically, which also drives the estimation error to zero, meaning that the DNN-based SMO model act as the real one. However, this does not imply perfect DNNs learning. In fact, $\hat{\Phi}$ and $\hat{\Psi}$, characterized by estimated weights, introduces unavoidable approximation errors. In particular, for the drift dynamics components

$$\begin{aligned} f - \hat{\Phi}_{k_\Phi} &= \tilde{\Phi}_{k_\Phi} + \varepsilon_\Phi \\ &= \Lambda_{\Phi_{k_\Phi}} \operatorname{vec}\left(\tilde{V}_{k_\Phi}\right) + \Delta_{\Phi_{k_\Phi}} + \sum_{j=0}^{k_\Phi - 1} \Lambda_{\Phi_j} \operatorname{vec}\left(\tilde{V}_j\right) + \sum_{j=1}^{k_\Phi - 1} \Xi_{\Phi_j} \Delta_{\Phi_j} + \varepsilon_\Phi, \end{aligned}$$

and for the control effectiveness term

$$\begin{aligned} B^{(i)} - \hat{\Psi}^{[i]}_{k_\Psi} &= \tilde{\Psi}^{[i]}_{k_\Psi} + \varepsilon^{(i)}_\Psi \\ &= \Lambda^{[i]}_{\Psi_{k_\Psi}} \operatorname{vec}\left(\tilde{U}^{[i]}_{k_\Psi}\right) + \sum_{j=0}^{k_\Psi - 1} \Lambda^{[i]}_{\Psi_j} \operatorname{vec}\left(\tilde{U}_j\right) + \Delta^{[i]}_{\Psi_{k_\Psi}} + \sum_{j=1}^{k_\Psi - 1} \Xi^{[i]}_{\Psi_j} \Delta_{\Psi_j} + \varepsilon^{(i)}_\Psi, \end{aligned}$$

with $i \in \{1, 2, \ldots, m\}$. These errors are not quantifiable, since depend on intrinsically unknown quantities $\varepsilon_\Phi, \Delta_{\Phi_j}$, with $j \in \{0, 1, \ldots, k_\Phi\}$, $\varepsilon_\Psi, \Delta_{\Psi_j}$, with $j \in \{0, 1, \ldots, k_\Psi\}$. Nevertheless, the analysis performed earlier demonstrates that all the above approximation errors are bounded and compensated by the SMO correction term, ensuring accurate state estimation despite imperfect DNN approximations. Moreover, appropriate weights initialization, as specified in Section 2.3.3, positively impacts both DNNs approximation accuracy and sliding mode reaching time, ensuring better performance.

3.3 DNN-SMO with Barrier Functions

While the analysis of the previous section establishes the convergence of the observer sliding variable and the boundedness of DNNs parameters, it does not give any performance guarantees during transients. This can be a point of concern in practical applications, as the estimation error may exhibit undesirable overshoot or rapid transient growth, particularly when the DNNs are still in the early stages of learning and their approximation accuracy is poor. To address this limitation Barrier Lyapunov Functions (BLFs) are introduced [30], imposing a virtual constraint on the observer sliding variable, which is forced to remain within a prescribed compact set during the entire system evolution, provided that the initial condition lies within this set. This mechanism not only improves the practical implementation by reducing overshoots, but also a more precise quantification of the estimation error in a bounded region, essential for the Model Predictive Control formulation developed in the subsequent chapter. The BLF is incorporated directly into the design of the weight adaptation laws, resulting in modified update rules that implicitly enforce the barrier constraint.

3.3.1 Preliminaries on BLFs

Before presenting the complete analysis, some preliminary concepts about BLFs are required. Let $\Omega \subset \mathbb{R}^m$ be an open set containing the origin as an interior point. A BLF is a smooth positive definite scalar function $\beta : \Omega \to \mathbb{R}$ such that

$$\lim_{y \to \partial\Omega} \beta(y) = +\infty, \tag{3.8}$$

where $\partial\Omega$ denotes the boundary of Ω. In particular, a BLF satisfies that $\beta(y(t)) \leq \bar{\beta}$, for $t \geq 0$ along the trajectories of $y(t)$, with $\bar{\beta} \in \mathbb{R}_+$, if $y(0) \in \Omega$. A BLF can be used to certify stability of a desired equilibrium point, together with the invariance property of Ω [30, 31]. In particular, when not explicitly stated, in this thesis, the log-type BLF is considered

$$\beta(y) = -\log\left(\frac{\varepsilon_y^2}{\varepsilon_y^2 - \|y\|^2}\right), \tag{3.9}$$

with $\varepsilon_y \in \mathbb{R}_{>0}$.

3.3.2 The DNN-SMO scheme with BLFs

The BLF-modified DNN-SMO scheme, depicted in Figure 3.2, is now presented and analyzed. Building upon the DNN-SMO architecture from Chapter 3.2, a BLF

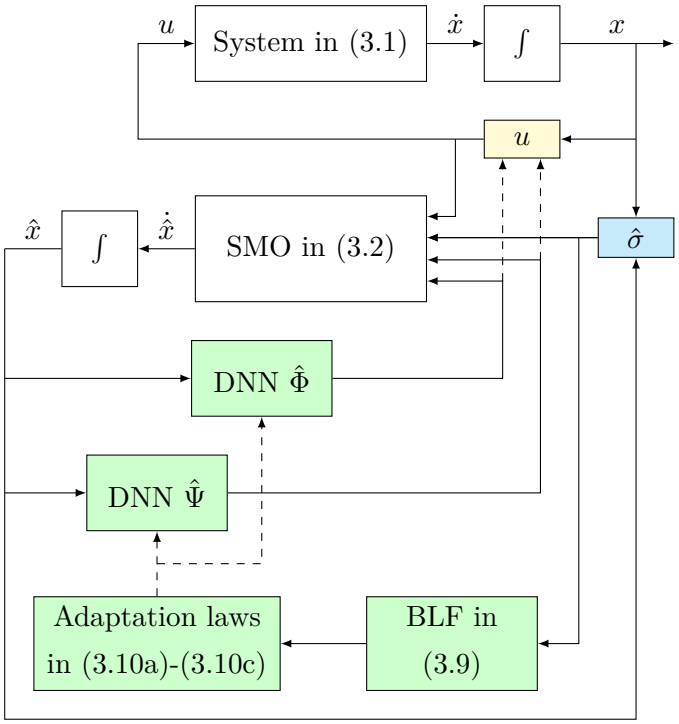

Figure 3.2: Block diagram of the BLF based DNN-SMO scheme. The blocks associated with the DNNs, the observer sliding variable and the control law are highlighted in green, blue and yellow, respectively.

$\beta : \mathbb{R}^n \to \mathbb{R}$ defined as in (3.8) to limit $\hat{\sigma}$ into the sets $\Omega_\sigma \coloneqq \{\hat{\sigma} \in \mathbb{R}^n : \|\hat{\sigma}\| \leq \varepsilon_\sigma\}$, with $\varepsilon_\sigma \in \mathbb{R}_{>0}$, is directly employed in the design of the weights adaptation law. In particular, the weights of the DNN $\hat{\Phi}$ are updated modifying the expression (3.5), according to

$$\operatorname{vec}\left(\dot{\hat{V}}_j\right) = \underset{\mathcal{B}_{\Phi_j}}{\operatorname{proj}}\left(\Gamma_{\Phi_j}\Lambda_{\Phi_j}^\top \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{C}^\top\hat{\sigma}\right), \tag{3.10a}$$

with $j \in \{0, \ldots, k_\Phi\}$. As for the $\hat{\Psi}$, for $j = \{0, 1, \ldots, k_{\Psi-1}\}$, the weights are adapted modifying the expressions (3.6) and (3.7), according to

$$\operatorname{vec}\left(\dot{\hat{U}}_j\right) = \underset{\mathcal{B}_{\Psi_j}}{\operatorname{proj}}\left(\Gamma_{\Psi_j}\left(\sum_{i=1}^{m} u_i(\Lambda_{\Psi_j}^{[i]})^\top\right)\frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{C}^\top\hat{\sigma}\right), \tag{3.10b}$$

$$\operatorname{vec}\left(\dot{\hat{U}}_{k_\Psi}^{[i]}\right) = \underset{\mathcal{B}_{\Psi_{k_\Psi}}}{\operatorname{proj}}\left(\Gamma_{\Psi_{k_\Psi}} u_i(\Lambda_{\Psi_{k_\Psi}}^{[i]})^\top \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{C}^\top\hat{\sigma}\right), \tag{3.10c}$$

with $i \in \{1, 2, \ldots, m\}$.

3.3.3 Sliding Mode Existence

Theorem 3.2. *Consider the system in* (3.1) *and the DNN-based SMO in* (3.2)*, with* l *and* $\hat{\sigma}$ *defined as in* (3.3) *and* (3.4)*, respectively. Moreover, the DNNs weights are adapted according to the adaptation laws* (3.10a) - (3.10c)*. If Assumption 1.1, 1.2, 3.1, and Proposition 3.1 hold, and* β *chosen to satisfy* (3.8)*, and*

$$\hat{\rho} > \frac{\bar{\lambda}(\hat{C})\{\bar{c}_\Phi + \bar{\varepsilon}_\Phi + m(\bar{c}_\Psi + \bar{\varepsilon}_\Psi)\|u\| + \bar{h}\} + \bar{\eta}}{\underline{\lambda}(\hat{C})},$$

with $\bar{\eta} \in \mathbb{R}_{>0}$ *being design parameter, then,* $\hat{\sigma}(x(t), \hat{x}(t)) \to 0_n$ *for* $t \to \infty$ *and it is guaranteed that* $\|\hat{\sigma}(x(t), \hat{x}(t))\| \leq \varepsilon_\sigma$ *for* $t \geq 0$*.*

Proof. See Appendix A.4. □

Thanks to this result, the observer sliding variable is explicitly bounded by a prescribed limit $\varepsilon_\sigma \in \mathbb{R}_{>0}$, addressing the key limitation of the DNN-SMO scheme illustrated at the beginning of this chapter.

Chapter 4

DNN-SMO based Control

As established in the previous chapter, the DNN-SMO architecture provides accurate approximation of unknown system dynamics. Moreover, the BLF-version guarantees known estimation bound from the first time instant increasing the overall performance. Building upon these results, the present chapter addresses the DNN-SMO based controller design.

4.1 DNN-SMO based ISM

As demonstrated in Chapter 1, ISM control eliminates the vulnerable reaching phase by enforcing sliding motion from initial time instant through a specifically designed integral sliding variable, defined as the sum of two components: a conventional SMC sliding variable $\sigma_0 : \mathcal{X} \to \mathbb{R}^m$ and the so-called transient variable $z : \mathcal{X} \to \mathbb{R}^m$. However, the computation of such a component demands precise knowledge of the system dynamics, often unavailable in practical applications. In this context, the DNN-SMO strategy naturally emerges as a possible solution, providing an estimated dynamics.

4.1.1 Problem Formulation

Consider a perturbed nonlinear control-affine system affected by matched disturbance

$$\dot{x}(t) = f(x(t), t) + B(x(t), t)u(t) + h(x(t), t), \tag{4.1}$$

where $x \in \mathcal{X} \subset \mathbb{R}^n$ is the state vector assumed measurable, meaning that the output vector coincides with the state vector, $u \in \mathbb{R}^m$ defines the control vector, $f : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ is the drift dynamics, $B : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^{n \times m}$ denotes the control

effectiveness matrix, while $h : \mathcal{X} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^m$ is the matched perturbation vector, defined as in (1.10). Moreover, f and B satisfy Assumption 1.1 and 1.2 respectively, while h is bounded according to Assumption 1.3. Given a desired state trajectory $x^\star \in \mathbb{R}^n$, the tracking error dynamics $e(t) = x(t) - x^\star(t) \in \mathbb{R}^n$ can be expressed as

$$\dot{e}(t) = f(x(t), t) + B(x(t), t)u(t) + h(x(t), t) - \dot{x}^\star(t),$$

with initial condition $e(0) = x(0) - x^\star(0)$. To stabilize the error system around the origin while rejecting the matched disturbance from the initial time instant, an ISM controller can be designed as outlined in Section 1.2. However, as specified in Section 3.1, the dynamics of the system is unknown, implying the impossibility to compute the transient variable, and, as consequence, enforce a sliding mode. To overcome this limitation, the DNN-SMO strategy is employed.

4.1.2 The DNN-SMO based ISM

Leveraging the theoretical results from Chapter 3, a DNN-based transient variable can be defined as

$$\begin{aligned}\hat{z}(\hat{x}(t)) &= \sigma_0(x(0)) + \int_0^t \frac{\partial \sigma_0(x(\tau))}{\partial x}\Big[\hat{f}(\hat{x}(\tau), \tau) + \hat{B}(\hat{x}(\tau), \tau)u_n(\tau) - \dot{x}^\star(\tau)\Big]d\tau \\ &= \sigma_0(x(0)) + \int_0^t \frac{\partial \sigma_0(x(\tau))}{\partial x}\Big[\hat{f}(\hat{x}(\tau), \tau) + \sum_{i=1}^m \hat{B}^{(i)}(\hat{x}(\tau), \tau)u_{n,i}(\tau) - \dot{x}^\star(\tau)\Big]d\tau.\end{aligned}$$

Substituting $\hat{f}$ and $\hat{B}^{(i)}$ with their definitions in (2.10) the above equation becomes

$$\hat{z}(\hat{x}(t)) = \sigma_0(x(0)) + \int_0^t \frac{\partial \sigma_0(x(\tau))}{\partial x}\Big[\hat{\Phi}_{k_\Phi} + \sum_{i=1}^m \hat{\Psi}_{k_\Psi}^{[i]} u_{n,i}(\tau) - \dot{x}^\star(\tau)\Big]d\tau, \tag{4.2}$$

implying that

$$\dot{\hat{z}}(\hat{x}(t)) = \frac{\partial \sigma_0(x(t))}{\partial x}\Big[\hat{\Phi}_{k_\Phi} + \sum_{i=1}^m \hat{\Psi}_{k_\Psi}^{[i]} u_{n,i}(t) - \dot{x}^\star(t)\Big],$$

with initial condition $\hat{z}(\hat{x}(0)) = \sigma_0(x(0))$. This formulation enables standard controller design while overcoming the limitation imposed by assumption. The DNN-SMO based ISM strategy can operate both with and without BLF modification, depending on the required performance level. Effectiveness is verified through simulations on two systems: a Duffing oscillator and a 3-DoF robotic manipulator. Results are presented in Chapter 5.

4.2 DNN-SMO based MPC/ISM

This section addresses the controller design problem for state and input constrained systems. In particular, the objective is to synthesize a control law that explicitly accounts for constraints. Among constrained control techniques, Model Predictive Control (MPC) naturally stands out due to its optimization-based framework, which inherently incorporates constraints within the optimization problem. Moreover, works such as [32, 33, 34, 35, 36] have introduced and developed a hybrid MPC/ISM architecture, shown in Figure 4.1, in which ISM mitigates matched disturbances while robust MPC with tightened constraints handles unmatched ones. Both components require complete knowledge of the nominal dynamics, which is unavailable by assumption in Section 3.1. In [9], a DNN-based MPC/ISM architecture is developed, demonstrating online DNN adaptation effectiveness. However, it suffers from transient estimation overshoot, critical for MPC feasibility, necessitating an auxiliary controller until DNNs accuracy stabilizes. The proposed scheme extends this framework by integrating the BLF based DNN-SMO strategy, which drastically reduces transient overshoot and enables immediate MPC/ISM deployment.

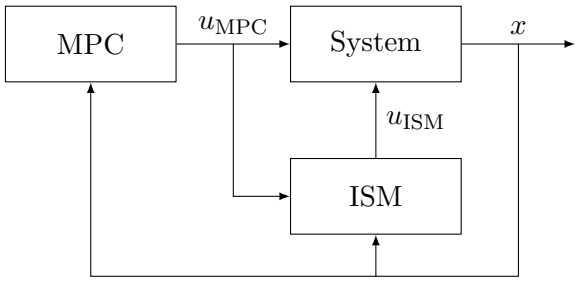

Figure 4.1: Block diagram of the MPC/ISM control architecture.

4.2.1 Problem Formulation

Consider the nonlinear perturbed system in (4.1) for which it is desired to impose the following state and input constraints

$$x \in \mathcal{X}_c \subset \mathcal{X}, \tag{4.3a}$$

$$u \in \mathcal{U} \subset \mathbb{R}^m, \tag{4.3b}$$

with $\mathcal{X}_c$ and $\mathcal{U}$ being compact sets containing the origin. In this thesis, $\mathcal{U}$ is assumed box-shaped, i.e., constraint set in which each i-th element of the control vector is

limited between a minimum and maximum value $\underline{u}_i, \bar{u}_i \in \mathbb{R}$, with $\underline{u}_i < \bar{u}_i$, as

$$\mathcal{U} \coloneqq \{u \in \mathbb{R}^m : \underline{u}_i \leq u_i \leq \bar{u}_i, \forall i\},$$

with $i \in \{1, 2, \ldots, m\}$. As previously detailed, the system (4.1) can be controlled rejecting matched disturbances while satisfying constraints (4.3), implementing an MPC/ISM controller that leverages ISM for disturbance rejection and robust MPC for constraint handling. Let $x^\star \in \mathbb{R}^n$ denote a desired state trajectory, with the conventional sliding variable σ_0 and transient variable $z(x(t))$ defined in Section 1.2. The nominal control law is selected as the MPC law $u_{\text{MPC}} \in \mathbb{R}^m$ (defined later) yielding the following full control law

$$\begin{aligned} u(t) &= u_n(t) + u_r(t) \\ &= u_{\text{MPC}}(t) - \rho \frac{\sigma(x(t))}{\|\sigma(x(t))\|}. \end{aligned} \tag{4.4}$$

Since part of the control compensates matched disturbances and, at the same time, it is required that $u(t) \in \mathcal{U}$, for all $t \geq 0$, a reduced constraint set for the input is defined as

$$\bar{\mathcal{U}} \coloneqq \{u \in \mathbb{R}^m : \underline{u}_i + \rho \leq u_i \leq \bar{u}_i - \rho, \forall i\},$$

with $\sigma_i \in \mathbb{R}$ being the i-th component of the integral sliding variable. Since the ISM controller provides robustness against matched disturbances from the initial time instant, the MPC solves the Finite-Horizon Optimal Control Problem (FHOCP) over nominal dynamics $h(x(t), t) = 0_m$. In particular, let $t_k \geq 0$ be the time instant at which the signal u must be computed, then a properly designed cost function is optimized over the control sequence $u_{[t_k, t_{k+N-1}]}$, where $N \in \mathbb{N}_{>0}$ represents the prediction horizon. Such a cost function can be defined as

$$J(x(t_k), u_{[t_k, t_{k+N-1}]}, N) = \int_{t_k}^{t_{k+N-1}} \ell(x(\tau), u(\tau)) d\tau + V_f(x(t_{k+N})), \tag{4.5}$$

with $\ell : \mathcal{X}_c \times \bar{\mathcal{U}} \to \mathbb{R}_{\geq 0}$ being the so-called stage cost, chosen as

$$\ell(x(\tau), u(\tau)) = \|x(\tau) - x^\star\|_Q^2 + \|u(\tau)\|_R^2$$

with $Q \in \mathbb{R}^{n \times n}$ and $R \in \mathbb{R}^{m \times m}$ being positive definite matrices. Regarding $V_f : \mathcal{X}_f \to \mathbb{R}_{\geq 0}$, it represents the terminal cost, instrumental for guaranteeing closed-loop stability. It can be selected as follows

$$V_f(x(t_{k+N})) = \|x(t_{k+N}) - x^\star\|_\Pi^2,$$

where $\Pi \in \mathbb{R}^{n\times n}$ is a positive definite matrix. Moreover, the so-called terminal constraint $x(t_{k+N}) \in \mathcal{X}_f$ must be introduced, with terminal set $\mathcal{X}_f \subseteq \mathcal{X}_c$ defined as

$$\mathcal{X}_f = \{x \in \mathcal{X}_c : \|x - x^\star\|_\Pi^2 < \xi\}, \tag{4.6}$$

with $\xi \in \mathbb{R}_{>0}$. A possible choice for the terminal cost and the terminal constraint is the one proposed in [32]. Therefore, MPC problem can be formulated as

$$\begin{aligned} \min_{u_{[t_k,t_{k+N-1}]}} \quad & J(x(t_k), u_{[t_k,t_{k+N-1}]}, N) \\ \text{s.t.} \quad & \text{dynamics (4.1) with } h(x(t),t) = 0_m \\ & x(t_k) \in \mathcal{X}_c,\ \forall t_k \in [t_k, t_{k+N-1}] \\ & u(t_k) \in \bar{\mathcal{U}}, \forall t_k \in [t_k, t_{k+N-1}] \\ & x(t_{k+N}) \in \mathcal{X}_f, \end{aligned}$$

and its result is the optimal control sequence $u^o_{[t_k,t_{k+N-1}]}$. Then, according to the Receding Horizon principle, the control input applied to the system is the first element of the optimal control sequence, denoted by $u_{\text{MPC}}(t) = u^o(t_k), \forall t \in [t_k, t_{k+1})$.

4.2.2 The DNN-SMO based MPC/ISM

Since the dynamics of system (4.1) is assumed unknown, it can be estimated using the BLF based DNN-SMO methodology, described in Chapter 3.3. In particular, the estimated transient variable $\hat{z}(\hat{x}(t))$, defined in (4.2), follows the dynamics

$$\dot{\hat{z}}(\hat{x}(t)) = C\Big[\hat{\Phi}_{k_\Phi} + \sum_{i=1}^{m} \hat{\Psi}^{[i]}_{k_\Psi}\, u_{\text{MPC},i}(t) - \dot{x}^\star(t)\Big],$$

with initial condition $\hat{z}(\hat{x}(0)) = \sigma_0(\hat{x}(0)) = \sigma_0(x(0))$. Furthermore, recalling Theorem 3.2, under its assumptions and selecting $\hat{\rho}$ as

$$\hat{\rho} > \frac{\bar{\lambda}(\hat{C})\{\bar{c}_\Phi + \bar{\varepsilon}_\Phi + m(\bar{c}_\Psi + \bar{\varepsilon}_\Psi)\|u\| + \bar{h}\} + \bar{\eta}}{\underline{\lambda}(\hat{C})},$$

imply that $\hat{\sigma}(x(t), \hat{x}(t)) \to 0_n$ for $t \to \infty$. Moreover, assuming $\hat{\sigma}(x(0), \hat{x}(0)) \in \Omega_\sigma$ it is guaranteed that $\|\hat{\sigma}(x(t), \hat{x}(t))\| \leq \varepsilon_\sigma$ for $t \geq 0$ by virtue of the BLF. Expanding $\hat{\sigma}$ it gets

$$\|\hat{\sigma}(x(t), \hat{x}(t))\| = \left\|\hat{C}\int_0^t (\dot{x}(\tau) - \dot{\hat{x}}(\tau))d\tau\right\| \leq \varepsilon_\sigma. \tag{4.7}$$

In the MPC context, it is useful to quantify how these bounds affect the discrepancy between the real dynamics and the internal prediction model employed by the controller. To this end, consider the following quantity

$$\left\|\hat{C}\int_0^{t_k} (\dot{x}(\tau) - \dot{\hat{x}}_{\text{MPC}}(\tau))d\tau\right\|. \tag{4.8}$$

The MPC prediction model is based on the DNNs estimates as

$$\dot{\hat{x}}_{\mathrm{MPC}}(t) = \dot{\hat{x}}(t) - \hat{\rho}\frac{\hat{\sigma}}{\|\hat{\sigma}\|}.$$

Substituting this expression into (4.8) it holds

$$\left\|\hat{C}\int_0^{t_k}\left(\dot{x}(\tau) - \dot{\hat{x}}(\tau) + \hat{\rho}\frac{\hat{\sigma}}{\|\hat{\sigma}\|}\right)d\tau\right\| = \left\|\hat{C}\int_0^{t_k}(\dot{x}(\tau) - \dot{\hat{x}}(\tau))d\tau + \hat{C}\int_0^{t_k}\hat{\rho}\frac{\hat{\sigma}}{\|\hat{\sigma}\|}d\tau\right\|.$$

Applying the triangle inequality, the first term can be directly bounded by exploiting the BLF property (4.7), obtaining

$$\left\|\hat{C}\int_0^{t_k}(\dot{x}(\tau) - \dot{\hat{x}}(\tau))d\tau + \hat{C}\int_0^{t_k}\hat{\rho}\frac{\hat{\sigma}}{\|\hat{\sigma}\|}d\tau\right\| \leq \varepsilon_\sigma + \left\|\hat{C}\int_0^{t_k}\hat{\rho}\frac{\hat{\sigma}}{\|\hat{\sigma}\|}d\tau\right\|.$$

For the second term, exploiting the sub multiplicative property of the norm and that $\frac{\hat{\sigma}}{\|\hat{\sigma}\|}$ is a unit vector, it gets

$$\begin{aligned}\left\|\hat{C}\int_0^{t_k}(\dot{x}(\tau) - \dot{\hat{x}}(\tau))d\tau + \hat{C}\int_0^{t_k}\hat{\rho}\frac{\hat{\sigma}}{\|\hat{\sigma}\|}d\tau\right\| &\leq \varepsilon_\sigma + \|\hat{C}\|\left\|\int_0^{t_k}\hat{\rho}\frac{\hat{\sigma}}{\|\hat{\sigma}\|}d\tau\right\| \\ &\leq \varepsilon_\sigma + \bar{\lambda}(\hat{C})\int_0^{t_k}\left\|\hat{\rho}\frac{\hat{\sigma}}{\|\hat{\sigma}\|}\right\|d\tau \\ &\leq \varepsilon_\sigma + \bar{\lambda}(\hat{C})\int_0^{t_k}\hat{\rho}d\tau \\ &\leq \varepsilon_\sigma + \bar{\lambda}(\hat{C})\hat{\rho}k\Delta t,\end{aligned}$$

where $\Delta t \in \mathbb{R}_{>0}$ is the sampling time of the MPC. Recalling the original left side of the inequality it has

$$\left\|\hat{C}\int_0^{t_k}(\dot{x}(\tau) - \dot{\hat{x}}_{\mathrm{MPC}}(\tau))d\tau\right\| \leq \varepsilon_\sigma + \bar{\lambda}(\hat{C})\hat{\rho}k\Delta t.$$

Finally, since $\hat{C}$ is symmetric positive definite, its minimum eigenvalue $\underline{\lambda}(\hat{C})$ provides a lower bound on the norm of the overall expression

$$\left\|\hat{C}\int_0^{t_k}(\dot{x}(\tau) - \dot{\hat{x}}_{\mathrm{MPC}}(\tau))d\tau\right\| \geq \underline{\lambda}(\hat{C})\left\|\int_0^{t_k}(\dot{x}(\tau) - \dot{\hat{x}}_{\mathrm{MPC}}(\tau))d\tau\right\|.$$

Combining these bounds yields an explicit inequality relating the integrated prediction error to the design parameters $\varepsilon_\sigma, \hat{C}, \Delta t$ and, $\hat{\rho}$

$$\begin{aligned}\underline{\lambda}(\hat{C})\left\|\int_0^{t_k}(\dot{x}(\tau) - \dot{\hat{x}}_{\mathrm{MPC}}(\tau))d\tau\right\| &\leq \varepsilon_\sigma + \bar{\lambda}(\hat{C})\hat{\rho}k\Delta t \\ \left\|\int_0^{t_k}(\dot{x}(\tau) - \dot{\hat{x}}_{\mathrm{MPC}}(\tau))d\tau\right\| &\leq \underbrace{\frac{\varepsilon_\sigma}{\underline{\lambda}(\hat{C})} + \frac{\bar{\lambda}(\hat{C})}{\underline{\lambda}(\hat{C})}\hat{\rho}k\Delta t}_{=:\varsigma},\end{aligned}$$

with bound $\varsigma \in \mathbb{R}_{>0}$ composed of two terms. The first quantifies the BLF effect on the observer error, providing time invariant (constant) estimation error bound.

The second represents the SMO correction action, compensating the DNNs residual approximation errors, with magnitude proportional to the integration interval, meaning that longer horizons yield larger worst-case prediction errors. A graphical representation of this effect is shown in Figure 4.2. The above result imply that,

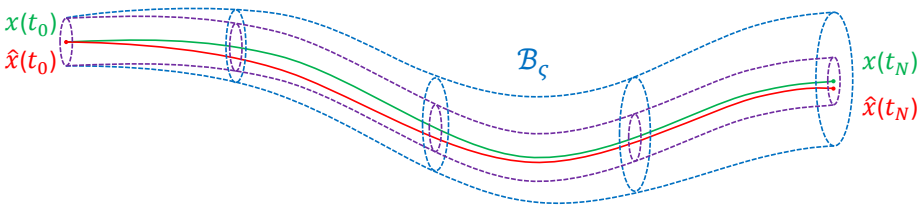

Figure 4.2: Example of error tube along the prediction time. The lines associated with x, $\hat{x}_{\text{MPC}}$ are in green and red, respectively. The violet tube represents the constant error bound due to the BLF constraint on the observer sliding variable, while the blue tube represents the total error bound, which expands linearly along the prediction horizon to account for the SMO correction action accumulation.

in the worst case, the DNNs estimation error is bounded by ς for x, enabling a reduced constraints set $\bar{\mathcal{X}}_c \subset \mathcal{X}_c$, obtained by tightening $\mathcal{X}_c$ with boundary ς in the x direction via Pontryagin difference

$$\bar{\mathcal{X}}_c \coloneqq \mathcal{X}_c \ominus \mathcal{B}_\varsigma, \tag{4.9}$$

with $\mathcal{B}_\varsigma \coloneqq \{x \in \mathcal{X}_c : \|x\| \leq \varsigma\}$. Then, the DNN-based MPC problem can be formulated as

$$\begin{aligned}
\min_{u_{[t_k,t_{k+N-1}]}} \quad & J(\hat{x}_{\text{MPC}}(t_k), u_{[t_k,t_{k+N-1}]}, N) \\
\text{s.t.} \quad & \dot{\hat{x}}_{\text{MPC}} = \hat{\Phi}_{k_\Phi} + \sum_{i=1}^{m} \hat{\Psi}^{[i]}_{k_\Psi} u_i, \\
& \hat{x}_{\text{MPC}}(t_k) = x(t_k), \\
& \hat{x}_{\text{MPC}}(t_k) \in \bar{\mathcal{X}}_c, \ \forall t_k \in [t_k, t_{k+N-1}] \\
& u(t_k) \in \bar{\mathcal{U}}, \forall t_k \in [t_k, t_{k+N-1}] \\
& \hat{x}_{\text{MPC}}(t_{k+N}) \in \mathcal{X}_f,
\end{aligned} \tag{4.10}$$

where the cost J and the terminal set $\mathcal{X}_f$ are chosen as in (4.5) and (4.6), respectively. Then, similarly as in the ideal case, the control input is chosen as the first

element of the optimal control sequence $u^o_{[t_k,t_{k+N-1}]}$, denoted as $u^o(t_k)$, resulting from (4.10), i.e.,

$$u_{\text{MPC}}(t) = u^o(t_k), \forall t \in [t_k, t_{k+1}). \tag{4.11}$$

4.2.3 Event-triggered MPC-internal DNNs Update

To avoid continuous changes in the MPC model due to networks update, the internal DNNs are maintained as frozen copies of the DNN-SMO networks. This approach naturally leads to an event-triggering condition, inspired by [36], given by

$$\left\| \int_{t_k}^{t_{k+1}} (\dot{x}(\tau) - \dot{\hat{x}}_{\text{MPC}}(\tau)) d\tau \right\| > \delta, \tag{4.12}$$

with $\delta \in \mathbb{R}_{>0}$ chosen such that $\delta \leq \varepsilon_\sigma / \underline{\lambda}(\hat{C})$, ensuring the boundedness of the prediction error within $\mathcal{B}_\varsigma$. If condition (4.12) holds, it means that the MPC prediction model is diverging from the real dynamics. Therefore, the MPC-internal DNNs are updated with the latest weights from the DNN-SMO networks. Moreover, even if in practice the triggering condition cannot be verified in continuous-time, a faster sampling rate than the one used in the MPC is adopted, ensuring that the MPC sequence $u^o_{[t_k,t_{k+N-1}]}$ remains valid until t_{k+1}.

The effectiveness of the proposed DNN-SMO based MPC/ISM scheme has been validated through simulations on the Duffing oscillator and a 3-DoF robotic manipulator. Results are presented in Chapter 5.

Chapter 5

Simulations and Results

The proposed DNN-SMO methodology is validated through detailed simulation on two benchmark systems, a Duffing oscillator and a 3-DoF robotic manipulator, highlighting the observer convergence properties, the impact of the BLF-based adaptation laws on transient performance, and the effective controller design enabled by accurate DNN dynamics approximation. Since both systems exhibit the partitioned position-velocity structure, detailed in Section 1.1.2, and exploiting the known the relationship between position and velocity, the DNNs are employed solely to approximate the velocity dynamics. In particular, $\hat{\Phi} : \mathcal{X} \to \mathbb{R}^m$ approximates the drift dynamics $f_2 : \mathcal{X} \to \mathbb{R}^m$, while $\hat{\Psi} : \mathcal{X} \to \mathbb{R}^{m \times m}$ approximates the control effectiveness matrix $\bar{B} : \mathcal{X} \to \mathbb{R}^{m \times m}$.

5.1 Duffing Oscillator

Depicted in Figure 5.1, the Duffing oscillator is a nonlinear damped-driven mechanical oscillator modeled as follows

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ x_1 - x_1^3 - x_2 + u \end{bmatrix}, \tag{5.1}$$

where $x \in \mathcal{X} \subset \mathbb{R}^2$ is the state vector, with x_1 and x_2 being the position and velocity, respectively, and $u \in \mathbb{R}$ is the control input. The nonlinearity arises from the cubic stiffness term $-x_1^3$, which renders the system ideal for testing the proposed DNN-SMO scheme under the assumption of fully unknown dynamics. Three simulations have been carried out, analyzing the behavior of the system in the case of the pure DNN-SMO based ISM compared with the BLF-modified version, and in the case of the control solution with MPC/ISM. In all scenarios, the DNNs $\hat{\Phi}$ and $\hat{\Psi}$ are configured with $k_\Phi = k_\Psi = 2$ hidden layers, each characterized by 16 neurons,

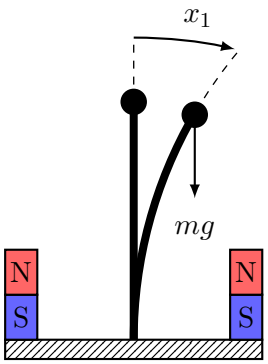

Figure 5.1: Graphical representation of the Duffing oscillator system.

all activated via tanh functions. Weights adaptation is modulated through learning rate matrices $\Gamma_{\Phi_j} = 5\ I, \Gamma_{\Psi_j} = I$, with I denoting an identity matrix of appropriate size.

5.1.1 DNN-SMO based ISM

This 40 seconds simulation aims to steer the system state vector to time-varying setpoints $x^\star = [1.25 \quad 0]^\top$ for $t \in [0, 20)$ s and $x^\star = [-0.25 \quad 0]^\top$ for $t \geq 20$ s, while being subject to matched disturbance $h = 0.1 \sin(t)$. The stabilizing control law is defined as

$$u_n(t) = -\frac{1}{\hat{\Psi}_{k_\Psi} + \epsilon} \left\{ \hat{\Phi}_{k_\Phi} + \begin{bmatrix} 0.5 & 1.5 \end{bmatrix} (x(t) - x^\star(t)) \right\}, \tag{5.2}$$

where $\epsilon \in \mathbb{R}$ is a small design parameter introduced to avoid singularity of the denominator. The conventional sliding variable is chosen as $\sigma_0 = (x_1 - x_1^\star) + (x_2 - x_2^\star)$ with control gain $\rho = 1$. As for the observer, $\hat{C} \in \mathbb{R}^{2\times 2}$ in (3.4) is chosen as $\hat{C} = I_2$ with observer gain $\hat{\rho} = 0.005$. The simulation time-step is set to 10^{-4} seconds. Results in Figure 5.2 demonstrate that, even applying a small observer gain, the observer sliding variable always remains limited and, in particular, converge to near-zero values (and theoretically to zero), indicating an accurate approximation of the system dynamics by the DNNs, confirmed by the integral sliding variable, which is also successfully driven to a very small value (and eventually zero) after an initial transient due to the first DNNs weights adaptation, preventing the transient variable $\hat{z}$ from fully compensating σ_0. Moreover, the system states are successfully controlled to the desired time-varying setpoints. To further enhance transient performance, the BLF-modified version is employed, setting the BLF limit set as $\varepsilon_\sigma = 0.2$. Figure 5.3 shows a dramatic performance improvement: the observer sliding variable norm remains strictly bounded and well below the threshold, staying nearly

zero throughout the simulation, ensuring that the observer and real system states effectively overlap while reducing transient overshoot by one order of magnitude.

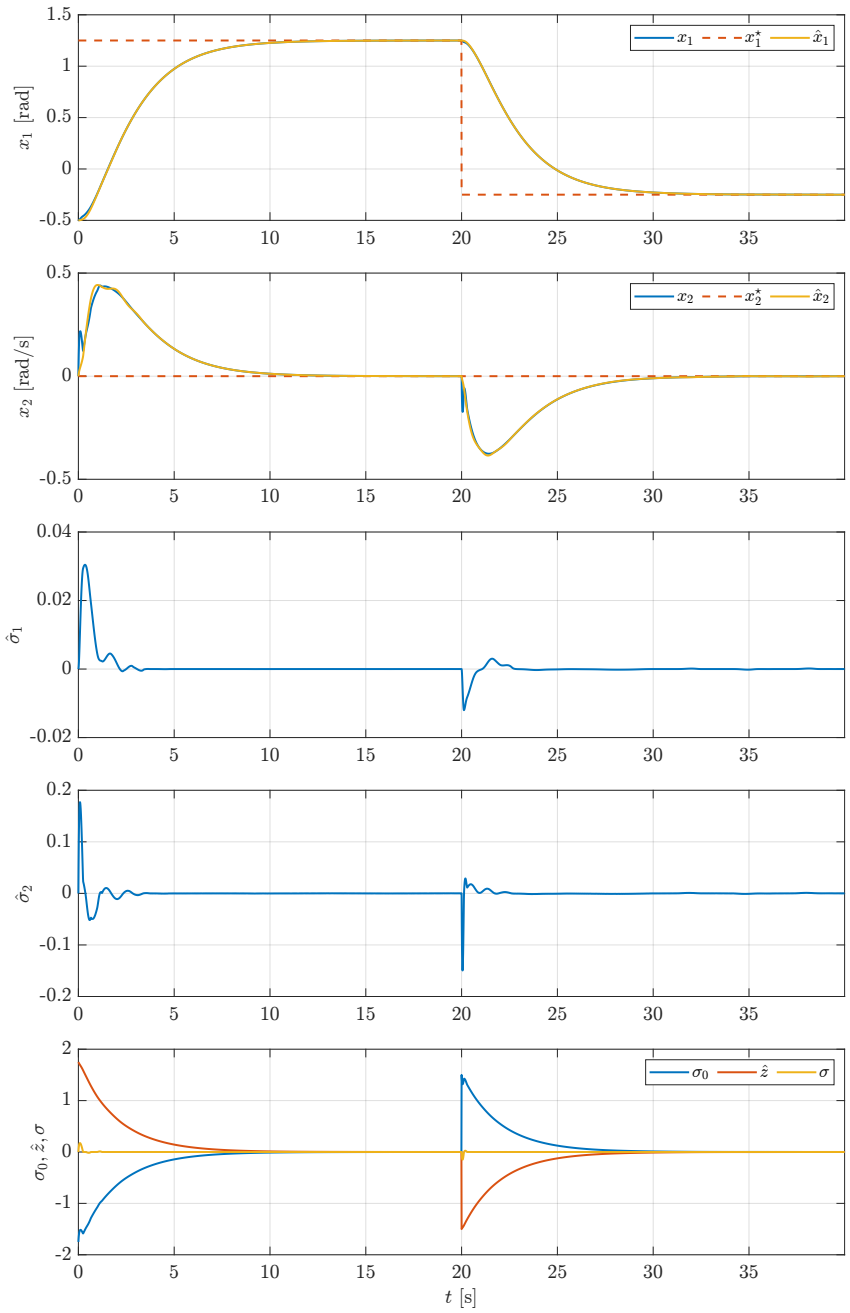

Figure 5.2: Duffing oscillator (DNN-SMO based ISM): time evolution of the system states (first and second rows), observer sliding variable components (third and fourth rows), and integral sliding variable (last row).

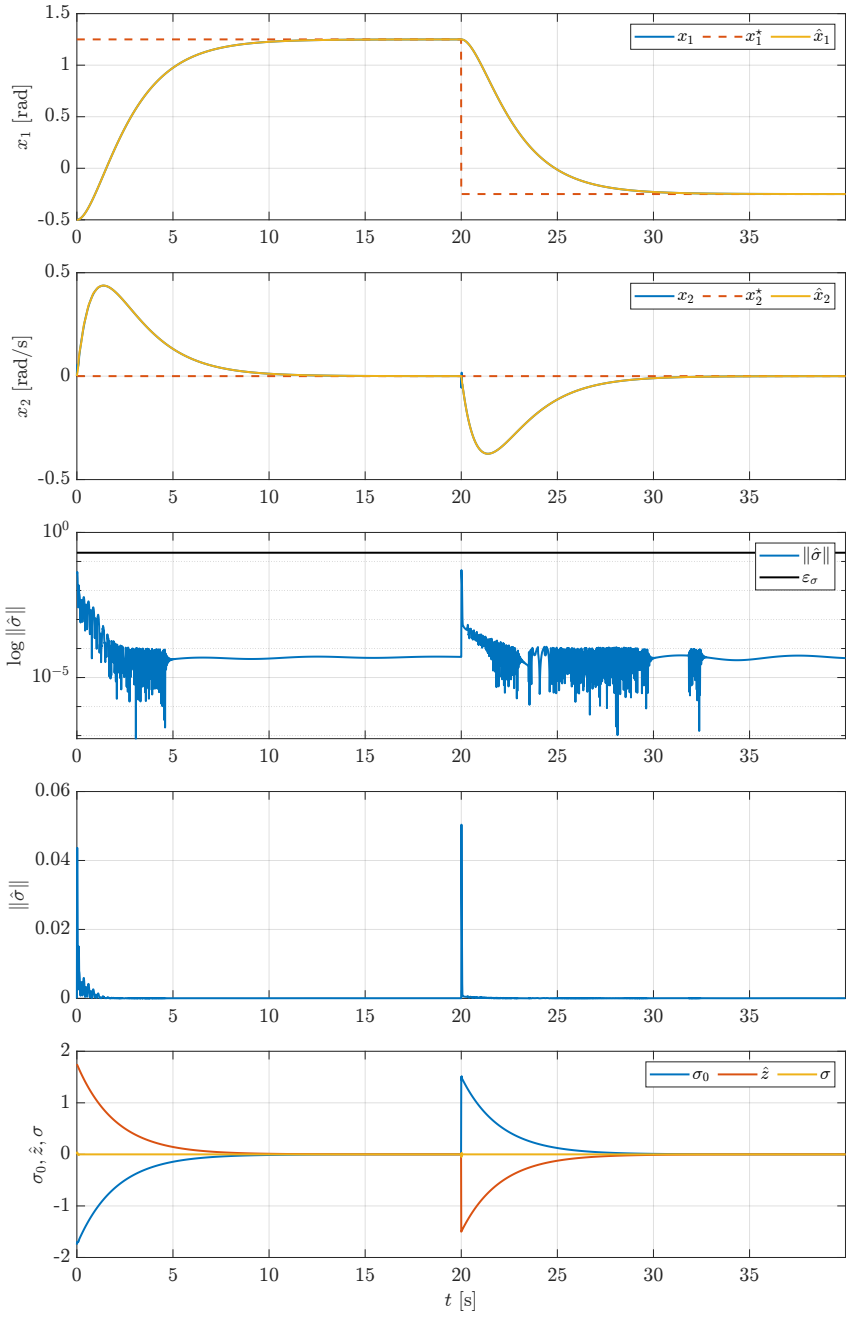

Figure 5.3: Duffing oscillator (DNN-SMO based ISM with BLF): time evolution of the system states (first and second rows), observer sliding variable norm in semi-logarithmic (third row) and linear scale (fourth row), and integral sliding variable (last row). The BLF limit is depicted with black solid line.

5.1.2 DNN-SMO based MPC/ISM

Consider the Duffing oscillator dynamics in (5.1), subject to state and input constraints defined as $\mathcal{X}_c \coloneqq [-1.8, 1.8] \times [-1.57, 1.57]$ and $\mathcal{U} \coloneqq \{u \in \mathbb{R} : |u| \leq 10\}$, respectively. This 30 seconds simulation aims to steer the system state vector to time-varying setpoints chosen as $x^\star = [0.5 \quad 0]^\top \in \mathcal{X}_c$ for $t \in [0, 10)$ seconds, $x^\star = [1.95 \quad 0]^\top \notin \mathcal{X}_c$ for $t \in [10, 20)$ seconds, and $x^\star = [0.35 \quad 0]^\top \in \mathcal{X}_c$ for $t \geq 20$ seconds, while subject to a matched disturbance $h(t) = 0.1 \sin(t)$. The control law follows (4.4), where the MPC employs a quadratic cost with $Q = 10\ I_2$ and $R = 0.1$, prediction horizon $N = 10$, and discretization step $t_{k+1} - t_k = 0.1$ seconds. The control gain has been chosen as $\rho = 0.2$. As for the observer parameters are chosen as in Section 5.1.1, while the BLF limit is set as $\varepsilon_\sigma = 0.4$. The simulation time-step is set to 10^{-3} seconds. The results of the simulation are presented in Figure 5.4, in which the time evolution of state trajectories, constrained control input, observer sliding variable norm, and sliding variable demonstrate the effectiveness of the proposed scheme to approximate the system dynamics enabling solution of the constrained optimization problem although the system model is completely unknown. Notably, during $t \in [10, 20)$ seconds, when $x_1^\star = 1.95 \notin \mathcal{X}_c$, the state $x_1(t)$ remains inside $\mathcal{X}_c$, well below the upper limit thanks to the constraint tightening (4.9) preventing constraint violation even for infeasible setpoints.

5.2 Robot Manipulator

Building upon the scalar Duffing case, a 3-DoF robotic manipulator, depicted in Figure 5.5, is considered, to validate the proposed schemes for multi-dimensional systems. Its dynamics, derived from Euler-Lagrange (EL) modeling [37], follows the canonical reduced form

$$\dot{x}(t) = \begin{bmatrix} \dot{q} \\ -M(q)^{-1} C((q, \dot{q})\dot{q} + g(q)) + M^{-1}(q)\tau \end{bmatrix},$$

where the state vector is $x = [q^\top \quad \dot{q}^\top]^\top \in \mathcal{X} \subset \mathbb{R}^6$, with $q \in \mathcal{X} \subset \mathbb{R}^3$ being the joint variables, also called generalized coordinates, $f_1 = \dot{q} \in \mathbb{R}^3$, $f_2 = -M(q)^{-1}\left(C(q, \dot{q})\dot{q} - g(q)\right) \in \mathbb{R}^3, \bar{B} = M(q)^{-1} \in \mathbb{R}^{3\times 3}$, and $u = \tau \in \mathbb{R}^3$. Since f_1 can be obtained directly from robot sensors measurements and $\bar{B}$ is assumed known for simulation purposes, only f_2 requires estimation. In particular, it has been employed the DNN $\hat{\Phi} : \mathcal{X} \to \mathbb{R}^3$ characterized by $k_\Phi = 3$ hidden layers with 16 neurons each and learning rate matrix $\Gamma_{\Phi_j} = 10\ I$.

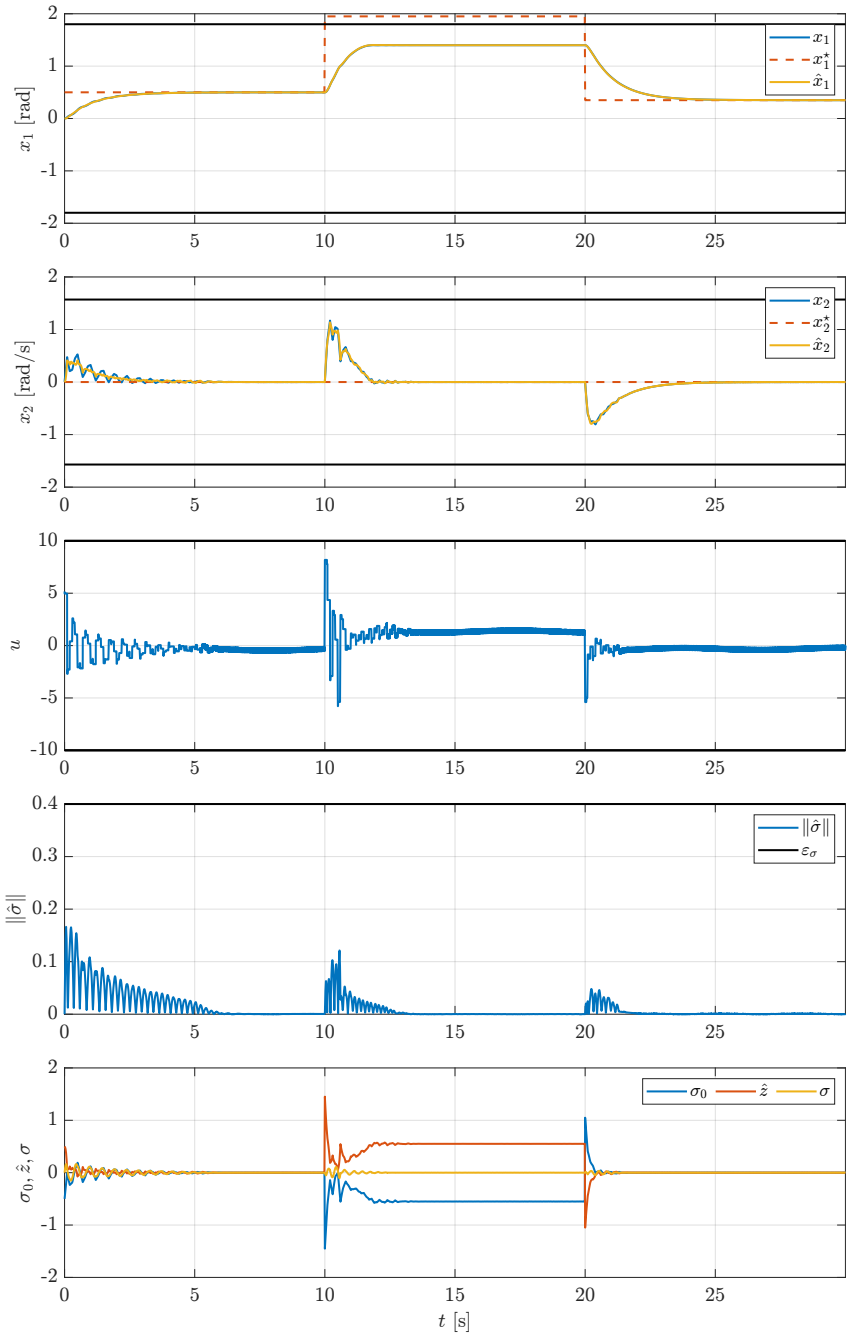

Figure 5.4: Duffing oscillator (DNN-SMO based MPC/ISM): time evolution of the system states (first and second rows), control input (third row), observer sliding variable norm (fourth row), and integral sliding variable (last row). The state and input constraints $\mathcal{X}_c$ and $\mathcal{U}$ and the BLF limit are depicted with black solid lines.

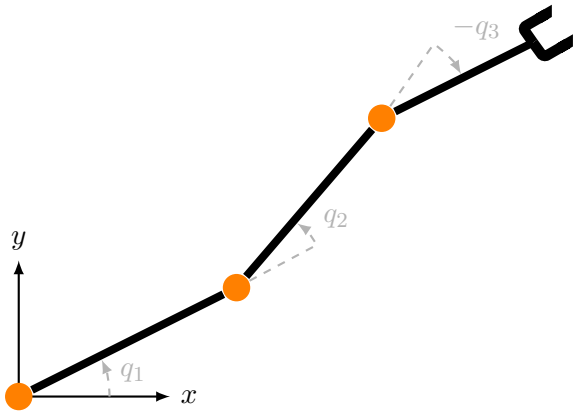

Figure 5.5: Graphical representation of the 3-DoF robotic manipulator.

5.2.1 DNN-SMO based ISM

This 20 seconds simulation aims to track the desired time-varying joint space trajectory

$$x^\star(t) = \begin{cases} \begin{bmatrix} \frac{\pi}{9} & \frac{\pi}{9} & -\frac{\pi}{18} & 0 & 0 & 0 \end{bmatrix}^\top & \text{if } t \in [0, 10) \text{ s} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^\top & \text{if } t \geq 10 \text{ s}, \end{cases}$$

while being subject to a matched disturbance

$$h(t) = \begin{bmatrix} 0.1 \sin(2t) & 0 & 0.05 \cos(2t) \end{bmatrix}^\top.$$

The stabilizing control law is defined as

$$u_n(t) = -\bar{B}^{-1} \left\{ \hat{\Phi}_{k_\Phi} + K_p \left(q(t) - q^\star(t) \right) + K_d \left(\dot{q}(t) - \dot{q}^\star(t) \right) \right\},$$

where the gain matrices $K_p = K_d = 10\ I_3$. The conventional sliding variable is chosen as $\sigma_0 = C_1(x_1 - x_1^\star) + C_2(x_2 - x_2^\star)$, with matrices $C_1 \in \mathbb{R}^3$ and $C_2 \in \mathbb{R}^3$ equal to the identity matrix and control gain $\rho = 0.2$. As for the observer, $\hat{C} \in \mathbb{R}^3$, appearing in (3.4), is chosen as $\hat{C}_1 = I_3$, while the observer gain is set as $\hat{\rho} = 0.05$. The simulation time-step is set to 10^{-4}. The simulation results, shown in Figures 5.6-5.8, confirm that, even for more complex multi-dimensional dynamics, both observer sliding variable and integral sliding variable converge to near-zero values, demonstrating accurate approximation of the system dynamics. Moreover, the reference tracking objective is successfully achieved.

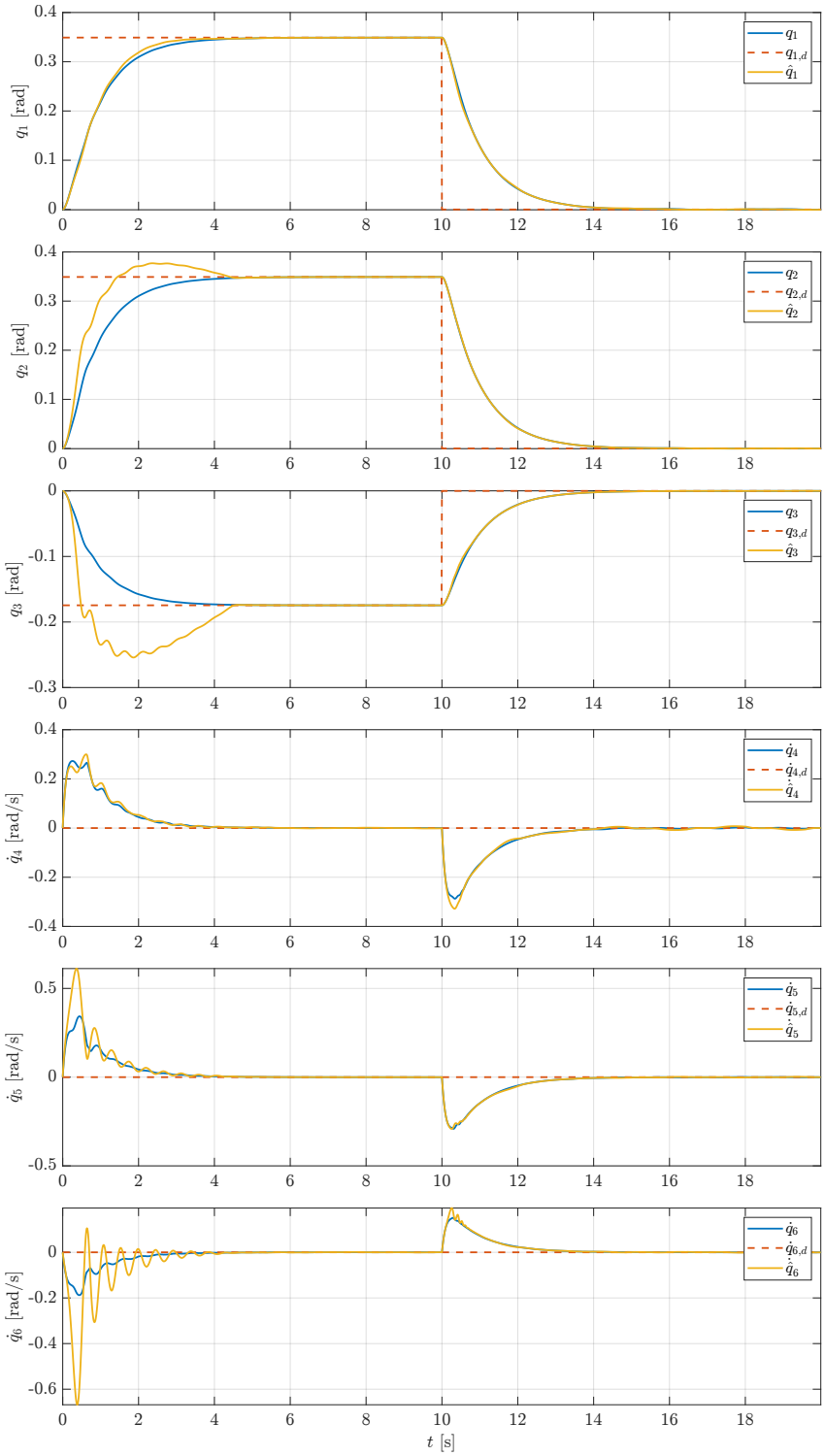

Figure 5.6: Robot (DNN-SMO based ISM): time evolution of the system states.

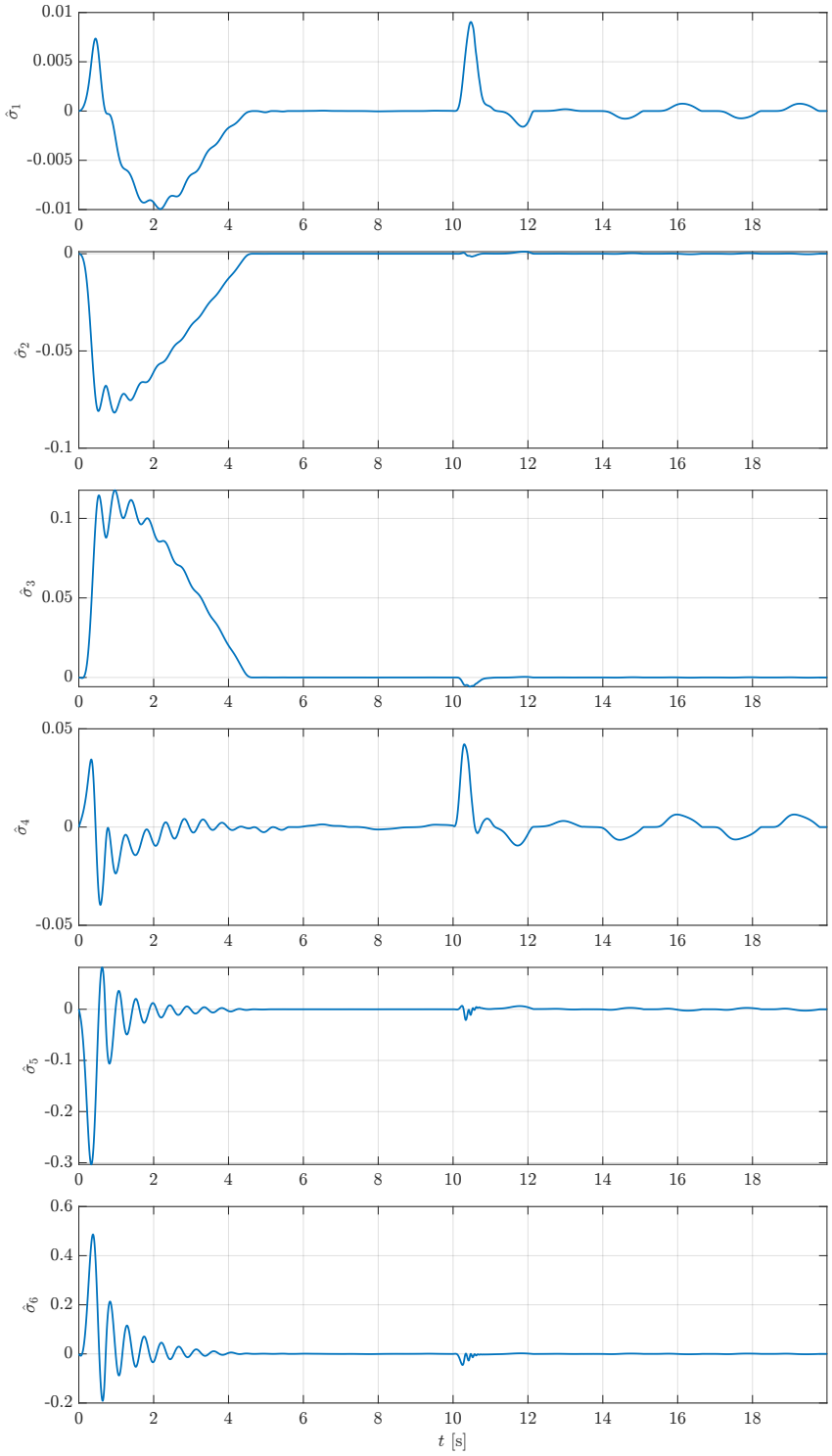

Figure 5.7: Robot (DNN-SMO based ISM): time evolution of the observer sliding variable components.

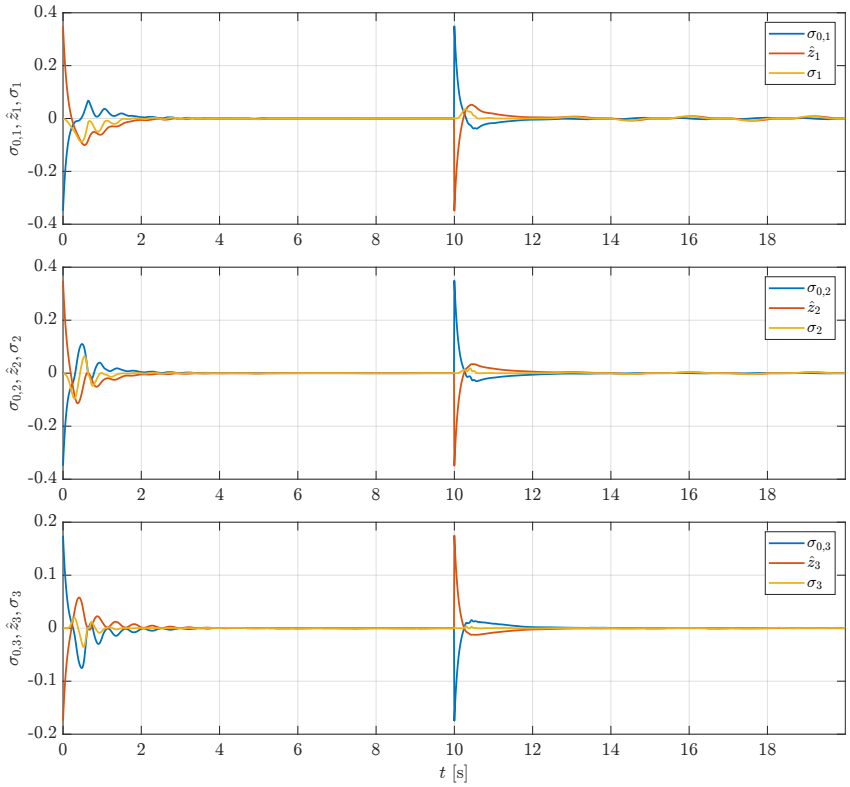

Figure 5.8: Robot (DNN-SMO based ISM): time evolution of the integral sliding variables.

Applying then the BLF-modified version, with $\varepsilon_\sigma = 0.1$, both the transient performance and the approximation accuracy are significantly enhanced as shown in Figures 5.9-5.11, where real and estimated states overlap almost perfectly, confirmed by the observer sliding variable norm always of order 10^{-3}.

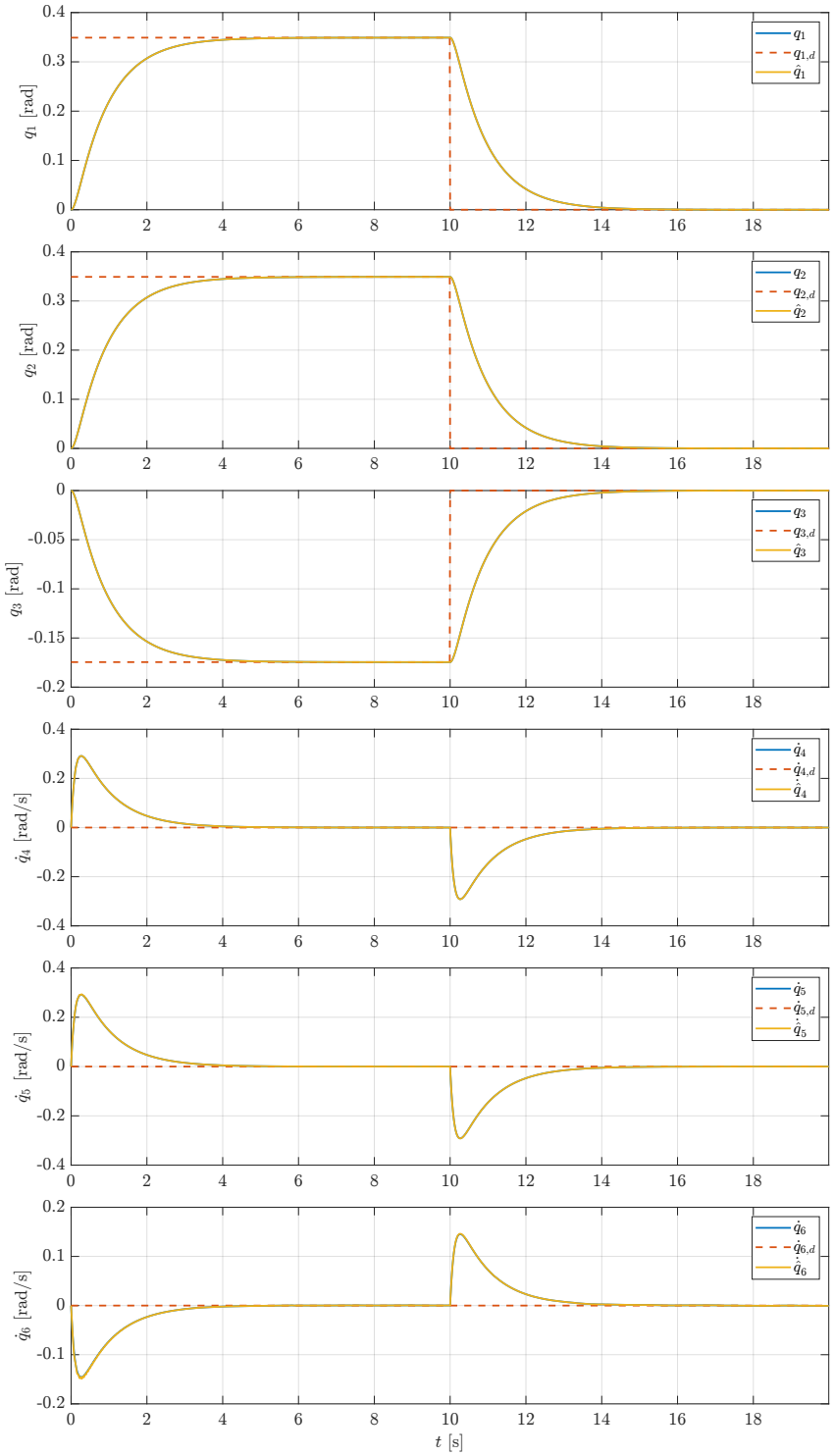

Figure 5.9: Robot (DNN-SMO based ISM with BLF): time evolution of the system states.

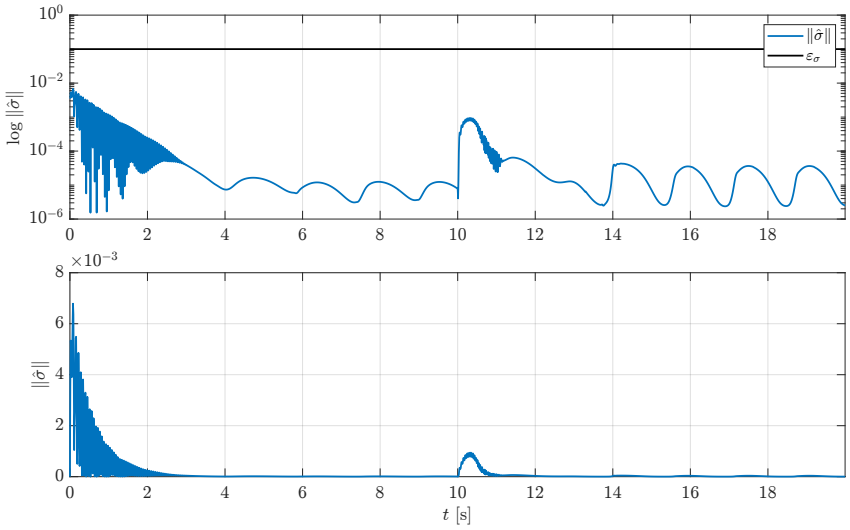

Figure 5.10: Robot (DNN-SMO based ISM with BLF): time evolution of the observer sliding variable norm in semi-logarithmic (first row) and linear scale (second row). The BLF limit is depicted with black solid line.

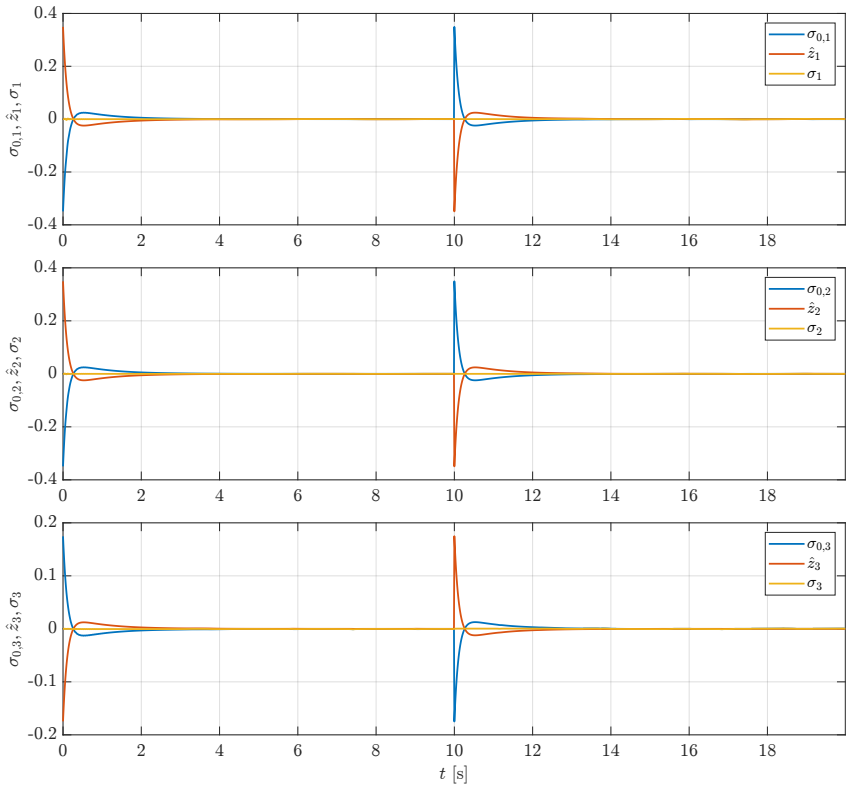

Figure 5.11: Robot (DNN-SMO based ISM with BLF): time evolution of the integral sliding variables.

5.2.2 DNN-SMO based MPC/ISM

Consider now the constrained case, with state and input constraints defined as

$$\mathcal{X}_c \coloneqq \begin{cases} (x_1, x_4) \in \left[-\frac{\pi}{6}, \frac{\pi}{6} \right] \times \left[-\frac{\pi}{2}, \frac{\pi}{2} \right] \\ (x_2, x_5) \in \left[-\frac{\pi}{6}, \frac{\pi}{6} \right] \times \left[-\frac{\pi}{2}, \frac{\pi}{2} \right] \\ (x_3, x_6) \in \left[-\frac{\pi}{9}, \frac{\pi}{9} \right] \times \left[-\frac{\pi}{2}, \frac{\pi}{2} \right], \end{cases}$$

$$\mathcal{U} \coloneqq \{u \in \mathbb{R}^3 : |u_1| \leq 50, |u_2| \leq 25, |u_3| \leq 5\},$$

respectively. The 30 seconds simulation aims to track the desired time-varying joint space trajectory

$$x^\star(t) = \begin{cases} \begin{bmatrix} \frac{\pi}{9} & \frac{\pi}{9} & -\frac{\pi}{18} & 0 & 0 & 0 \end{bmatrix}^\top \in \mathcal{X}_c & \text{if } t \in [0, 10) \text{ s} \\ \begin{bmatrix} \frac{7}{36}\pi & \frac{\pi}{9} & -\frac{5}{36}\pi & 0 & 0 & 0 \end{bmatrix}^\top \notin \mathcal{X}_c & \text{if } t \in [10, 20) \text{ s} \\ \begin{bmatrix} \frac{\pi}{12} & \frac{\pi}{12} & 0 & 0 & 0 & 0 \end{bmatrix}^\top \in \mathcal{X}_c & \text{if } t \geq 20 \text{ s}, \end{cases}$$

while subject to a matched disturbance

$$h(t) = \begin{bmatrix} 0.1 \sin(2t) & 0 & 0.05 \cos(2t) \end{bmatrix}^\top.$$

The control law follows (4.4), where the MPC employs a quadratic cost with $Q = \text{diag}\{5000, 5000, 5000, 10000, 10000, 10000\}$ and $R = I_3$, prediction horizon $N = 10$, and discretization step $t_{k+1} - t_k = 0.1$ seconds. The control gain has been chosen as $\rho = 0.1$. As for the observer parameters and BLF limit are chosen as in Section 5.2.1. The simulation time-step is set to 10^{-3} seconds. The results in Figures 5.12-5.15 confirm the effectiveness of the proposed architecture even for multi-dimensional systems. Observer sliding variable norm on the order of 10^{-3} demonstrates accurate dynamics approximation, also underlined by the integral sliding variables consistently near zero, enabling constrained optimization despite model uncertainties. The scheme successfully tracks setpoints while respecting constraints, even for infeasible references.

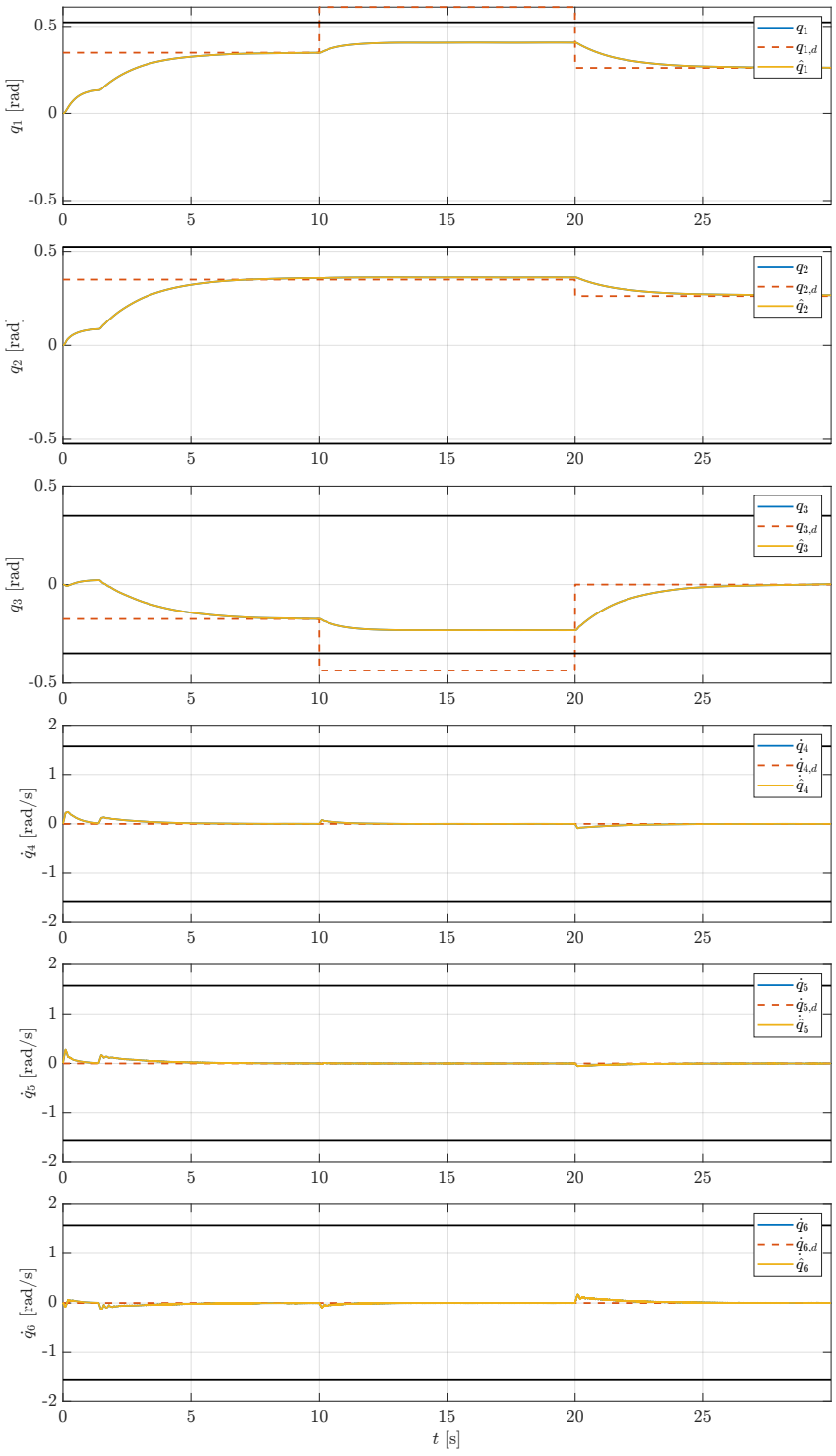

Figure 5.12: Robot (DNN-SMO based MPC/ISM): time evolution of the system states. The state constraints $\mathcal{X}_c$ are depicted with black solid lines.

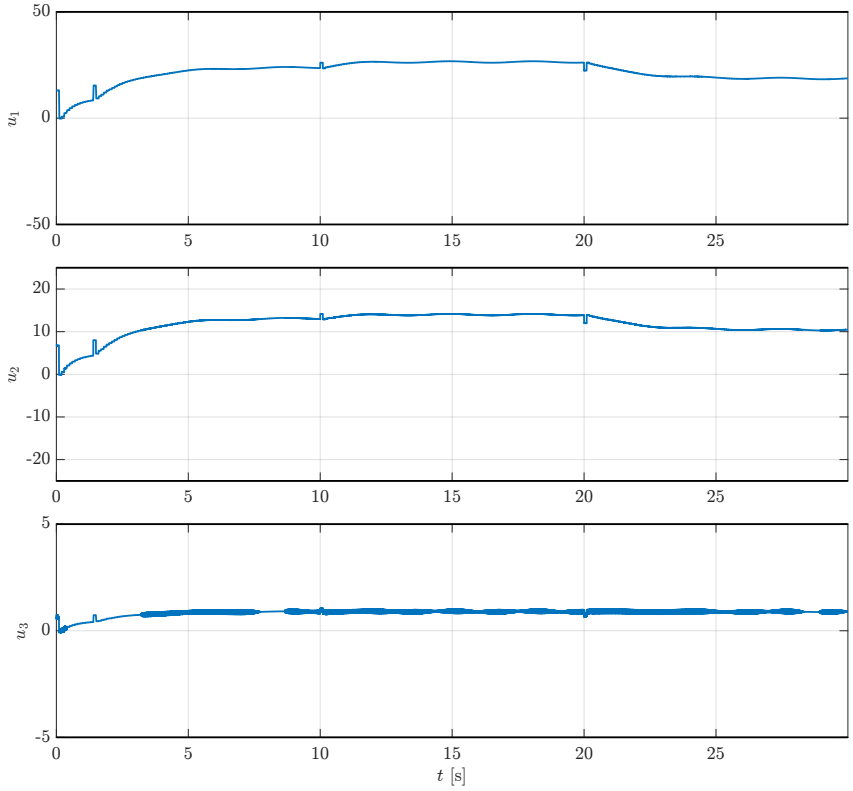

Figure 5.13: Robot (DNN-SMO based MPC/ISM): time evolution of the control inputs. The input constraints $\mathcal{U}$ are depicted with black solid lines.

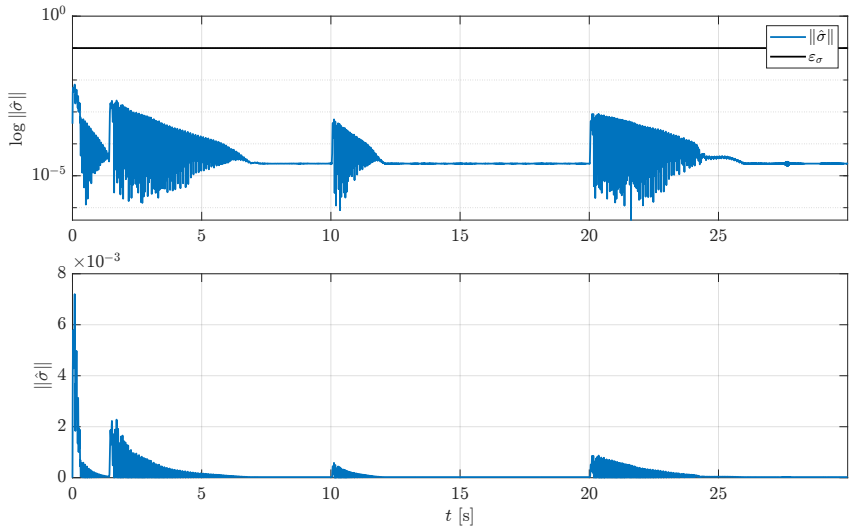

Figure 5.14: Robot (DNN-SMO based MPC/ISM): time evolution of the observer sliding variable norm in semi-logarithmic (first row) and linear scale (second row). The BLF limit is depicted with black solid line.

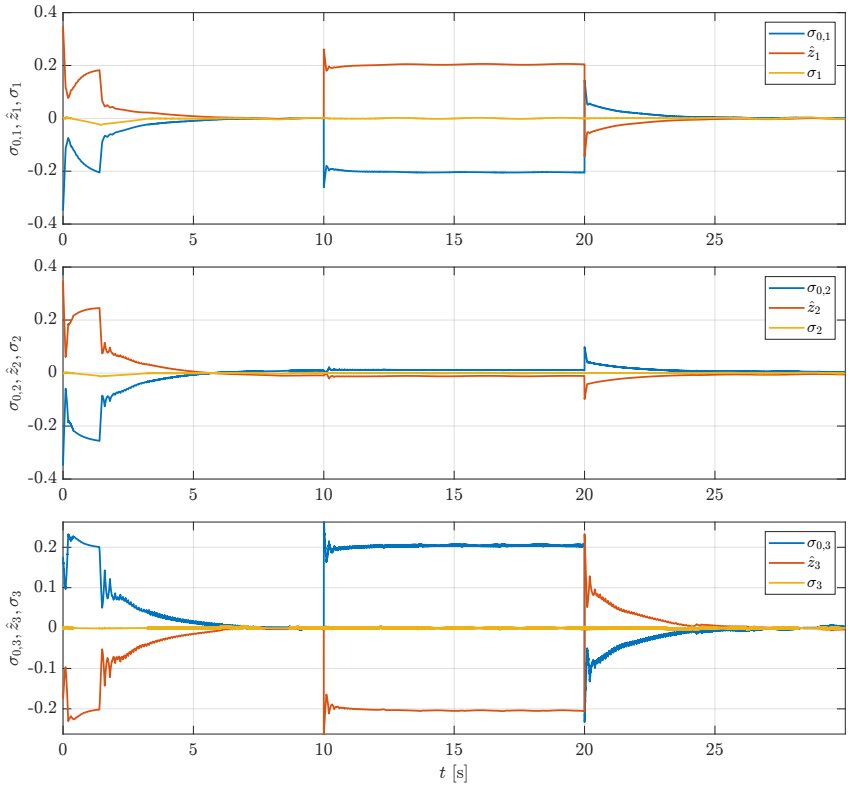

Figure 5.15: Robot (DNN-SMO based MPC/ISM): time evolution of the integral sliding variables.

5.3 Impact of Design Parameters on Performance

After having shown the results of the proposed methodology, this section investigates its sensitivity to key design parameters. For simulation purposes, the analysis is carried out on the Duffing oscillator, defined previously, controlled via DNN-SMO based ISM to steer the system state vector to $x^\star = [1.25 \quad 0]^\top$ while being subject to matched disturbance $h = 0.1\sin(t)$ and with nominal control law defined as in (5.2), by varying one parameter at a time and monitoring the norm of the observer error, the norm of the DNN-state estimation error, calculated as $\|x(t) - \hat{x}_{\text{DNN}}(t)\|$, with $\hat{x}_{\text{DNN}} \in \mathcal{X}$ being the state vector evolved according to the solely DNN dynamics, and the ISM sliding variable, taken as indices of the approximation accuracy. The simulation time-step is set to 10^{-4} unless otherwise specified.

5.3.1 Hidden Layers vs. Neurons

As stated at the beginning of Section 2.3, a key choice for DNNs concerns the selection of the number of hidden layers k, called depth, and the neuron count per layer L_k, known as width. In this regard, different network configurations are compared, to assess how depth and width affect approximation accuracy. In Figure 5.16, the depth is fixed $k_\Phi = k_\Psi = 2$, while the width is varied among $8, 16$, and, 32 neurons per layer. Conversely, in Figure 5.17, the width is fixed to 16 neurons per layer and the depth is varied by considering $1, 2$, and 3 hidden layers. In both cases, increasing either the width or the depth leads to a reduction of the observer error norm, the DNN-state estimation error norm, and a faster convergence of the sliding variable, confirming that richer network parametrization improve the approximation capabilities. To further compare depth and width, Figure 5.18 contrasts two network parameterizations: a shallow architecture with one hidden layer of 16 neurons and a deeper architecture with three hidden layers of 8 neurons each. The deeper network achieves lower estimation errors, leading to an almost ideal dynamics estimation, as confirmed by the sliding variable being steered to zero from the very first time instants, thus confirming that increasing depth is more beneficial than merely increasing width, as stated in Section 2.3.

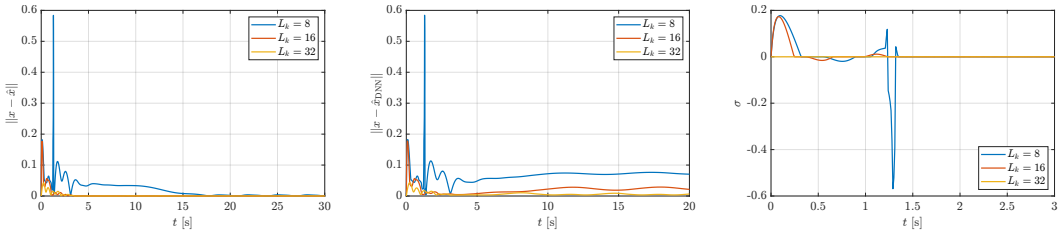

Figure 5.16: Impact of L_k on DNNs approximation accuracy. Starting from the left it has: the observer error norm, the DNN-state estimation error norm, and the sliding variable.

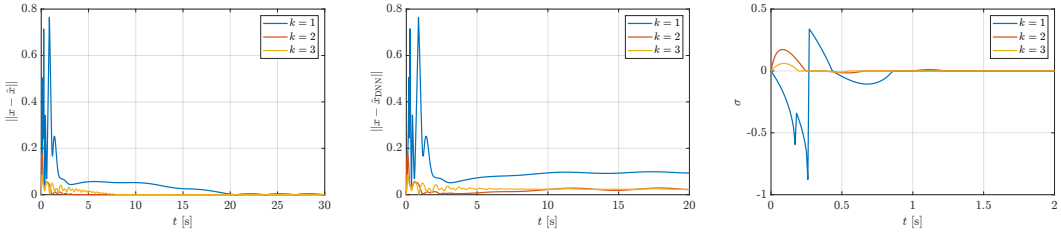

Figure 5.17: Impact of k on DNNs approximation accuracy. Starting from the left it has: the observer error norm, the DNN-state estimation error norm, and the sliding variable.

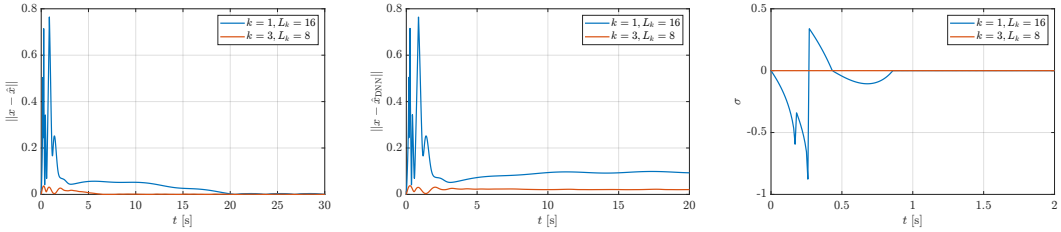

Figure 5.18: Depth vs. width comparison. Starting from the left it has: the observer error norm, the DNN-state estimation error norm, and the sliding variable.

5.3.2 The Learning Rates

Another DNNs key design parameter is the learning rate matrices. As shown in Figure 5.19, increasing it, improves the convergence speed of both the observer error norm and DNN-state estimation error norm, yielding faster convergence of the sliding variable. However, excessively large values lead to instability and even divergence.

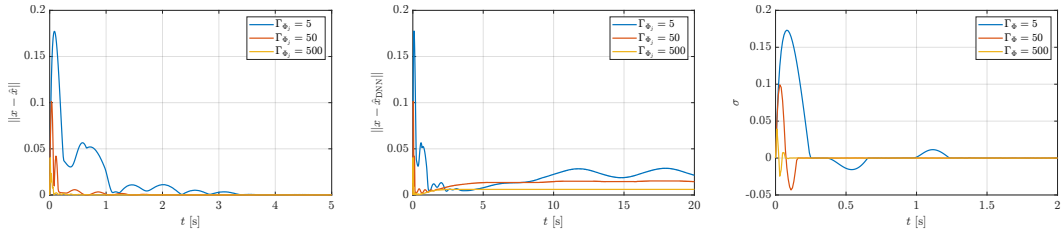

Figure 5.19: Impact of Γ_{Φ_j} on DNNs approximation accuracy. Starting from the left it has: the observer error norm, the DNN-state estimation error norm, and the sliding variable.

5.3.3 The Observer Gain

Let now examine the impact of the observer gain $\hat{\rho}$. Figure 5.20 shows the simulation results in three scenarios: $\hat{\rho} = 0.05, 0.005$, and 0.0005. As $\hat{\rho}$ increases, the SMO estimation error norm reduces, but at the same time the DNN-state estimation error norm grows. This behavior can be explained by the fact that, for high observer gain, a significant portion of the unknown dynamics is directly compensated by the SMO term, so that the DNNs no longer learn the full dynamics and their adaptation becomes less effective. Therefore, a practical trade-off must be achieved to ensure fast and robust SMO convergence without hindering the DNNs learning.

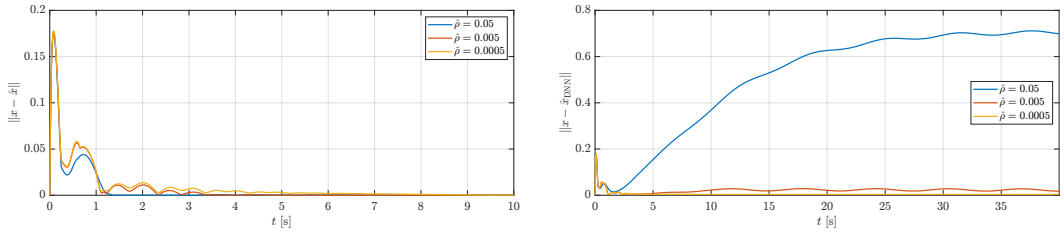

Figure 5.20: Impact of $\hat{\rho}$ on DNNs approximation accuracy. Starting from the left it has: the observer error norm, and the DNN-state estimation error norm.

5.3.4 The BLF Bound

The last parameter considered in this analysis is the BLF bound ε_σ, which defines the prescribed compact set Ω_σ, as detailed in Section 3.3.2, in which the observer sliding variable is constrained, provided that the initial condition lies within this set, thereby mitigating transient overshoots. This effect is demonstrated throughout this chapter and specifically highlighted in Figure 5.21, where, smaller values of ε_σ result in faster convergence, confirming the theoretical enhancement. However, an excessively tight bound can cause numerical instability or even divergence. This

sensitivity arises because the BLF is embedded directly within the adaptation laws (3.10a)-(3.10c), effectively acting as a time-varying learning gain: when the observer sliding variable approaches the boundary, the gain increases sharply. Consequently, ε_σ shares the same fundamental trade-offs identified in the learning rates analysis above. Moreover, this phenomenon is closely tied to the simulation time-step employed, as verified in Figure 5.22. Specifically, by setting $\varepsilon_\sigma = 0.2$ and the time-step 10^{-3}, the system diverges, while by reducing it to 10^{-4} or 10^{-5}, the convergence is recovered. Thus, in practical applications, where time-steps of 10^{-3} are typically employed, larger ε_σ is required.

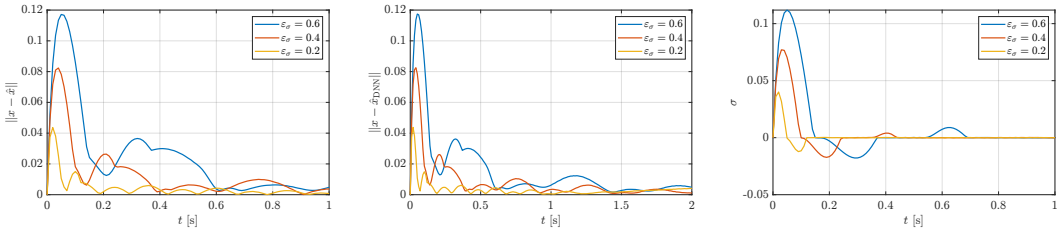

Figure 5.21: Impact of ε_σ on DNNs approximation accuracy. Starting from the left it has: the observer error norm, the DNN-state estimation error norm, and the sliding variable.

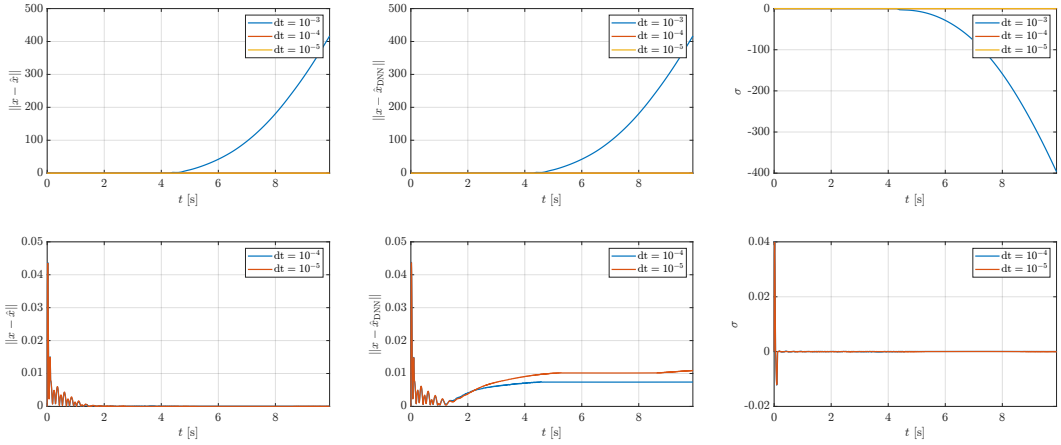

Figure 5.22: Impact of simulation time-step on DNNs approximation accuracy with fixed $\varepsilon_\sigma = 0.2$. Starting from the left it has: the observer error norm, the DNN-state estimation error norm, and the sliding variable. First row shows all three cases, while the second row focuses on time-step 10^{-4} and 10^{-5}.

Chapter 6

Conclusions and Future Works

In this thesis, a novel observer-based control framework integrating Deep Neural Networks and Sliding Modes has been developed, theoretically analyzed, and validated through numerical simulations. In the first part, the fundamental concepts of sliding modes and neural networks have been introduced, providing the theoretical basis for the proposed methodology. Building upon these foundations, the second part has formulated an integrated Deep Neural Network-based Sliding Mode Observer architecture, referred to as DNN-SMO, designed for the online approximation of the unknown dynamics of a perturbed nonlinear system. Additionally, a BLF-enhanced version is presented to guarantee bounded transient performance. The estimated dynamics have then been incorporated into two control strategies: an ISM controller ensuring robustness against matched uncertainties, and a combined MPC/ISM scheme addressing state and input constraints while preserving the robustness properties. Finally, numerical simulations on the Duffing oscillator and 3-DoF robotic manipulator, including a sensitivity analysis to key design parameters, confirmed the effectiveness of the proposed schemes, paving the way for real-world deployment of learning-based robust control systems. Future research directions include:

- the extension of the proposed framework to high-order sliding mode observer to further reduce chattering;
- the formal derivation of recursive feasibility and closed-loop stability proofs for the DNN-SMO based MPC/ISM scheme, for which this work has provided a preliminary numerical validation;
- the experimental validation of the developed architectures on real systems to assess their performance under real-world sensing and actuation constraints.

Appendices

Appendix A

Proofs

A.1 Proof of Theorem 1.1

Consider the Lyapunov-like candidate function $v : \mathcal{X} \to \mathbb{R}$ defined as (1.6). Then, its first time-derivative can be computed as

$$\dot{v}(x(t)) = \frac{d\|\sigma(x(t))\|}{dt}.$$

Exploiting the γ-reaching condition (1.7) holds

$$\frac{d\|\sigma(x(t))\|}{dt} \leq -\gamma\|\sigma(x(t))\|.$$

Integrating with respect to the time between 0 and t_r one obtains

$$\int_0^{t_r} d\|\sigma(x(t))\| \leq \int_0^{t_r} -\gamma dt,$$
$$\|\sigma(x(t_r))\| - \|\sigma(x(0))\| \leq -\gamma\ t_r.$$

Since the reaching time is defined as the time instant in which the sliding mode is enforced, then $\sigma(x(t_r)) = 0_m$. Hence, rearranging the terms and substituting $x(0) = x_0$, it gets

$$t_r \leq \frac{\|\sigma(x_0)\|}{\gamma},$$

concluding the proof.

A.2 Proof of Theorem 1.3

Let $v : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be the Lyapunov-like candidate function chosen as

$$v = \frac{1}{2}\hat{\sigma}^\top \hat{\sigma}.$$

Then, its first time-derivative can be computed as

$$\begin{aligned}
\dot{v} &= \hat{\sigma}^\top \dot{\hat{\sigma}} \\
&= \hat{\sigma}^\top \hat{C}(\dot{y} - \dot{\hat{y}}) \\
&= \hat{\sigma}^\top \hat{C}\left(\frac{\partial \psi}{\partial x}\dot{x} + \frac{\partial \psi}{\partial t} - \frac{\partial \psi}{\partial \hat{x}}\dot{\hat{x}} - \frac{\partial \psi}{\partial t}\right) \\
&= \hat{\sigma}^\top \hat{C}\left(\frac{\partial \psi}{\partial x}(f + Bu + h) - \frac{\partial \psi}{\partial \hat{x}}(\hat{f} + \hat{B}u + l)\right).
\end{aligned} \tag{A.1}$$

For convenience, it is possible to separate the terms that must be compensated, obtaining

$$\begin{aligned}
\dot{v} &= \hat{\sigma}^\top \hat{C} \underbrace{\left(\frac{\partial \psi}{\partial x}(f + Bu + h) - \frac{\partial \psi}{\partial \hat{x}}(\hat{f} + \hat{B}u)\right)}_{\Delta} - \hat{\sigma}^\top \hat{C}\frac{\partial \psi}{\partial \hat{x}} l \\
&= \hat{\sigma}^\top \hat{C}\Delta - \hat{\sigma}^\top \hat{C}\frac{\partial \psi}{\partial \hat{x}} l.
\end{aligned}$$

Substituting l as in (1.16), it holds that

$$\dot{v} = \hat{\sigma}^\top \hat{C}\Delta - \hat{\sigma}^\top \hat{C}\frac{\partial \psi}{\partial \hat{x}}\hat{\rho}\frac{\hat{\sigma}}{\|\hat{\sigma}\|}.$$

If Assumption 1.8 holds, then $\dot{v}$ can be upper bounded as

$$\dot{v} \leq \|\hat{\sigma}\|\|\hat{C}\|\bar{\Delta} - \hat{\sigma}^\top \hat{C}\hat{\rho}\frac{\partial \psi}{\partial \hat{x}}\frac{\hat{\sigma}}{\|\hat{\sigma}\|}. \tag{A.2}$$

Remembering that $\hat{C}\frac{\partial \psi}{\partial \hat{x}}$ is symmetric positive definite (Assumption 1.7), it leads to

$$\hat{\sigma}^\top \hat{C}\frac{\partial \psi}{\partial \hat{x}}\hat{\sigma} \geq \|\hat{\sigma}\|^2 \underline{\lambda}\left(\hat{C}\frac{\partial \psi}{\partial \hat{x}}\right),$$

where $\underline{\lambda}\left(\hat{C}\frac{\partial \psi}{\partial \hat{x}}\right)$ is the minimum eigenvalue of $\hat{C}\frac{\partial \psi}{\partial \hat{x}}$. Inserting the above inequality into (A.2) it has

$$\begin{aligned}
\dot{v} &\leq \|\hat{\sigma}\|\|\hat{C}\|\bar{\Delta} - \|\hat{\sigma}\|\hat{\rho}\underline{\lambda}\left(\hat{C}\frac{\partial \psi}{\partial \hat{x}}\right) \\
&\leq \|\hat{\sigma}\|\left\{\|\hat{C}\|\bar{\Delta} - \hat{\rho}\underline{\lambda}\left(\hat{C}\frac{\partial \psi}{\partial \hat{x}}\right)\right\}.
\end{aligned}$$

Then, choosing $\hat{\rho}$ as in (1.17), one has $\dot{v} \leq -\bar{\eta}\|\hat{\sigma}\|$, with $\bar{\eta} \in \mathbb{R}_{>0}$ being a design parameter, ensuring $\hat{\sigma}(y(t), \hat{y}(t)) \to 0_n$ for $t \to \infty$, concluding the proof.

A.3 Proof of Theorem 3.1

Consider the Lyapunov-like candidate function $v : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ defined as

$$v = \frac{1}{2}\hat{\sigma}^\top\hat{\sigma} + \frac{1}{2}\sum_{j=0}^{k_\Phi} \mathrm{vec}\left(\tilde{V}_j\right)^\top \Gamma_{\Phi_j}^{-1}\mathrm{vec}\left(\tilde{V}_j\right) + \frac{1}{2}\sum_{j=0}^{k_\Psi} \mathrm{vec}\left(\tilde{U}_j\right)^\top \Gamma_{\Psi_j}^{-1}\mathrm{vec}\left(\tilde{U}_j\right),$$

where $\hat{\sigma}$ is the observer sliding variable defined previously. The above function is characterized by a first time-derivative equal to

$$\dot{v} = \hat{\sigma}^\top\dot{\hat{\sigma}} + \sum_{j=0}^{k_\Phi} \mathrm{vec}\left(\tilde{V}_j\right)^\top \Gamma_{\Phi_j}^{-1}\mathrm{vec}\left(\dot{\tilde{V}}_j\right) + \sum_{j=0}^{k_\Psi} \mathrm{vec}\left(\tilde{U}_j\right)^\top \Gamma_{\Psi_j}^{-1}\mathrm{vec}\left(\dot{\tilde{U}}_j\right). \tag{A.3}$$

The expression of $\dot{\hat{\sigma}}$ can be computed as follows

$$\begin{aligned}
\dot{\hat{\sigma}} &= \hat{C}(\dot{x} - \dot{\hat{x}}) \\
&= \hat{C}(f + Bu + h - \hat{\Phi}_{k_\Phi} - \sum_{i=1}^{m} \hat{\Psi}_{k_\Psi}^{[i]} u_i - l) \\
&= \hat{C}(\Phi_{k_\Phi} + \varepsilon_\Phi + \sum_{i=1}^{m} \Psi_{k_\Psi}^{[i]} u_i + \varepsilon_\Psi u + h - \hat{\Phi}_{k_\Phi} - \sum_{i=1}^{m} \hat{\Psi}_{k_\Psi}^{[i]} u_i - l) \\
&= \hat{C}(\tilde{\Phi}_{k_\Phi} + \varepsilon_\Phi + \sum_{i=1}^{m} \tilde{\Psi}_{k_\Psi}^{[i]} u_i + \varepsilon_\Psi u + h - l).
\end{aligned} \tag{A.4}$$

Since $\tilde{V}_j = V_j - \hat{V}_j$ and $\tilde{U}_j = U_j - \hat{U}_j$, with V_j and U_j constants, it holds that $\dot{\tilde{V}}_j = -\dot{\hat{V}}_j$ and $\dot{\tilde{U}}_j = -\dot{\hat{U}}_j$. Hence, substituting the above expressions in (A.3) leads to

$$\begin{aligned}
\dot{v} =& \hat{\sigma}^\top\hat{C}\{\tilde{\Phi}_{k_\Phi} + \varepsilon_\Phi + \sum_{i=1}^{m} \tilde{\Psi}_{k_\Psi}^{[i]} u_i + \varepsilon_\Psi u + h - l\} + \\
&- \sum_{j=0}^{k_\Phi} \mathrm{vec}\left(\tilde{V}_j\right)^\top \Gamma_{\Phi_j}^{-1}\mathrm{vec}\left(\dot{\hat{V}}_j\right) - \sum_{j=0}^{k_\Psi} \mathrm{vec}\left(\tilde{U}_j\right)^\top \Gamma_{\Psi_j}^{-1}\mathrm{vec}\left(\dot{\hat{U}}_j\right).
\end{aligned}$$

Expanding the last two terms to separate the last layer from the inner ones, one has that

$$\begin{aligned}\dot{v} =&\hat{\sigma}^\top \hat{C}\{\tilde{\Phi}_{k_\Phi} + \varepsilon_\Phi + \sum_{i=1}^{m} \tilde{\Psi}_{k_\Psi}^{[i]} u_i + \varepsilon_\Psi u + h - l\} + \\ &- \sum_{j=0}^{k_\Phi - 1} \mathrm{vec}\left(\tilde{V}_j\right)^\top \Gamma_{\Phi_j}^{-1} \mathrm{vec}\left(\dot{\hat{V}}_j\right) - \mathrm{vec}\left(\tilde{V}_{k_\Phi}\right)^\top \left(\Gamma_{\Phi_{k_\Phi}}\right)^{-1} \mathrm{vec}\left(\dot{\hat{V}}_{k_\Phi}\right) + \\ &- \sum_{j=0}^{k_\Psi - 1} \mathrm{vec}\left(\tilde{U}_j\right)^\top \Gamma_{\Psi_j}^{-1} \mathrm{vec}\left(\dot{\hat{U}}_j\right) - \sum_{i=1}^{m} \mathrm{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right)^\top \Gamma_{\Psi_{k_\Psi}}^{-1} \mathrm{vec}\left(\dot{\hat{U}}_{k_\Psi}^{[i]}\right).\end{aligned}$$

Then, substituting $\tilde{\Phi}_{k_\Phi}$ and $\tilde{\Psi}_{k_\Psi}^{[i]}$ with (2.14) and (2.19), respectively, it holds that

$$\begin{aligned}\dot{v} = \hat{\sigma}^\top \hat{C}\{&\Lambda_{\Phi_{k_\Phi}} \mathrm{vec}\left(\tilde{V}_{k_\Phi}\right) + \Delta_{\Phi_{k_\Phi}} + \sum_{j=0}^{k_\Phi - 1} \Lambda_{\Phi_j} \mathrm{vec}\left(\tilde{V}_j\right) + \sum_{j=1}^{k_\Phi - 1} \Xi_{\Phi_j} \Delta_{\Phi_j} + \\ &+ \varepsilon_\Phi + \sum_{i=1}^{m} \left(\Lambda_{\Psi_{k_\Psi}}^{[i]} \mathrm{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right) + \Delta_{\Psi_{k_\Psi}}^{[i]} + \sum_{j=0}^{k_\Psi - 1} \Lambda_{\Psi_j}^{[i]} \mathrm{vec}\left(\tilde{U}_j\right) + \sum_{j=1}^{k_\Psi - 1} \Xi_{\Psi_j}^{[i]} \Delta_{\Psi_j} \right) u_i + \\ &+ \varepsilon_\Psi u + h - l\} - \sum_{j=0}^{k_\Phi - 1} \mathrm{vec}\left(\tilde{V}_j\right)^\top \Gamma_{\Phi_j}^{-1} \mathrm{vec}\left(\dot{\hat{V}}_j\right) + \\ &- \mathrm{vec}\left(\tilde{V}_{k_\Phi}\right)^\top \left(\Gamma_{\Phi_{k_\Phi}}\right)^{-1} \mathrm{vec}\left(\dot{\hat{V}}_{k_\Phi}\right) - \sum_{j=0}^{k_\Psi - 1} \mathrm{vec}\left(\tilde{U}_j\right)^\top \Gamma_{\Psi_j}^{-1} \mathrm{vec}\left(\dot{\hat{U}}_j\right) + \\ &- \sum_{i=1}^{m} \mathrm{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right)^\top \Gamma_{\Psi_{k_\Psi}}^{-1} \mathrm{vec}\left(\dot{\hat{U}}_{k_\Psi}^{[i]}\right).\end{aligned}$$

For convenience, it is possible separate the terms so that it is easier to identify the ones that must be compensated, obtaining

$$\begin{aligned}\dot{v} = \hat{\sigma}^\top \hat{C}\{&\Delta_{\Phi_{k_\Phi}} + \sum_{j=1}^{k_\Phi - 1} \Xi_{\Phi_j} \Delta_{\Phi_j} + \varepsilon_\Phi + \sum_{i=1}^{m} \left(\Delta_{\Psi_{k_\Psi}}^{[i]} + \sum_{j=1}^{k_\Psi - 1} \Xi_{\Psi_j}^{[i]} \Delta_{\Psi_j} \right) u_i + \\ &+ \varepsilon_\Psi u + h - l\} + \hat{\sigma}^\top \hat{C} \Lambda_{\Phi_{k_\Phi}} \mathrm{vec}\left(\tilde{V}_{k_\Phi}\right) + \hat{\sigma}^\top \hat{C} \sum_{j=0}^{k_\Phi - 1} \Lambda_{\Phi_j} \mathrm{vec}\left(\tilde{V}_j\right) + \\ &+ \hat{\sigma}^\top \hat{C} \sum_{i=1}^{m} \left(\Lambda_{\Psi_{k_\Psi}}^{[i]} \mathrm{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right) \right) u_i + \hat{\sigma}^\top \hat{C} \sum_{i=1}^{m} \left(\sum_{j=0}^{k_\Psi - 1} \Lambda_{\Psi_j}^{[i]} \mathrm{vec}\left(\tilde{U}_j\right) \right) u_i + \\ &- \sum_{j=0}^{k_\Phi - 1} \mathrm{vec}\left(\tilde{V}_j\right)^\top \Gamma_{\Phi_j}^{-1} \mathrm{vec}\left(\dot{\hat{V}}_j\right) - \mathrm{vec}\left(\tilde{V}_{k_\Phi}\right)^\top \left(\Gamma_{\Phi_{k_\Phi}}\right)^{-1} \mathrm{vec}\left(\dot{\hat{V}}_{k_\Phi}\right) + \\ &- \sum_{j=0}^{k_\Psi - 1} \mathrm{vec}\left(\tilde{U}_j\right)^\top \Gamma_{\Psi_j}^{-1} \mathrm{vec}\left(\dot{\hat{U}}_j\right) - \sum_{i=1}^{m} \mathrm{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right)^\top \Gamma_{\Psi_{k_\Psi}}^{-1} \mathrm{vec}\left(\dot{\hat{U}}_{k_\Psi}^{[i]}\right).\end{aligned}$$

With the aim of compensating the terms that depend on the weight estimation errors $\tilde{V}_j, \tilde{U}_j$ and $\tilde{U}_{k_\Psi}^{[i]}$, one can select the weight adaptation laws $\dot{\hat{V}}_j, \dot{\hat{U}}_j$ and $\dot{\hat{U}}_{k_\Psi}^{[i]}$ as

in (3.5), (3.6) and (3.7), respectively, obtaining

$$\begin{aligned}
\dot{v} = \hat{\sigma}^\top \hat{C}\{&\Delta_{\Phi_{k_\Phi}} + \sum_{j=1}^{k_\Phi-1} \Xi_{\Phi_j}\Delta_{\Phi_j} + \varepsilon_\Phi + \sum_{i=1}^{m}\left(\Delta^{[i]}_{\Psi_{k_\Psi}} + \sum_{j=1}^{k_\Psi-1}\Xi^{[i]}_{\Psi_j}\Delta_{\Psi_j}\right)u_i + \\
&+ \varepsilon_\Psi u + h - l\} + \hat{\sigma}^\top\hat{C}\Lambda_{\Phi_{k_\Phi}}\mathrm{vec}\left(\tilde{V}_{k_\Phi}\right) + \hat{\sigma}^\top\hat{C}\sum_{j=0}^{k_\Phi-1}\Lambda_{\Phi_j}\mathrm{vec}\left(\tilde{V}_j\right) + \\
&+ \hat{\sigma}^\top\hat{C}\sum_{i=1}^{m}\left(\Lambda^{[i]}_{\Psi_{k_\Psi}}\mathrm{vec}\left(\tilde{U}^{[i]}_{k_\Psi}\right)\right)u_i + \hat{\sigma}^\top\hat{C}\sum_{i=1}^{m}\left(\sum_{j=0}^{k_\Psi-1}\Lambda^{[i]}_{\Psi_j}\mathrm{vec}\left(\tilde{U}_j\right)\right)u_i + \\
&- \sum_{j=0}^{k_\Phi-1}\mathrm{vec}\left(\tilde{V}_j\right)^\top\Gamma^{-1}_{\Phi_j}\underset{\mathcal{B}_{\Phi_j}}{\mathrm{proj}}\left(\Gamma_{\Phi_j}\Lambda^\top_{\Phi_j}\hat{C}^\top\hat{\sigma}\right) + \\
&- \mathrm{vec}\left(\tilde{V}_{k_\Phi}\right)^\top\left(\Gamma_{\Phi_{k_\Phi}}\right)^{-1}\underset{\mathcal{B}_{\Phi_{k_\Phi}}}{\mathrm{proj}}\left(\Gamma_{\Phi_{k_\Phi}}(\Lambda_{\Phi_{k_\Phi}})^\top\hat{C}^\top\hat{\sigma}\right) + \\
&- \sum_{j=0}^{k_\Psi-1}\mathrm{vec}\left(\tilde{U}_j\right)^\top\Gamma^{-1}_{\Psi_j}\underset{\mathcal{B}_{\Psi_j}}{\mathrm{proj}}\left(\Gamma_{\Psi_j}\left(\sum_{i=1}^{m}u_i(\Lambda^{[i]}_{\Psi_j})^\top\right)\hat{C}^\top\hat{\sigma}\right) + \\
&- \sum_{i=1}^{m}\mathrm{vec}\left(\tilde{U}^{[i]}_{k_\Psi}\right)^\top\Gamma^{-1}_{\Psi_{k_\Psi}}\underset{\mathcal{B}_{\Psi_{k_\Psi}}}{\mathrm{proj}}\left(\Gamma_{\Psi_{k_\Psi}}u_i(\Lambda^{[i]}_{\Psi_{k_\Psi}})^\top\hat{C}^\top\hat{\sigma}\right).
\end{aligned}$$

Then exploiting points 3 and 4 of Lemma 3.1 the above equality is transformed into

$$\begin{aligned}
\dot{v} \leq \sigma^\top C\{&\Delta_{\Phi_{k_\Phi}} + \sum_{j=1}^{k_\Phi-1} \Xi_{\Phi_j}\Delta_{\Phi_j} + \varepsilon_\Phi + \sum_{i=1}^{m}\left(\Delta^{[i]}_{\Psi_{k_\Psi}} + \sum_{j=1}^{k_\Psi-1}\Xi^{[i]}_{\Psi_j}\Delta_{\Psi_j}\right)u_i + \\
&+ \varepsilon_\Psi u + h - l\} + \sigma^\top C\Lambda_{\Phi_{k_\Phi}}\mathrm{vec}\left(\tilde{V}_{k_\Phi}\right) + \sigma^\top C\sum_{j=0}^{k_\Phi-1}\Lambda_{\Phi_j}\mathrm{vec}\left(\tilde{V}_j\right) + \\
&+ \sigma^\top C\sum_{i=1}^{m}\left(\Lambda^{[i]}_{\Psi_{k_\Psi}}\mathrm{vec}\left(\tilde{U}^{[i]}_{k_\Psi}\right)\right)u_i + \sigma^\top C\sum_{i=1}^{m}\left(\sum_{j=0}^{k_\Psi-1}\Lambda^{[i]}_{\Psi_j}\mathrm{vec}\left(\tilde{U}_j\right)\right)u_i + \\
&- \sum_{j=0}^{k_\Phi-1}\mathrm{vec}\left(\tilde{V}_j\right)^\top\Lambda^\top_{\Phi_j}C^\top\sigma - \mathrm{vec}\left(\tilde{V}_{k_\Phi}\right)^\top(\Lambda_{\Phi_{k_\Phi}})^\top C^\top\sigma + \\
&- \sum_{j=0}^{k_\Psi-1}\mathrm{vec}\left(\tilde{U}_j\right)^\top\left(\sum_{i=1}^{m}u_i(\Lambda^{[i]}_{\Psi_j})^\top\right)C^\top\sigma - \sum_{i=1}^{m}\mathrm{vec}\left(\tilde{U}^{[i]}_{k_\Psi}\right)^\top u_i(\Lambda^{[i]}_{\Psi_{k_\Psi}})^\top C^\top\sigma.
\end{aligned} \tag{A.5}$$

Recalling that $\dot{v} \in \mathbb{R}$, allows to write the following identities

$$\begin{aligned}
\sum_{j=0}^{k_\Phi-1}\mathrm{vec}\left(\tilde{V}_j\right)^\top\Lambda^\top_{\Phi_j}\hat{C}^\top\hat{\sigma} &= \left(\sum_{j=0}^{k_\Phi-1}\mathrm{vec}\left(\tilde{V}_j\right)^\top\Lambda^\top_{\Phi_j}\hat{C}^\top\hat{\sigma}\right)^\top \\
&= \hat{\sigma}^\top\hat{C}\sum_{j=0}^{k_\Phi-1}\Lambda_{\Phi_j}\mathrm{vec}\left(\tilde{V}_j\right),
\end{aligned}$$

$$\begin{aligned}
\operatorname{vec}\left(\tilde{V}_{k_\Phi}\right)^\top (\Lambda_{\Phi_{k_\Phi}})^\top \hat{C}^\top \hat{\sigma} &= \left(\operatorname{vec}\left(\tilde{V}_{k_\Phi}\right)^\top (\Lambda_{\Phi_{k_\Phi}})^\top \hat{C}^\top \hat{\sigma}\right)^\top \\
&= \hat{\sigma}^\top \hat{C} \Lambda_{\Phi_{k_\Phi}} \operatorname{vec}\left(\tilde{V}_{k_\Phi}\right), \\
\sum_{j=0}^{k_\Psi - 1} \operatorname{vec}\left(\tilde{U}_j\right)^\top \left(\sum_{i=1}^{m} u_i (\Lambda_{\Psi_j}^{[i]})^\top\right) \hat{C}^\top \hat{\sigma} &= \left(\sum_{j=0}^{k_\Psi - 1} \operatorname{vec}\left(\tilde{U}_j\right)^\top \left(\sum_{i=1}^{m} u_i (\Lambda_{\Psi_j}^{[i]})^\top\right) \hat{C}^\top \hat{\sigma}\right)^\top \\
&= \hat{\sigma}^\top \hat{C} \sum_{j=0}^{k_\Psi - 1} \sum_{i=1}^{m} u_i \Lambda_{\Psi_j}^{[i]} \operatorname{vec}\left(\tilde{U}_j\right) \\
&= \hat{\sigma}^\top \hat{C} \sum_{i=1}^{m} \left(\sum_{j=0}^{k_\Psi - 1} \Lambda_{\Psi_j}^{[i]} \operatorname{vec}\left(\tilde{U}_j\right)\right) u_i, \\
\sum_{i=1}^{m} \operatorname{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right)^\top u_i (\Lambda_{\Psi_{k_\Psi}}^{[i]})^\top \hat{C}^\top \hat{\sigma} &= \left(-\sum_{i=1}^{m} \operatorname{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right)^\top u_i (\Lambda_{\Psi_{k_\Psi}}^{[i]})^\top \hat{C}^\top \hat{\sigma}\right)^\top \\
&= \hat{\sigma}^\top \hat{C} \sum_{i=1}^{m} \left(\Lambda_{\Psi_{k_\Psi}}^{[i]} \operatorname{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right)\right) u_i,
\end{aligned}$$

which, if substituted in (A.5), this last one further simplifies into

$$\begin{aligned}
\dot{v} \leq \hat{\sigma}^\top \hat{C} \{ &\Delta_{\Phi_{k_\Phi}} + \sum_{j=1}^{k_\Phi - 1} \Xi_{\Phi_j} \Delta_{\Phi_j} + \varepsilon_\Phi + \\
&+ \sum_{i=1}^{m} \left(\Delta_{\Psi_{k_\Psi}}^{[i]} + \sum_{j=1}^{k_\Psi - 1} \Xi_{\Psi_j}^{[i]} \Delta_{\Psi_j}\right) u_i + \varepsilon_\Psi u + h - l \}.
\end{aligned}$$

Then, substituting the expression of l in (3.3), one has that

$$\begin{aligned}
\dot{v} \leq \hat{\sigma}^\top \hat{C} \{ &\Delta_{\Phi_{k_\Phi}} + \sum_{j=1}^{k_\Phi - 1} \Xi_{\Phi_j} \Delta_{\Phi_j} + \varepsilon_\Phi + \\
&+ \sum_{i=1}^{m} \left(\Delta_{\Psi_{k_\Psi}}^{[i]} + \sum_{j=1}^{k_\Psi - 1} \Xi_{\Psi_j}^{[i]} \Delta_{\Psi_j}\right) u_i + \varepsilon_\Psi u + h - \hat{\rho} \frac{\hat{\sigma}}{\|\hat{\sigma}\|} \}.
\end{aligned}$$

By means of Assumption 1.1, 1.2, 1.4, and 2.2, and Proposition 3.1, it holds that

$$\dot{v} \leq \|\hat{\sigma}\| \|\hat{C}\| \{\bar{c}_\Phi + \bar{\varepsilon}_\Phi + m\left(\bar{c}_\Psi + \bar{\varepsilon}_\Psi\right) \|u\| + \bar{h}\} - \hat{\sigma}^\top \hat{C} \hat{\rho} \frac{\hat{\sigma}}{\|\hat{\sigma}\|}. \tag{A.6}$$

Remembering that $\hat{C}$ is symmetric positive definite (Assumption 3.1), it holds

$$\hat{\sigma}^\top \hat{C} \hat{\sigma} \geq \|\hat{\sigma}\|^2 \underline{\lambda}(\hat{C}),$$

where $\underline{\lambda}(\hat{C})$ is the minimum eigenvalue of $\hat{C}$. Inserting the above inequality into (A.6) it has

$$\begin{aligned}
\dot{v} &\leq \|\hat{\sigma}\| \|\hat{C}\| \{\bar{c}_\Phi + \bar{\varepsilon}_\Phi + m\left(\bar{c}_\Psi + \bar{\varepsilon}_\Psi\right) \|u\| + \bar{h}\} - \hat{\rho} \frac{\|\hat{\sigma}\|^2 \underline{\lambda}(\hat{C})}{\|\hat{\sigma}\|} \\
&\leq \|\hat{\sigma}\| \|\hat{C}\| \{\bar{c}_\Phi + \bar{\varepsilon}_\Phi + m\left(\bar{c}_\Psi + \bar{\varepsilon}_\Psi\right) \|u\| + \bar{h}\} - \hat{\rho} \|\hat{\sigma}\| \underline{\lambda}(\hat{C}).
\end{aligned}$$

Grouping respect $\|\sigma\|$ the above inequality becomes

$$\dot{v} \leq \|\hat{\sigma}\|\{\|\hat{C}\| \left[\bar{c}_\Phi + \bar{\varepsilon}_\Phi + m\left(\bar{c}_\Psi + \bar{\varepsilon}_\Psi\right)\|u\| + \bar{h}\right] - \hat{\rho}\underline{\lambda}(\hat{C})\}.$$

If the control gain is chosen as follows

$$\hat{\rho} > \frac{\bar{\lambda}(\hat{C})\{\bar{c}_\Phi + \bar{\varepsilon}_\Phi + m(\bar{c}_\Psi + \bar{\varepsilon}_\Psi)\|u\| + \bar{h}\} + \bar{\eta}}{\underline{\lambda}(\hat{C})},$$

where $\bar{\lambda}(\hat{C})$ is the greatest eigenvalue of $\hat{C}$ and $\bar{\eta} \in \mathbb{R}_{>0}$, then the first time-derivative of the Lyapunov function is bounded as

$$\dot{v} \leq -\bar{\eta}\|\hat{\sigma}\|,$$

and, as consequence it is guaranteed that $\hat{\sigma}(x(t), \hat{x}(t)) \to 0_n$ for $t \to \infty$, which concludes the proof.

A.4 Proof of Theorem 3.2

Consider the Lyapunov candidate function $v : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ defined as

$$v(x) = \frac{1}{2}\beta(\hat{\sigma}) + \frac{1}{2}\sum_{j=0}^{k_\Phi} \text{vec}\left(\tilde{V}_j\right)^\top \Gamma_{\Phi_j}^{-1} \text{vec}\left(\tilde{V}_j\right) + \frac{1}{2}\sum_{j=0}^{k_\Psi} \text{vec}\left(\tilde{U}_j\right)^\top \Gamma_{\Psi_j}^{-1} \text{vec}\left(\tilde{U}_j\right).$$

Then, its first time-derivative can be computed observing that, according to the chain rule, it holds that

$$\dot{\beta} = \frac{d\beta}{dt} = \frac{d\beta}{d\|\hat{\sigma}\|^2}\frac{d\|\hat{\sigma}\|^2}{dt} = \frac{d\beta}{d\|\hat{\sigma}\|^2}\frac{d(\hat{\sigma}^\top\hat{\sigma})}{dt} = 2\frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top\dot{\hat{\sigma}}.$$

Hence, $\dot{v}$ is equal to

$$\dot{v} = \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top\dot{\hat{\sigma}} + \sum_{j=0}^{k_\Phi} \text{vec}\left(\tilde{V}_j\right)^\top \Gamma_{\Phi_j}^{-1} \text{vec}\left(\dot{\tilde{V}}_j\right) + \sum_{j=0}^{k_\Psi} \text{vec}\left(\tilde{U}_j\right)^\top \Gamma_{\Psi_j}^{-1} \text{vec}\left(\dot{\tilde{U}}_j\right). \quad \text{(A.7)}$$

Substituting the expression of $\dot{\hat{\sigma}}$, as in (A.4), in (A.7) and remembering that $\dot{\tilde{V}}_j = -\dot{\hat{V}}_j$ and $\dot{\tilde{U}}_j = -\dot{\hat{U}}_j$ it holds that

$$\begin{aligned}\dot{v} =& \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top\hat{C}\{\tilde{\Phi}_{k_\Phi} + \varepsilon_\Phi + \sum_{i=1}^{m}\tilde{\Psi}_{k_\Psi}^{[i]}u_i + \varepsilon_\Psi u + h - l\} + \\ &- \sum_{j=0}^{k_\Phi} \text{vec}\left(\tilde{V}_j\right)^\top \Gamma_{\Phi_j}^{-1} \text{vec}\left(\dot{\hat{V}}_j\right) - \sum_{j=0}^{k_\Psi} \text{vec}\left(\tilde{U}_j\right)^\top \Gamma_{\Psi_j}^{-1} \text{vec}\left(\dot{\hat{U}}_j\right).\end{aligned}$$

Expanding the last two terms to separate the last layer from the inner ones, one has that

$$\begin{aligned}\dot{v} =& \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top \hat{C}\{\tilde{\Phi}_{k_\Phi} + \varepsilon_\Phi + \sum_{i=1}^{m} \tilde{\Psi}_{k_\Psi}^{[i]} u_i + \varepsilon_\Psi u + h - l\} + \\ &- \sum_{j=0}^{k_\Phi - 1} \mathrm{vec}\left(\tilde{V}_j\right)^\top \Gamma_{\Phi_j}^{-1} \mathrm{vec}\left(\dot{\tilde{V}}_j\right) - \mathrm{vec}\left(\tilde{V}_{k_\Phi}\right)^\top \left(\Gamma_{\Phi_{k_\Phi}}\right)^{-1} \mathrm{vec}\left(\dot{\tilde{V}}_{k_\Phi}\right) + \\ &- \sum_{j=0}^{k_\Psi - 1} \mathrm{vec}\left(\tilde{U}_j\right)^\top \Gamma_{\Psi_j}^{-1} \mathrm{vec}\left(\dot{\tilde{U}}_j\right) - \sum_{i=1}^{m} \mathrm{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right)^\top \Gamma_{\Psi_{k_\Psi}}^{-1} \mathrm{vec}\left(\dot{\tilde{U}}_{k_\Psi}^{[i]}\right).\end{aligned}$$

Then, substituting $\tilde{\Phi}_{k_\Phi}$ and $\tilde{\Psi}_{k_\Psi}^{[i]}$ with (2.14) and (2.19), respectively, the above equation becomes

$$\begin{aligned}\dot{v} =& \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top \hat{C}\{\Lambda_{\Phi_{k_\Phi}} \mathrm{vec}\left(\tilde{V}_{k_\Phi}\right) + \Delta_{\Phi_{k_\Phi}} + \sum_{j=0}^{k_\Phi - 1} \Lambda_{\Phi_j} \mathrm{vec}\left(\tilde{V}_j\right) + \sum_{j=1}^{k_\Phi - 1} \Xi_{\Phi_j} \Delta_{\Phi_j} + \\ &+ \varepsilon_\Phi + \sum_{i=1}^{m} \left(\Lambda_{\Psi_{k_\Psi}}^{[i]} \mathrm{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right) + \Delta_{\Psi_{k_\Psi}}^{[i]} + \sum_{j=0}^{k_\Psi - 1} \Lambda_{\Psi_j}^{[i]} \mathrm{vec}\left(\tilde{U}_j\right) + \sum_{j=1}^{k_\Psi - 1} \Xi_{\Psi_j}^{[i]} \Delta_{\Psi_j} \right) u_i + \\ &+ \varepsilon_\Psi u + h - l\} - \sum_{j=0}^{k_\Phi - 1} \mathrm{vec}\left(\tilde{V}_j\right)^\top \Gamma_{\Phi_j}^{-1} \mathrm{vec}\left(\dot{\tilde{V}}_j\right) + \\ &- \mathrm{vec}\left(\tilde{V}_{k_\Phi}\right)^\top \left(\Gamma_{\Phi_{k_\Phi}}\right)^{-1} \mathrm{vec}\left(\dot{\tilde{V}}_{k_\Phi}\right) - \sum_{j=0}^{k_\Psi - 1} \mathrm{vec}\left(\tilde{U}_j\right)^\top \Gamma_{\Psi_j}^{-1} \mathrm{vec}\left(\dot{\tilde{U}}_j\right) + \\ &- \sum_{i=1}^{m} \mathrm{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right)^\top \Gamma_{\Psi_{k_\Psi}}^{-1} \mathrm{vec}\left(\dot{\tilde{U}}_{k_\Psi}^{[i]}\right).\end{aligned}$$

Rearranging the terms, one obtains

$$\begin{aligned}\dot{v} =& \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top \hat{C}\{\Delta_{\Phi_{k_\Phi}} + \sum_{j=1}^{k_\Phi - 1} \Xi_{\Phi_j} \Delta_{\Phi_j} + \varepsilon_\Phi + \sum_{i=1}^{m} \left(\Delta_{\Psi_{k_\Psi}}^{[i]} + \sum_{j=1}^{k_\Psi - 1} \Xi_{\Psi_j}^{[i]} \Delta_{\Psi_j} \right) u_i + \\ &+ \varepsilon_\Psi u + h - l\} + \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top \hat{C} \Lambda_{\Phi_{k_\Phi}} \mathrm{vec}\left(\tilde{V}_{k_\Phi}\right) + \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top \hat{C} \sum_{j=0}^{k_\Phi - 1} \Lambda_{\Phi_j} \mathrm{vec}\left(\tilde{V}_j\right) + \\ &+ \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top \hat{C} \sum_{i=1}^{m} \left(\Lambda_{\Psi_{k_\Psi}}^{[i]} \mathrm{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right) \right) u_i + \\ &+ \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top \hat{C} \sum_{i=1}^{m} \left(\sum_{j=0}^{k_\Psi - 1} \Lambda_{\Psi_j}^{[i]} \mathrm{vec}\left(\tilde{U}_j\right) \right) u_i - \sum_{j=0}^{k_\Phi - 1} \mathrm{vec}\left(\tilde{V}_j\right)^\top \Gamma_{\Phi_j}^{-1} \mathrm{vec}\left(\dot{\tilde{V}}_j\right) + \\ &- \mathrm{vec}\left(\tilde{V}_{k_\Phi}\right)^\top \left(\Gamma_{\Phi_{k_\Phi}}\right)^{-1} \mathrm{vec}\left(\dot{\tilde{V}}_{k_\Phi}\right) - \sum_{j=0}^{k_\Psi - 1} \mathrm{vec}\left(\tilde{U}_j\right)^\top \Gamma_{\Psi_j}^{-1} \mathrm{vec}\left(\dot{\tilde{U}}_j\right) + \\ &- \sum_{i=1}^{m} \mathrm{vec}\left(\tilde{U}_{k_\Psi}^{[i]}\right)^\top \Gamma_{\Psi_{k_\Psi}}^{-1} \mathrm{vec}\left(\dot{\tilde{U}}_{k_\Psi}^{[i]}\right).\end{aligned}$$

Substituting the adaptation laws (3.10a), (3.10b) and (3.10c) and exploiting the property of the projection operator Lemma 3.1, it holds that

$$\begin{aligned}
\dot{v} \leq & \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top\hat{C}\{\Delta_{\Phi_{k_\Phi}} + \sum_{j=1}^{k_\Phi-1}\Xi_{\Phi_j}\Delta_{\Phi_j} + \varepsilon_\Phi + \sum_{i=1}^{m}\left(\Delta^{[i]}_{\Psi_{k_\Psi}} + \sum_{j=1}^{k_\Psi-1}\Xi^{[i]}_{\Psi_j}\Delta_{\Psi_j}\right)u_i + \\
& + \varepsilon_\Psi u + h - l\} + \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top\hat{C}\Lambda_{\Phi_{k_\Phi}}\mathrm{vec}\left(\tilde{V}_{k_\Phi}\right) + \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top\hat{C}\sum_{j=0}^{k_\Phi-1}\Lambda_{\Phi_j}\mathrm{vec}\left(\tilde{V}_j\right) + \\
& + \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top\hat{C}\sum_{i=1}^{m}\left(\Lambda^{[i]}_{\Psi_{k_\Psi}}\mathrm{vec}\left(\tilde{U}^{[i]}_{k_\Psi}\right)\right)u_i + \\
& + \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top\hat{C}\sum_{i=1}^{m}\left(\sum_{j=0}^{k_\Psi-1}\Lambda^{[i]}_{\Psi_j}\mathrm{vec}\left(\tilde{U}_j\right)\right)u_i - \sum_{j=0}^{k_\Phi-1}\mathrm{vec}\left(\tilde{V}_j\right)^\top\Lambda_{\Phi_j}^\top\frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{C}^\top\hat{\sigma} + \\
& - \mathrm{vec}\left(\tilde{V}_{k_\Phi}\right)^\top(\Lambda_{\Phi_{k_\Phi}})^\top\frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{C}^\top\hat{\sigma} + \\
& - \sum_{j=0}^{k_\Psi-1}\mathrm{vec}\left(\tilde{U}_j\right)^\top\left(\sum_{i=1}^{m}u_i(\Lambda^{[i]}_{\Psi_j})^\top\right)\frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{C}^\top\hat{\sigma} + \\
& - \sum_{i=1}^{m}\mathrm{vec}\left(\tilde{U}^{[i]}_{k_\Psi}\right)^\top u_i(\Lambda^{[i]}_{\Psi_{k_\Psi}})^\top\frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{C}^\top\hat{\sigma}.
\end{aligned} \tag{A.8}$$

Then, since $\dot{v} \in \mathbb{R}$, it is possible to cancel out the terms that depend on the weight error, having

$$\begin{aligned}
\dot{v} \leq & \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top\hat{C}\{\Delta_{\Phi_{k_\Phi}} + \sum_{j=1}^{k_\Phi-1}\Xi_{\Phi_j}\Delta_{\Phi_j} + \varepsilon_\Phi + \\
& + \sum_{i=1}^{m}\left(\Delta^{[i]}_{\Psi_{k_\Psi}} + \sum_{j=1}^{k_\Psi-1}\Xi^{[i]}_{\Psi_j}\Delta_{\Psi_j}\right)u_i + \varepsilon_\Psi u + h - l\}.
\end{aligned}$$

Then, substituting the expression of l in (3.3), one has that

$$\begin{aligned}
\dot{v} \leq & \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top\hat{C}\{\Delta_{\Phi_{k_\Phi}} + \sum_{j=1}^{k_\Phi-1}\Xi_{\Phi_j}\Delta_{\Phi_j} + \varepsilon_\Phi + \\
& + \sum_{i=1}^{m}\left(\Delta^{[i]}_{\Psi_{k_\Psi}} + \sum_{j=1}^{k_\Psi-1}\Xi^{[i]}_{\Psi_j}\Delta_{\Psi_j}\right)u_i + \varepsilon_\Psi u + h\} - \frac{d\beta}{d\|\hat{\sigma}\|^2}\hat{\sigma}^\top\hat{C}\hat{\rho}\frac{\hat{\sigma}}{\|\hat{\sigma}\|}.
\end{aligned}$$

Since $\frac{d\beta}{d\|\hat{\sigma}\|^2} \in \mathbb{R}_{>0}$ and exploiting Assumption 1.1, 1.2, 1.4, and 2.2, and Proposition 3.1, the above quantity can be bounded as

$$\dot{v} \leq \left|\frac{d\beta}{d\|\hat{\sigma}\|^2}\right|\|\hat{\sigma}\|\|\hat{C}\|\{\bar{c}_\Phi + \bar{\varepsilon}_\Phi + m\left(\bar{c}_\Psi + \bar{\varepsilon}_\Psi\right)\|u\| + \bar{h}\} - \hat{\rho}\left|\frac{d\beta}{d\|\hat{\sigma}\|^2}\right|\hat{\sigma}^\top\hat{C}\frac{\hat{\sigma}}{\|\hat{\sigma}\|}. \tag{A.9}$$

Remembering that $\hat{C}$ is symmetric positive definite it holds

$$\hat{\sigma}^\top\hat{C}\hat{\sigma} \geq \|\hat{\sigma}\|^2\underline{\lambda}(\hat{C}),$$

where $\underline{\lambda}(\hat{C})$ is the minimum eigenvalue of $\hat{C}$. Inserting the above inequality into (A.9) it has

$$\begin{aligned}
\dot{v} &\leq \left|\frac{d\beta}{d\|\hat{\sigma}\|^2}\right| \|\hat{\sigma}\|\|\hat{C}\|\{\bar{c}_\Phi + \bar{\varepsilon}_\Phi + m\left(\bar{c}_\Psi + \bar{\varepsilon}_\Psi\right)\|u\| + \bar{h}\} - \hat{\rho}\left|\frac{d\beta}{d\|\hat{\sigma}\|^2}\right| \frac{\|\hat{\sigma}\|^2\underline{\lambda}(\hat{C})}{\|\hat{\sigma}\|} \\
&\leq \left|\frac{d\beta}{d\|\hat{\sigma}\|^2}\right| \|\hat{\sigma}\|\|\hat{C}\|\{\bar{c}_\Phi + \bar{\varepsilon}_\Phi + m\left(\bar{c}_\Psi + \bar{\varepsilon}_\Psi\right)\|u\| + \bar{h}\} - \hat{\rho}\left|\frac{d\beta}{d\|\hat{\sigma}\|^2}\right| \|\hat{\sigma}\|\underline{\lambda}(\hat{C}) \\
&\leq \left|\frac{d\beta}{d\|\hat{\sigma}\|^2}\right| \|\hat{\sigma}\|\{\|\hat{C}\|[\bar{c}_\Phi + \bar{\varepsilon}_\Phi + m\left(\bar{c}_\Psi + \bar{\varepsilon}_\Psi\right)\|u\| + \bar{h}] - \hat{\rho}\underline{\lambda}(\hat{C})\}.
\end{aligned}$$

Hence, choosing

$$\hat{\rho} > \frac{\bar{\lambda}(\hat{C})\{\bar{c}_\Phi + \bar{\varepsilon}_\Phi + m(\bar{c}_\Psi + \bar{\varepsilon}_\Psi)\|u\| + \bar{h}\} + \bar{\eta}}{\underline{\lambda}(\hat{C})},$$

with $\bar{\eta} \in \mathbb{R}_{>0}$ being a design parameter and with $\bar{\lambda}(\hat{C})$ the greatest eigenvalue of $\hat{C}$, it holds that

$$\dot{v} \leq -\bar{\eta}\left|\frac{d\beta}{d\|\hat{\sigma}\|^2}\right| \|\hat{\sigma}\|.$$

As consequence it is guaranteed that $\hat{\sigma}(x(t), \hat{x}(t)) \to 0_n$ for $t \to \infty$. Moreover, assuming $\hat{\sigma}(x(0), \hat{x}(0)) \in \Omega_\sigma$, it is guaranteed that $\|\hat{\sigma}(x(t), \hat{x}(t))\| \leq \varepsilon_\sigma$, for $t \geq 0$ by virtue of the BLF. This concludes the proof.

Bibliography

[1] K. Zhou and J. C. Doyle, *Essentials of robust control*, vol. 104. Prentice hall Upper Saddle River, NJ, 1998.

[2] A. Weinmann, *Uncertain models and robust control*. Springer Science & Business Media, 2012.

[3] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[4] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[5] J. Hastad, "Almost optimal lower bounds for small depth circuits," in *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pp. 6–20, 1986.

[6] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, "The expressive power of neural networks: A view from the width," *Advances in neural information processing systems*, vol. 30, 2017.

[7] A. Ferrara, G. P. Incremona, E. Vacchini, and N. Sacchi, "Design of neural networks based sliding mode control and observation: An overview," in *2024 17th International Workshop on Variable Structure Systems (VSS)*, pp. 142–147, 2024.

[8] E. Vacchini, N. Sacchi, G. P. Incremona, and A. Ferrara, "Design of a deep neural network-based integral sliding mode control for nonlinear systems under fully unknown dynamics," *IEEE Control Systems Letters*, vol. 7, pp. 1789–1794, 2023.

[9] N. Sacchi, E. Vacchini, G. P. Incremona, and A. Ferrara, "Model predictive control with deep neural network based integral sliding modes generation for

a class of uncertain nonlinear systems," *IFAC-PapersOnLine*, vol. 58, no. 5, pp. 84–89, 2024.

[10] S. Stebler, W. MacKunis, N. Ramos-Pedroza, and M. Reyhanoglu, "A dynamic neural network-based sliding mode observer method for a class of uncertain dynamic systems," in *2017 IEEE Conference on Control Technology and Applications (CCTA)*, pp. 1–6, IEEE, 2017.

[11] A. Isidori, *Nonlinear control systems: an introduction*. Springer, 1985.

[12] V. I. Utkin, *Sliding modes in control and optimization*. Springer Science & Business Media, 2013.

[13] Y. Shtessel, C. Edwards, L. Fridman, A. Levant, *et al.*, *Sliding mode control and observation*, vol. 10. Springer, 2014.

[14] H. K. Khalil and J. W. Grizzle, *Nonlinear systems*, vol. 3. Prentice hall Upper Saddle River, NJ, 2002.

[15] C. Edwards and S. K. Spurgeon, *Sliding mode control: theory and applications*. CRC press, 1998.

[16] B. Draženović, "The invariance conditions in variable structure systems," *Automatica*, vol. 5, no. 3, pp. 287–295, 1969.

[17] I. M. Boiko, "Analysis of chattering in sliding mode control systems with continuous boundary layer approximation of discontinuous control," in *Proceedings of the 2011 American control conference*, pp. 757–762, IEEE, 2011.

[18] A. Levant, "Chattering analysis," *IEEE transactions on automatic control*, vol. 55, no. 6, pp. 1380–1389, 2010.

[19] V. Utkin and J. Shi, "Integral sliding mode in systems operating under uncertainty conditions," in *Proceedings of 35th IEEE conference on decision and control*, vol. 4, pp. 4591–4596, IEEE, 1996.

[20] D. G. Luenberger, "Observing the state of a linear system," *IEEE transactions on military electronics*, vol. 8, no. 2, pp. 74–80, 2007.

[21] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[22] A. Pinkus, "Approximation theory of the mlp model in neural networks," *Acta numerica*, vol. 8, pp. 143–195, 1999.

[23] G. Montúfar, R. Pascanu, K. Cho, and Y. Bengio, "On the number of linear regions of deep neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[24] D. S. Bernstein, *Matrix mathematics: theory, facts, and formulas*. Princeton university press, 2009.

[25] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, JMLR Workshop and Conference Proceedings, 2010.

[26] R. Freeman and P. V. Kokotovic, *Robust nonlinear control design: state-space and Lyapunov techniques*. Springer Science & Business Media, 2008.

[27] H. Kaufman, I. Barkana, and K. Sobel, *Direct adaptive control algorithms: theory and applications*. Springer Science & Business Media, 2012.

[28] M. Krstic, P. V. Kokotovic, and I. Kanellakopoulos, *Nonlinear and adaptive control design*. John Wiley & Sons, Inc., 1995.

[29] F. Lewis, A. Yesildirek, and K. Liu, "Multilayer neural-net robot controller with guaranteed tracking performance," *IEEE Transactions on Neural Networks*, vol. 7, no. 2, pp. 388–399, 1996.

[30] K. P. Tee, S. S. Ge, and E. H. Tay, "Barrier lyapunov functions for the control of output-constrained nonlinear systems," *Automatica*, vol. 45, no. 4, pp. 918–927, 2009.

[31] K. P. Tee and S. S. Ge, "Control of nonlinear systems with full state constraint using a barrier lyapunov function," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pp. 8618–8623, IEEE, 2009.

[32] M. Rubagotti, D. M. Raimondo, A. Ferrara, and L. Magni, "Robust model predictive control with integral sliding mode in continuous-time sampled-data nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 56, no. 3, pp. 556–570, 2010.

[33] A. Ferrara, G. P. Incremona, and L. Magni, "Model-based event-triggered robust mpc/ism," in *2014 european control conference (ECC)*, pp. 2931–2936, IEEE, 2014.

[34] G. P. Incremona, A. Ferrara, and L. Magni, "Hierarchical model predictive/sliding mode control of nonlinear constrained uncertain systems," *IFAC-PapersOnLine*, vol. 48, no. 23, pp. 102–109, 2015.

[35] A. Ferrara, G. P. Incremona, and L. Magni, "A robust mpc/ism hierarchical multi-loop control scheme for robot manipulators," in *52nd IEEE Conference on decision and control*, pp. 3560–3565, IEEE, 2013.

[36] G. P. Incremona, A. Ferrara, and L. Magni, "Asynchronous networked mpc with ism for uncertain nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 9, pp. 4305–4317, 2017.

[37] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: modelling, planning and control*. Springer, 2009.