POLITECNICO DI MILANO
DIPARTIMENTO DI ELETTRONICA, INFORMAZIONE E BIOINGEGNERIA
DOCTORAL PROGRAMME IN INFORMATION TECHNOLOGY

RECONCILING DEEP LEARNING AND CONTROL THEORY: RECURRENT NEURAL NETWORKS FOR MODEL-BASED CONTROL DESIGN

Doctoral Dissertation of:
Fabio Bonassi

Supervisor:
Prof. Riccardo Scattolini

Co-supervisor:
Prof. Marcello Farina

Tutor:
Prof. Lorenzo Mario Fagiano

The Chair of the Doctoral Program:
Prof. Luigi Piroddi

Year 2023 – Cycle XXXV

A Silvia.
Ad Alessandro e Leonardo.

Acknowledgements

It is difficult to find the right words to pay tribute to those who have invaluably contributed to my PhD journey culminating with this dissertation.

I would like to express my uttermost gratitude to Prof. Riccardo Scattolini for his wise and visionary guidance through the turbulent waters that have characterized these years; for his words of regard, which motivated me to give my best; for the trust he placed in me, giving me the possibility to do research and to teach in his master's courses. I am greatly indebted to him for this experience that made me grow personally and professionally.

I am also extremely grateful to Prof. Marcello Farina for his precious support, for his brilliant technical insights, and for encouraging me to go deeper into reasoning even when simple solutions are not in sight. I will not forget the mornings spent at the whiteboard, struggling with proofs that seemed too hard to accomplish.

I would like to thank all my co-authors for the work we did together. In particular, I thank Eng. Enrico Terzi for his pioneering work on the topics discussed in this dissertation. Special thanks to Eng. Alessio La Bella, who is not only a dear friend to me, but also a source of inspiration and motivation. I hereby forgive him for convincing me to do the PhD.

At DEIB, I found brilliant colleagues and precious friends. I would like to thank Alessandro, Anna, Lucrezia, Matteo, Marco, Roberto, Valentina, William, and all those who gave me lighthearted moments. I also want to thank my dear friends Davide, Diego, Giorgio, Giulia C., Giulia F., Stefano *et al.* for their sincere friendship. No matter near or far, chatting with you is always immensely enriching.

I want to express my gratitude to Prof. Thomas Schön for hosting me to Uppsala University, and for patiently guiding me through the investigation of deep energy-based models. I also thank Johannes and Fredrik for their kind support, and all those at Uppsala University who made me feel at home: André, Anna, Antônio, Carl A., Carl J., Håkan, Jennifer, Jonas, Ludvig, Rouqi *et al*. Your welcome warmed the cold Swedish days and made the stay in beautiful Sweden even more enjoyable. In return, I do hope to have persuaded you to use the apple slicer.

This journey would not have been possible without the loving support of Silvia. With you by my side, everything is lighter, the future looks brighter and the challenges are more surmountable.

Finally, I wish to express my gratitude to my parents and sister for their constant generous support, and for teaching me the values of study, dedication and humbleness. My journey is also, and especially, owed to you.

Milan, October 2022

Fabio Bonassi

Financial Support

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 953348.

Fabio Bonassi
Politecnico di Milano
Dipartimento di Elettronica, Informazione e Bioingegneria
`fabio.bonassi@polimi.it`

Abstract

THIS doctoral thesis aims to establish a theoretically-sound framework for the adoption of Recurrent Neural Network (RNN) models in the context of nonlinear system identification and model-based control design. The idea, long advocated by practitioners, of exploiting the remarkable modeling performances of RNNs to learn black-box models of unknown nonlinear systems, and then using such models to synthesize model-based control laws, has already shown considerable potential in many practical applications.

On the other hand, the adoption of these architectures by the control systems community has been so far limited, mainly because the generality of these architectures makes it difficult to attain general properties and to build solid theoretical foundations for their safe and profitable use for control design.

To address these gaps, we first provide a control engineer-friendly description of the most common RNN architectures, i.e., Neural NARXs (NNARXs), Gated Recurrent Units (GRUs), and Long Short-Term Memory networks (LSTMs), as well as their training procedure. The stability properties of these architectures are then analyzed, using common nonlinear systems' stability notions such as the Input-to-State Stability (ISS), the Input-to-State Practical Stability (ISPS), and the Incremental Input-to-State Stability (δISS). In particular, sufficient conditions for these properties are devised for the considered RNN architectures, and it is shown how to enforce these conditions during the training procedure, in order to learn provenly stable RNN models.

Model-based control strategies are then synthesized for these models. In particular, nonlinear model predictive control schemes are first designed: in this context, the model's δISS is shown to enable the attainment of nominal closed-loop stability and, under a suitable design of the control scheme, also robust asymptotic zero-error output regulation. Then, an alternative computationally-lightweight control scheme, based on the internal model control strategy, is proposed, and its closed-loop properties are discussed.

The performances of these control schemes are tested on several nonlinear benchmark systems, demonstrating the potentiality of the proposed framework.

Finally, some fundamental issues for the practical implementation of RNN-based control strategies are mentioned. In particular, we discuss the need for the safety verification of RNN models and their adaptation in front of changes of the plant's behavior, the definition of RNN structures that exploit qualitative physical knowledge of the system to boost the performances and interpretability of these models, and the problem of designing control schemes that are robust to the unavoidable plant-model mismatch.

Glossary

δAS	Incremental Asymptotic Stability.
δIOS	Incremental Input-to-Output Stability.
δISS	Incremental Input-to-State Stability.
AS	Asymptotic Stability.
CEP	Certainty Equivalence Principle.
DL	Deep Learning.
FFNN	Feed-Forward Neural Network.
FHOCP	Finite Horizon Optimal Control Problem.
GD	Gradient Descent.
GRU	Gated Recurrent Units.
IMC	Internal Model Control.
IOPS	Input-to-Output Practical Stability.
ISPS	Input-to-State Practical Stability.
ISS	Input-to-State Stability.
LSTM	Long Short-Term Memory.
ML	Machine Learning.

MPC	Model Predictive Control.
MPRS	Multilevel Pseudo Random Signal.
MSE	Mean Square Error.
NMPC	Nonlinear Model Predictive Control.
NN	Neural Network.
NNARX	Neural Nonlinear AutoRegressive eXogenous.
RL	Reinforcement Learning.
RNN	Recurrent Neural Network.
TBPTT	Truncated Back-Propagation Through Time.

Contents

CHAPTER 1

Introduction

In recent decades, data-driven control has become a solid and flourishing research topic in the systems and control community. Essentially, *data-driven control* refers to all the approaches that, based on informative data collected from the operation of a physical dynamical system, allow to synthesize a controller with no, or only partial, knowledge of the physical laws governing the system itself.

There are many reasons why a control system designer may prefer data-driven control approaches over traditional model-based synthesis, to the point that discussing all of these reasons exhaustively might even be overwhelming. For instance, determining the first-principles model of a plant[1] is, in general, a time-consuming task that requires in-depth knowledge of the specific system under consideration. Not infrequently the underlying physical laws are not even entirely known to the control system designer. Also, first-principle models are often derived operating the plant around nominal conditions, and when the plant operating point needs to be changed, the model needs to be derived again and the controller synthesis

[1] In this work we will use the terms “physical system” and “plant” synonymously to refer to the physical system, characterized by potentially unknown physical laws, for which we want to synthesize a control system.

needs to be repeated. The main idea behind data-driven control strategies is that the data collected from the physical system can be exploited to alleviate the need for human knowledge in the loop [1].

Data-driven control strategies can be broadly grouped in two categories, i.e., the indirect methods and the direct methods. In *indirect methods*, the data collected from the physical system are employed to learn an approximate dynamical model of the system by means of suitable system identification procedures [2]. Then, traditional model-based control strategies, among which one of the most prominent is Model Predictive Control (MPC) [3], are employed to synthesize a control law based on such identified model. Recent notable examples of learning-based indirect control algorithms are based, for example, on Koopman operator [4], set-membership [5], and Gaussian Processes [6, 7]. On the other hand, *direct methods* allow to directly learn a control law from the plant's data. As an example, popular direct approaches are represented by the so-called Virtual Reference Feedback Tuning [8] and by the more recent Data-Enabled Predictive Control [9].

In recent years, increasingly sophisticated data-driven control strategies have been proposed, as the control systems community experienced a fruitful interchange of ideas with the Deep Learning (DL) community. Among all DL tools, Neural Networks (NNs) are those that have received the greatest interest for control applications mainly for two reasons: their flexibility, which allows them to be used in a multitude of different contexts and for different purposes, and their well-assessed universal approximation capabilities [10, 11]. In light of these peculiarities, the use of NN for data-driven control began to be investigated as early as the eighties [12]. However, for many years this interest cooled down due to several shortcomings, such as the limited availability of data and computational resources, as well as the insufficient performances of the NN architectures used at the time.

Significant advances were made only in the 2000s, which are indeed regarded as the golden age of deep learning. In particular, the following developments are to be considered the main factors that have bolstered the application of NNs in data-driven control:

- The increasing availability of large and informative datasets.
- The formulation of novel NN architectures, such as Recurrent Neural Networks (RNNs), that can process data over time and hence learn temporal patterns [13].
- The availability of optimization algorithms for training, which are increasingly efficient and computationally lightweight [14].

- The development of high-quality open-source software libraries that allow to easily train and deploy a wide variety of NN architectures, such as PyTorch and TensorFlow.

The use of NNs for control has been advocated by many contributions, both in the early days – see e.g. [12, 15, 16] – and in more recent years, see e.g. [17–20]. In the following section, an attempt is made to classify the main frameworks in which NNs can be employed for control-related problems.

1.1 Teleonomy of NNs for control

We propose six categories in which most of the control-related applications of NNs can be classified. It should be noted, however, that this list is not meant to be exhaustive since, given the flexibility of NNs, there are several approaches that may not fall into the categories listed below.

Black-box modeling

The most common approach is to employ neural networks as black-box models of the unknown plant in a nonlinear system identification framework [2]. In this case, experiments – designed to suitably excite the system – are carried out and input-output data is collected. An appropriate NN architecture is then chosen and trained to learn the dynamics of the unknown system. That is, the weights of the NN are progressively tuned to make it an accurate model of the system, i.e., a model able to reconstruct the input-output relationship across the temporal dimension.

While black-box nonlinear identification is known to be an hard task, due e.g. to the importance of the design of experiment and to the stability of the numerical algorithm that fits the model to the data, the use of NNs as black-box models has been widely explored both in the academia [16, 19, 21, 22] and in the industry [23–25]. Two of the reasons behind this popularity are the egregious modeling performance of NNs and the generality of the approach, which can be applied to different systems with marginal changes.

Gray-box modeling

Another context in which NNs stand out is that of gray-box modeling. Indeed, when first-principle models are available, they often depend on unknown terms and functions that are difficult to model, e.g., physical relationships that depend on internal variables and unmodeled states. In such cases, NNs can be employed to learn these components from data, which allows to blend classic physical modeling with learning.

This paradigm, known as *Theory-Guided Data Science* [26], allows to avoid complex and over-parametrized black-box models, while achieving better interpretability and modeling performances outside the identification domain. On the other hand, however, this approach requires a good knowledge of the physical system, and the architecture adopted is deeply related to the system itself.

Examples of this strategy are [27] and [28], where the high-level knowledge of mechanical systems is complemented with the use of a NN to learn part of the nonlinear state update functions. A similar approach is proposed in [29], in which NNs are used to learn the kinetic parameters of a CSTR system.

Uncertainty modeling

When a nominal model of the plant is available, be it identified from data or derived from physical equations, a NN can be used to learn the model uncertainty [30], i.e. the plant-model mismatch. Such approach allows to refine the existing model and to improve its accuracy [31]. This, in turn, can significantly enhance the closed-loop performances.

In fact, as an alternative to incorporating the NN uncertainty model within the nominal model, it is actually possible to leverage such uncertainty model to design a control system which guarantees robust stability properties. Note that, when the nominal system is linear, the plant-model mismatch learned by the NN is essentially the linearization error [32]. Based on the nominal linear system, a robust controller can be designed to guarantee robust closed-loop stability and constraint satisfaction in spite of the linearization error.

Approximating computationally-intensive control laws

Another application context for NNs is the approximation of computationally onerous control laws. In fact, there are cases in which the plant's model is known, yet the application of a specific control law is impossible because of its computational cost that would prevent its implementation in real time. This is often the case for Nonlinear MPC laws, as they require to solve a potentially extremely heavyweight optimization problem in real time, especially critical when the system displays fast dynamics. While for linear systems, under mild assumptions, the state-feedback MPC law admits an exact explicit form [33], for nonlinear systems one can, at best, approximate the control law with sufficient accuracy.

In this framework, owing to their approximation capabilities and extremely low online computational demands[2], NNs have often been used as surrogates for MPC control laws [34–36]. It has been recently shown that, if properly used, these networks can preserve the closed-loop stability [37, 38] and fulfill input [39] and state constraints [40]. Recently, it has been shown that NNs can be even employed to successfully approximate robust predictive control laws with guarantees [41].

Other possible approaches that fall into this category are those in which NNs are used to map the nonlinear system model into a linear model defined in the features space, with respect to which explicit control laws, or control laws with limited computational burden, can be synthesized [42, 43].

Direct control learning

Neural networks can also be used in the context of direct data-driven control strategies, in which one seeks to synthesize a control law directly from input and output data collected from the physical system. Many direct approaches for the synthesis of controllers for nonlinear systems rely upon dictionaries of kernel functions for the approximation of the control law, see e.g. [44]. In this context, some recent work has shown that NNs are suitable candidates for learning control laws from data, [45, 46], as they not only lead to superior performances compared to linear control structures, but also, as shown in [47], they can even be designed to fulfill input constraints.

Deep Reinforcement Learning

Lastly, an increasingly popular approach is the so-called deep Reinforcement Learning (RL). Although deep RL has a similar goal to optimal control, namely to find an optimal control law, in the former it is assumed that neither a model of the system nor a cost function against which to evaluate the control action itself are available [48]. Instead, deep RL relies on the availability of a system simulator on which to conduct closed-loop experiments, and an exogenously generated reward signal that quantifies the goodness of the control action. While there is a multitude of deep RL strategies to solve the problem [48–50], many of them share the use of a number of NNs to determine a value function approximation that satisfies

[2] The computational cost of the MPC is not eliminated; it is just brought forward in time, from the real-time deployment to the offline training of the NN.

Bellman's optimality equation and a state-feedback control law that minimizes (or maximizes) such value function.

The use of deep RL strategies for controlling systems with continuous control actions has been, in the past, limited by its computational cost and by the interpretability and reliability of the learned control law. Despite this, some recent work tried to address these issues by combining RL with MPC, see e.g. [51, 52], with the goal of making the control law safer and less computationally intensive.

1.2 Open problems, motivations and contributions

In this work we focus on indirect data-driven control synthesis techniques. More specifically, on the use of recurrent neural networks as black-box models of dynamical systems, to be used in conjunction with model-based control strategies. Therefore, the approach considered herein consists of two distinct steps:

i. The *system identification* step, in which a black-box (recurrent) neural network model of the system is identified, i.e., learned from the data.

ii. The *model-based control synthesis* step, where such model is used to synthesize a model-based control law, in general under the so-called Certainty Equivalence Principle (CEP)[3].

Let us point out that these two steps are strictly related, as the choice of the model's architecture, its properties, and its identification can dramatically affect the control strategy to adopt, its closed-loop performances, and even its closed-loop stability properties. It is therefore necessary to develop a theoretically-sound unified approach which formalizes the use of neural models for the synthesis of model-based control laws.

Before delving into the details of the proposed contribution, let us motivate why, among the many approaches discussed in Section 1.1, we investigated precisely the strategy discussed above.

The first reason is that this approach is very general, since it only assumes stability-related properties of the plant to identify. No requirements on the structure of the physical system are imposed, nor the partial knowledge of the physical laws by which it is governed is necessary; a suitable amount of informative input-output data is enough to carry out the identification procedure. Furthermore, this approach can be combined with the

[3] The CEP is often introduced for indirect data-driven control design, and it consists in assuming that the identified model exactly matches the real system.

most advanced model-based control strategies available which, compared to many direct synthesis methods and deep RL strategies, allows to easily enforce input and output constraints and to track arbitrary output setpoints.

A further reason is that, over the years, this approach has attracted significant interest from control practitioners, in particular in conjunction with RNN models. Recurrent neural networks are stateful NNs, i.e. they feature internal loops storing memory of the past data, which make them particularly suitable for learning dynamical systems. Accounts of applications of RNNs stem from the chemical [39, 53, 54] and pharmaceutical [25] process control domains, industrial manufacturing plant management [24], buildings' HVAC optimization [55], microgrid optimal energy management [56], just to mention a few.

Despite the popularity of this approach, however, there are open problems that need to be addressed from the perspective of control theory.

a. The *safety* and *generalizability* of the RNN black-model must be ensured. The former property means that, for a set of possible input trajectories to the RNN model, it must be guaranteed that the output trajectories are contained within a safe set. The second, instead, relates to ensuring that the accuracy of the model does not degrade excessively in regions not "too far" from the training set. Notably, these two properties imply that the RNN model should be robust to perturbations on the inputs, i.e., these perturbations ideally should cause neither significant fluctuations of model's outputs nor a significant loss of the model's accuracy.

b. A control architecture guaranteeing the closed-loop stability for a broad family of RNN architectures should be devised.

Despite the relevance of these problems, they have yet to be fully tackled by the control systems community. Indeed, although for specific (and, generally, overly simple) architectures some theoretical work is present, for many of the newer, more powerful architectures such problems remain open. We aim to fill these gaps by proposing a theoretical framework for employing RNNs for the identification and control of stable unknown dynamical systems. More specifically, this work brings the following contributions.

i. We show how the main RNN architectures can be represented as discrete time nonlinear dynamical systems in state space form, and we discuss how these models can be used to identify unknown stable dynamical systems that display stability-like properties.

ii. We derive sufficient conditions under which it is possible to guarantee that these networks are Input-to-State Stable (ISS), Input-to-State Practically Stable (ISPS), and Incrementally Input-to-State Stable (δISS). We show how these properties, which are instrumental to ensuring the RNN model's safety and robustness against input perturbations, can be enforced during the training procedure of the RNN model.

iii. We devise a nominally closed-loop stable Nonlinear Model Predictive Control (NMPC) law, which is then extended to guarantee asymptotic offset-free tracking capabilities for constant output reference signals.

iv. An alternative control scheme, based on Internal Model Control, is also proposed owing to its low online computational cost.

1.3 Thesis structure

This doctoral thesis is structured as follows.

Part I – Learning stable RNN models

The first part of the thesis is entirely devoted to the learning of stable RNN models. To this end, in Chapter 2 the ISPS, ISS, and δISS properties are defined, and their relevance in the context of system identification and control design is discussed.

In Chapter 3, the RNN architectures considered in this thesis are described in detail, and sufficient conditions for their ISPS, ISS, and δISS properties are devised. These conditions are then exploited in Chapter 4 in order to train provenly ISPS, ISS, and δISS RNN models. To this end, a suitable training procedure based on the popular truncated back-propagation through time approach is described.

The content of this part is based on the following papers.

- F. Bonassi, M. Farina, and R. Scattolini, “Stability of discrete-time feed-forward neural networks in NARX configuration,” in *19th IFAC Symposium on System Identification (SYSID 2021)*, 2021, pp. 547–552, IFAC-PapersOnLine 54.7
- E. Terzi, F. Bonassi, M. Farina, and R. Scattolini, “Learning model predictive control with long short-term memory networks,” *International Journal of Robust and Nonlinear Control*, vol. 31, no. 18, pp. 8877–8896, 2021
- F. Bonassi, M. Farina, and R. Scattolini, “On the stability properties of gated recurrent units neural networks,” *Systems & Control Letters*, vol. 157, p. 105049, 2021
- F. Bonassi, A. La Bella, G. Panzani, M. Farina, and R. Scattolini, “Deep Long-Short Term Memory networks: Stability properties and Experimental validation,” in *2023 European Control Conference (ECC)*, 2023, *(Under review)*

- F. Bonassi, M. Farina, J. Xie, and R. Scattolini, "On Recurrent Neural Networks for learning-based control: recent results and ideas for future developments," *Journal of Process Control*, vol. 114, pp. 92–104, 2022

Part II – Control design

The second part of the thesis is devoted to the design of model-based control strategies that rely on the trained δISS RNN models. In particular, in Chapter 5 the control problem is introduced, and an overview of the available scientific literature on the topic is provided.

Then, considering one of the proposed RNN architectures (i.e., δISS GRUs), in Chapter 6 two control schemes based on NMPC are proposed: a first simpler scheme, inspired to [22], that allows to guarantee the nominal closed-loop stability; a second scheme, characterized by a slightly more involved design phase, which also attains asymptotic error-free output tracking of constant references.

In Chapter 7 the same control problem is addressed for another class of RNN models (i.e., NNARXs), yielding a nominally closed-loop stable NMPC that is able to achieve offset-free output tracking.

Then, with the aim of providing an alternative control scheme that involves the lowest possible online computational burden, in Chapter 8 the synthesis of Internal Model Control schemes and their closed-loop properties are discussed.

The content of this part is based on the following published papers.

- F. Bonassi, C. F. Oliveira da Silva, and R. Scattolini, "Nonlinear MPC for Offset-Free Tracking of systems learned by GRU Neural Networks," in *3rd IFAC Conference on Modelling, Identification and Control of Nonlinear Systems (MICNON 2021)*, 2021, pp. 54–59, IFAC-PapersOnLine 54.14
- F. Bonassi, J. Xie, M. Farina, and R. Scattolini, "An Offset-Free Nonlinear MPC scheme for systems learned by Neural NARX models," in *2022 IEEE 61st Conference on Decision and Control (CDC)*, 2022, pp. 2123–2128
- F. Bonassi and R. Scattolini, "Recurrent neural network-based Internal Model Control of unknown nonlinear stable systems," *European Journal of Control*, p. 100632, 2022
- F. Bonassi, A. La Bella, M. Farina, and R. Scattolini, "Nonlinear MPC design for incrementally ISS systems with application to GRU networks," *Automatica*, 2023, *(In preparation)*

Part III – Towards practical applications of deep learning for control

In the third and final part, along the lines of [20], the main challenges concerning the use of RNNs for modeling and control design in applicative contexts are discussed.

In particular, in Chapter 9, the problems of safety verification of RNN models and their fine-tuning during the plant's lifespan are outlined, and preliminary solutions are concisely discussed. The research trend of physics-based NNs is also introduced, and the problem of designing control laws with robustness guarantees [64] is mentioned. At last, conclusions are drawn in Chapter 10.

The content of this last part is based on (or, at least, hints to) the following contributions

- F. Bonassi, E. Terzi, M. Farina, and R. Scattolini, "LSTM neural networks: Input to state stability and probabilistic safety verification," in *Learning for Dynamics and Control*. PMLR, 2020, pp. 85–94
- F. Bonassi, J. Xie, M. Farina, and R. Scattolini, "Towards lifelong learning of recurrent neural networks for control design," in *2022 European Control Conference (ECC)*, 2022, pp. 2018–2023
- F. Bonassi, M. Farina, J. Xie, and R. Scattolini, "On Recurrent Neural Networks for learning-based control: recent results and ideas for future developments," *Journal of Process Control*, vol. 114, pp. 92–104, 2022
- J. Xie, F. Bonassi, M. Farina, and R. Scattolini, "Robust offset-free nonlinear model predictive control learned by neural nonlinear autoregressive exogenous models," *International Journal of Robust and Nonlinear Control*, 2022, *(Under review, arXiv preprint 2210.06801)*

Source code availability statement

The source codes implementing the approaches described in this thesis are available upon request to the author, or at the following link:

https://bonassifabio.github.io/phd-thesis

1.4 Notation

In the remainder of this work, the following notation is adopted. Given a vector $v \in \mathbb{R}^n$, we denote by v' its transpose and by $\|v\|_p$ its ℓ_p norm (where $p \geq 1$), i.e.

$$\|v\|_p = \left(\sum_{i=1}^{n} \big|[v]_i\big|^p \right)^{\frac{1}{p}}.$$

The weighted norm of v is indicated by $\|v\|_Q^2 = v'Qv$. A similar notation is adopted for matrices, where the induced ℓ_p norm of a generic matrix A is indicated by

$$\|A\|_p = \max_{v} \frac{\|Av\|_p}{\|v\|_p}.$$

We denote by $\underline{\varsigma}_A$ and $\bar{\varsigma}_A$ the minimum and maximum singular values of matrix A, respectively.

The Hadamard (i.e., element-wise) product between the vectors u and v is denoted by $u \circ v$. We use square brackets primarily for concatenation of vectors, e.g. $z = [u', v']'$, but when followed by subscript i they indicate the i-th component of the vector. For instance, $[v]_i$ represents the i-th component of the vector v.

Time-varying vectors and scalars are denoted by a subscript indicating the time instant to which they refer. In this regard, the letter k and t commonly refer to a discrete time instant. Thus, v_k represents the quantity v at time $k \in \mathbb{Z}_{\geq 0}$.

By sequence we define the realization, between two time instants k_1 and k_2 (with $k_1 \leq k_2$), of some time-varying quantity. Sequences are compactly denoted by $v_{k_1:k_2}$, where $v_{k_1:k_2} = \{v_{k_1}, v_{k_1+1}, ..., v_{k_2}\}$. The $\ell_{p,q}$ norm of a sequence is defined as

$$\|v_{k_1:k_2}\|_{p,q} = \left\| \left[\|v_{k_1}\|_p, \|v_{k_1+1}\|_p, ..., \|v_{k_2}\|_p \right]' \right\|_q.$$

A noteworthy case is that of $q = \infty$, in which

$$\|v_{k_1:k_2}\|_{p,\infty} = \max_{t \in \{k_1, ..., k_2\}} \|v_t\|_p.$$

The main activation functions used throughout the paper are the $\tanh$ and logistic functions, indicated by $\phi(x) = \tanh(x)$ and $\sigma(x) = \frac{1}{1+\exp(-x)}$, respectively. Note that, when applied to vectors, these activation functions are intended to be applied element-wise. Thus, for example,

$$\sigma(v) = \left[\sigma([v]_1), ..., \sigma([v]_n) \right]'.$$

When dealing with multi-layer neural networks, the quantities associated to each layer are indicated by the superscript $^{(l)}$, i.e., the layer index wrapped by parentheses. Therefore, for example, $x^{(l)}$ denotes a quantity related to the l-th layer.

By $\text{diag}(q_1, ..., q_n)$ we indicate an n-by-n diagonal matrix having $q_1, ..., q_n$ on the main diagonal, whereas $I_{n,m}$ represents an n-by-m matrix having ones on the main diagonal. Similarly, $1_{n,m}$ and $0_{n,m}$ represent n-by-m matrices filled with ones and zeros, respectively.

Sets are generally indicated by calligraphic letters, e.g. $\mathcal{S}$, and their interior part is denoted by $\text{Int}(\cdot)$, e.g. $\text{Int}(\mathcal{S})$.

In the remainder of this work, additional notation will be specified as *Notation Addenda*.

Part I

Learning stable RNN models

CHAPTER 2

Stability notions

In this chapter, we formally present the stability notions considered in this work, namely the Input-to-State Stability (ISS), the Input-to-State Practical Stability (ISPS), and the Incremental Input-to-State Stability (δISS). These stability properties have been introduced for continuous-time systems in [67], [68], [69], with the aim of providing a natural tool for stability analysis of nonlinear systems. Other notions previously used for stability analysis of neural networks, such as Asymptotic Stability (AS) and Incremental Asymptotic Stability (δAS) [69], are also presented, and their relationship to ISPS, ISS, and δISS is investigated. Finally, the implications of these stability conditions for identification and control purposes are discussed.

Before delving into the details, however, let us introduce few required definitions.

Definition 2.1 ($\mathcal{K}$ function)**.** *A continuous function $\Psi(s) : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is of class $\mathcal{K}$ if $\Psi(0) = 0$ and $\Psi(s)$ is strictly increasing with its argument.*

Definition 2.2 ($\mathcal{K}_\infty$ function)**.** *A continuous function $\Psi(s) : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is of class $\mathcal{K}_\infty$ if it is of class $\mathcal{K}$ and $\Psi(s) \xrightarrow{s \to +\infty} +\infty$.*

Definition 2.3 ($\mathcal{KL}$ function)**.** *A continuous function $\Psi(s,k) : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is of class $\mathcal{KL}$ if it is of class $\mathcal{K}_\infty$ with respect to its first argument and, for any $s \in \mathbb{R}_{\geq 0}$, $\Psi(s,k) \xrightarrow{k \to +\infty} 0$.*

In this thesis we consider discrete-time nonlinear systems in the following general form

$$\Sigma : \begin{cases} x_{k+1} = f_d(x_k, u_k) \\ y_k = g(x_k) \end{cases}, \tag{2.1}$$

where $u \in \mathbb{R}^{n_u}$, $x \in \mathbb{R}^{n_x}$, and $y \in \mathbb{R}^{n_y}$ are the input, state, and output vectors, respectively. The subscript $_d$ here denotes the dependency of f upon a vector of parameters, i.e. d, such that $f_0(0,0) = 0$. That is, when d is null, the origin is an equilibrium of the system.

Furthermore, in the following the stability properties are stated with respect to the sets $\mathcal{X} \subseteq \mathbb{R}^{n_x}$ and $\mathcal{U} \subseteq \mathbb{R}^{n_u}$, both containing the origin. Therefore, the stability notions here discussed are *regional*, i.e., they only apply within these sets. In case $\mathcal{X} = \mathbb{R}^{n_x}$ and $\mathcal{U} = \mathbb{R}^{n_u}$ the *global* stability definitions are recovered. This choice is motivated by the fact that, as discussed in Section 3, RNNs may enjoy regional stability properties with respect to specific sets $\mathcal{X}$ and $\mathcal{U}$, while global stability definitions would be overly restrictive.

Assumption 2.1. *The set $\mathcal{X}$ with respect to which the stability properties are stated is assumed to be a Positive Invariant Set. That is, for any $u \in \mathcal{U}$,*

$$x \in \mathcal{X} \implies f_d(x,u) \in \mathcal{X}.$$

Notation Addendum 2.1. *Given the generic discrete-time nonlinear system Σ described by* (2.1)*, we denote by $x_k(x_0, u_{0:k}; \Sigma)$ the state of the system Σ at time k, when its initial state is x_0 and it is fed by the input sequence $u_{0:k}$. With a slight abuse of notation, we also denote by $x_{0:k}(x_0, u_{0:k}; \Sigma)$ the resulting state trajectory, i.e.*

$$x_{0:k}(x_0, u_{0:k}; \Sigma) = \big\{x_0,\, x_1(x_0, u_{0:1}; \Sigma), ..., x_k(x_0, u_{0:k}; \Sigma)\big\}. \tag{2.2}$$

For the sake of compactness, in the following Σ may be omitted from (2.2) *when the underlying system is evident from the context.*

Notation Addendum 2.2. *Given a vector v extracted from $\mathcal{V}$, we indicate by $\mathcal{V}_{k_1:k_2}$ the set of possible sequences $v_{k_1:k_2}$. That is,*

$$\mathcal{V}_{k_1:k_2} = \big\{v_{k_1:k_2} \big| \, v_\tau \in \mathcal{V} \; \forall \tau \in \{k_1, ..., k_2\}\big\}. \tag{2.3}$$

2.1 ISS, ISPS, and δISS

In light of the preliminary considerations expressed above, we are now in the position of stating the considered stability properties for nonlinear discrete-time systems. It is worth pointing out that, despite the following definitions are stated along the lines of existing discrete-time ISS [70], ISPS [71], and δISS [72] notions, they are recast in a slightly more general form by adopting a generic ℓ_p norm in place of the ℓ_2 norm on which the aforementioned contributions rely. The equivalence between the proposed and traditionally adopted definitions will then be proved. Finally, the Lyapunov theoretical framework behind these properties will be summarized, and the relationship among these properties will be discussed.

2.1.1 Definition of the stability properties

Definition 2.4 (ℓ_p-ISPS)**.** *System* (2.1) *is said to be ℓ_p-ISPS with respect to the sets $\mathcal{X}$ and $\mathcal{U}$ if there exist a scalar $\varrho \geq 0$ and functions $\beta \in \mathcal{KL}$ and $\gamma \in \mathcal{K}_\infty$ such that, for any $k \in \mathbb{Z}_{\geq 0}$, any $x_0 \in \mathcal{X}$, and any input sequence $u_{0:k} \in \mathcal{U}_{0:k}$, it holds that*

$$\|x_k(x_0, u_{0:k})\|_p \leq \beta(\|x_0\|_p, k) + \gamma(\|u_{0:k}\|_{p,\infty}) + \varrho. \tag{2.4}$$

Definition 2.5 (ℓ_p-ISS)**.** *System* (2.1) *is said to be ℓ_p-ISS with respect to $\mathcal{X}$ and $\mathcal{U}$ if it is ℓ_p-ISPS over $\mathcal{X}$ and $\mathcal{U}$ with $\varrho = 0$. That is, there exists functions $\beta \in \mathcal{KL}$ and $\gamma \in \mathcal{K}_\infty$ such that*

$$\|x_k(x_0, u_{0:k})\|_p \leq \beta(\|x_0\|_p, k) + \gamma(\|u_{0:k}\|_{p,\infty}). \tag{2.5}$$

Notice that both ISS and ISPS guarantee that, regardless of the initial conditions of the system, the state is asymptotically bounded by a function, γ, which is strictly increasing with the maximum input[1] applied to the system. That is, bounded inputs lead to state trajectories that are asymptotically bounded in a set whose amplitude decreases with the amplitude of the applied input trajectories. Smaller input trajectories thus result in tighter bounds.

Under mild assumptions on the smoothness of the the function $g(\cdot)$ of (2.1), the ISS/ISPS of the system also entail the boundedness of the system's output reachable set [20,73], and the ISS/ISPS functions β and γ can be leveraged to compute a (conservative) bound of such set.

A further stability notion considered in this work is that of δISS. This property has been originally proposed for continuous-time nonlinear sys-

[1] By "maximum input" we informally refer to the maximum ℓ_p norm of the input sequence from time 0 to k.

tems [69], and it has been recently extended to discrete-time nonlinear systems in [72]. As for ISS, we here extend the δISS definition by considering arbitrary ℓ_p norms in place of the ℓ_2 norms adopted in [72]. The relationship between the two formulations is then discussed.

Definition 2.6 (ℓ_p-δISS)**.** *System (2.1) is said to be ℓ_p-δISS with respect to the sets $\mathcal{X}$ and $\mathcal{U}$ if there exist functions $\beta \in \mathcal{KL}$ and $\gamma \in \mathcal{K}_\infty$ such that, for any $k \in \mathbb{Z}_{\geq 0}$, any pair of initial states $x_{a,0} \in \mathcal{X}$ and $x_{b,0} \in \mathcal{X}$, and any pair input sequence $u_{a,0:k} \in \mathcal{U}_{0:k}$ and $u_{b,0:k} \in \mathcal{U}_{0:k}$, it holds that*

$$\|x_{a,k} - x_{b,k}\|_p \leq \beta(\|x_{a,0} - x_{b,0}\|_p, k) + \gamma(\|u_{a,0:k} - u_{b,0:k}\|_{p,\infty}), \qquad (2.6)$$

where $x_{a,k} = x_k(x_{a,0}, u_{a,0:k})$ and $x_{b,k} = x_k(x_{b,0}, u_{b,0:k})$.

Since the function β is of class $\mathcal{KL}$, this stability property implies that, for any pair of initial conditions, the resulting state trajectories asymptotically depend only on the input sequences u_a and u_b. In particular, the ℓ_p distance between such state trajectories is bounded by a function that is strictly increasing with the maximum distance between the two inputs. This implies that, asymptotically, the effects of initial conditions vanish, which relieves the problem of correctly initializing the model, and that the closer two input sequences, the smaller is the maximum distance between the resulting state trajectories.

At this stage, let us reconcile the provided definitions with the those available in the literature. To this end, the following Lemma is stated.

Lemma 2.1. *For any $p \geq 1$ and $q \geq 1$, it holds that*

i. If the system is ℓ_p-ISS, then it is also ℓ_q-ISS;

ii. If the system is ℓ_p-ISPS, then it is also ℓ_q-ISPS;

iii. If the system is ℓ_p-δISS, then it is also ℓ_q-δISS.

Proof. See Appendix A.1.1. □

Notice that Lemma 2.1 implies that the ℓ_p-ISPS, ℓ_p-ISS, and ℓ_p-δISS properties are equivalent, respectively, to the ISPS [71], the ISS [70], and the δISS [72] properties proposed in the literature, as these latter are formulated using the ℓ_2 norm.

2.1.2 Lyapunov functions

In this section, we introduce the concepts of ℓ_p-ISS, ℓ_p-ISPS , and ℓ_p-δISS Lyapunov function, based on [70], [71], and [72], respectively.

Definition 2.7 (ℓ_p-ISPS Lyapunov function)**.** *A continuous function $V(x) : \mathbb{R}^{n_x} \to \mathbb{R}_{\geq 0}$ is said to be an ℓ_p-ISS Lyapunov function for system* (2.1) *if there exist scalars $\varrho_1 \geq 0$ and $\varrho_2 \geq 0$, and $\mathcal{K}_\infty$ functions α_1, α_2, α_3, and α_4, such that, for any state $x_k \in \mathcal{X}$, and any input $u_k \in \mathcal{U}$, it holds that*

$$\begin{aligned} \alpha_1(\|x_k\|_p) \leq V(x_k) &\leq \alpha_2(\|x_k\|_p) + \varrho_1, \\ V(f_d(x_k, u_k)) - V(x_k) &\leq -\alpha_3(\|x_k\|_p) + \alpha_4(\|u_k\|_p) + \varrho_2. \end{aligned} \tag{2.7}$$

Definition 2.8 (ℓ_p-ISS Lyapunov function)**.** *If a continuous function $V(x) : \mathbb{R}^{n_x} \to \mathbb{R}_{\geq 0}$ is an ℓ_p-ISPS Lyapunov function for system* (2.1) *over sets $\mathcal{X}$ and $\mathcal{U}$ and $\varrho_1 = \varrho_2 = 0$, then it is said to be an ℓ_p-ISS Lyapunov function. That is, there exist $\mathcal{K}_\infty$ functions α_1, α_2, α_3, and α_4, such that*

$$\begin{aligned} \alpha_1(\|x_k\|_p) \leq V(x_k) &\leq \alpha_2(\|x_k\|_p), \\ V(f_d(x_k, u_k)) - V(x_k) &\leq -\alpha_3(\|x_k\|_p) + \alpha_4(\|u_k\|_p). \end{aligned} \tag{2.8}$$

Definition 2.9 (ℓ_p-δISS Lyapunov function)**.** *A continuous function $V(x_a, x_b) : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \to \mathbb{R}_{\geq 0}$ is said to be an ℓ_p-δISS Lyapunov function for system* (2.1) *if there exist $\mathcal{K}_\infty$ functions α_1, α_2, α_3, and α_4 such that, for any pair of states $x_{a,k} \in \mathcal{X}$ and $x_{b,k} \in \mathcal{X}$, and any pair of inputs $u_{a,k} \in \mathcal{U}$ and $u_{b,k} \in \mathcal{U}$, it holds that*

$$\begin{aligned} \alpha_1(\|x_{a,k} - x_{b,k}\|_p) \leq V(x_{a,k}, x_{b,k}) \leq& \alpha_2(\|x_{a,k} - x_{b,k}\|_p) \\ V(x_{a,k+1}, x_{b,k+1}) - V(x_{a,k}, x_{b,k}) \leq& -\alpha_3(\|x_{a,k} - x_{b,k}\|_p) \\ &+ \alpha_4(\|u_{a,k} - u_{b,k}\|_p), \end{aligned} \tag{2.9}$$

where $x_{a,k+1} = f_d(x_{a,k}, u_{a,k})$ and $x_{b,k+1} = f_d(x_{b,k}, u_{b,k})$.

We now show that the existence of any ℓ_p Lyapunov function implies the existence of any corresponding ℓ_q Lyapunov function.

Lemma 2.2. *For any $p \geq 1$ and $q \geq 1$, it holds that*

i. If the system admits an ℓ_p-ISPS Lyapunov function, then it also admits an ℓ_q-ISPS Lyapunov function;

ii. If the system admits an ℓ_p-ISS Lyapunov function, then it also admits an ℓ_q-ISS Lyapunov function;

iii. If the system admits an ℓ_p-δISS Lyapunov function, then it also admits an ℓ_q-δISS Lyapunov function.

Proof. See Appendix A.1.2. □

Lemma 2.2 allows us to conclude the equivalence between:

- The existence of any ℓ_p-ISS Lyapunov function and the (ℓ_2) ISS Lyapunov function described in [70];
- The existence of any ℓ_p-ISPS Lyapunov function and the (ℓ_2) ISPS Lyapunov function adopted in [71];
- The existence of any ℓ_p-δISS Lyapunov function and the (ℓ_2) δISS Lyapunov function stated in [72].

Owing to such equivalences, we are now in the position to state the relationship between these Lyapunov functions and the associated stability properties.

Proposition 2.1. *If system* (2.1) *admits an ℓ_p-ISS Lyapunov function in the sets $\mathcal{X}$ and $\mathcal{U}$, then it is ℓ_p-ISS with respect to such sets, in the sense specified by Definition* 2.5.

Proof. In light of Lemma 2.2, the existence of an ℓ_p-ISS Lyapunov function implies the existence of an ℓ_2-ISS Lyapunov function. Lemma 3.5 of [70] guarantees that the system is ℓ_2-ISS which, owing to Lemma 2.1, implies that it is ℓ_p-ISS. □

Proposition 2.2. *If system* (2.1) *admits an ℓ_p-ISPS Lyapunov function in the sets $\mathcal{X}$ and $\mathcal{U}$, then it is ℓ_p-ISPS with respect to such sets, in the sense specified by Definition* 2.4.

Proof. Lemma 2.2 ensures that an ℓ_2-ISPS Lyapunov function exists. In light of Theorem 2.5 of [71] the system is hence ℓ_2-ISPS. Owing to Lemma 2.1, the ℓ_p-ISPS of the system is guaranteed. □

Proposition 2.3. *If system* (2.1) *admits an ℓ_p-δISS Lyapunov function over the sets $\mathcal{X}$ and $\mathcal{U}$, then it is ℓ_p-δISS with respect to such sets, in the sense specified by Definition* 2.6.

Proof. The existence of an ℓ_p-δISS Lyapunov function implies, by Lemma 2.2, the existence of an ℓ_2-δISS Lyapunov function. The system is therefore guaranteed to be ℓ_2-δISS by Theorem 1 of [72]. In light of Lemma 2.1 one can thus ensure that the system is ℓ_p-δISS. □

Let us summarize and comment the theoretical framework described above. In Definitions 2.4, 2.5, and 2.6, generalizations of the popular ISPS, ISS, and δISS stability properties have been provided. Such generalized properties are equivalent to the traditional ones, but that they are sometimes easier to prove for RNNs. Finally, in Propositions 2.1, 2.2, and 2.3 the relationship between ℓ_p-ISS, ℓ_p-ISPS, and ℓ_p-δISS, and their respective Lyapunov functions is stated coherently with existing literature.

Henceforth, in light of the equivalence of ℓ_p-ISS notions, ℓ_p-ISPS notions, and ℓ_p-δISS notions, for the sake of simplicity, we will henceforth generically refer to "ISS" and "ISS Lyapunov function" as any generic ℓ_p-ISS and ℓ_p-ISS Lyapunov function, respectively; to "ISPS" and "ISPS Lyapunov function" as any generic ℓ_p-ISPS and ℓ_p-ISPS Lyapunov function, respectively; to "δISS" and "δISS Lyapunov function" as any generic ℓ_p-δISS and ℓ_p-δISS Lyapunov function. This choice allows to simplify the exposition of this work and, owing to Lemma 2.1 and 2.2, does not imply any inaccuracy. Indeed, albeit in general we resort to ℓ_2-ISS, ℓ_2-ISPS, and ℓ_2-δISS, in Chapter 3 other ℓ_p norms may be used to attain simpler mathematical developments.

2.1.3 Relationships among properties

Before discussing the implications of ISS, ISPS, and δISS for system identification and control, it is worth to mention the relationships between these notions.

Among the three properties, the weakest is the ISPS, which only guarantees the boundedness of the state trajectories.

As discussed in Section 2.1.1, the ISS property is a particular, and stronger, case of ISPS, where $\varrho = 0$. Notice that, for the considered systems, when d is null the ISS is always implied by ISPS. Notably, in addition to guaranteeing the state boundedness, ISS also guarantees its asymptotic convergence to the origin in case of asymptotically null inputs.

Finally, one can claim that δISS is an even stronger property. When the origin is an equilibrium of the system, which is always the case if one has a null d, the δISS implies the system's ISS. On the other hand, when d is not null, proving that the δISS implies the ISPS requires to consider further assumptions, namely, that the equilibrium manifold[2] is non-empty, which is hard to guarantee in general. Nonetheless, in Section 3 we show that the proposed conditions for the δISS of the systems under analysis also entail its ISPS, without the need of requiring further conditions.

[2] The equilibrium manifold is defined as the set $\mathcal{S}_d = \{(\bar{x}, \bar{u}) \in \mathcal{X} \times \mathcal{U} : \bar{x} = f_d(\bar{x}, \bar{u})\}$.

2.2 Other stability notions

In this section we introduce additional stability properties, which were originally proposed for continuous-time systems [69] and later repurposed for discrete-time systems, see e.g. [72]. Albeit these stability properties are weaker than δISS (specifically, they are implied by this latter) they are here discussed because most of the existing literature on the stability of RNNs is based on such properties.

In order to present these stability notions, it is useful to first recall the notion of stability of the motion of a system. Let us therefore consider the motion $x_{a,k}(x_{a,0}, u_{0:k})$, associated to the initial state $x_{a,0} \in \mathcal{X}$ and to the input sequence $u_{a,0:k} \in \mathcal{U}_{0:k}$. The motion $x_{a,k}$ is said to be asymptotically stable over $\mathcal{X}$ if, for any motion $x_{b,k}(x_{b,0}, u_{0:k})$, with $x_{b,0} \in \mathcal{X}$, there exists a $\mathcal{KL}$ function β such that, at any time instant $k \in \mathbb{Z}_{\geq 0}$,

$$\|x_{a,k} - x_{b,k}\|_p \leq \beta(\|x_{a,0} - x_{b,0}\|_p, k) \tag{2.10}$$

where, for compactness, $x_{a,k} = x_{a,k}(x_{a,0}, u_{0:k})$ and $x_{b,k} = x_{b,k}(x_{b,0}, u_{0:k})$. When all the motions of system (2.1) satisfy this asymptotic stability definition, the system is said to be incrementally asymptotically stable with respect to the sets $\mathcal{X}$ and $\mathcal{U}$, as shown in the following definition.

Definition 2.10 (δAS)**.** *System* (2.1) *is said to be (ℓ_p) Incrementally Asymptotically Stable (δAS) with respect to $\mathcal{X}$ and $\mathcal{U}$ if there exists a function $\beta \in \mathcal{KL}$ such that* (2.10) *holds for any pair of initial states $x_{a,0} \in \mathcal{X}$ and $x_{b,0} \in \mathcal{X}$, for any input sequence $u_{0:k} \in \mathcal{U}_{0:k}$, and at any $k \in \mathbb{Z}_{\geq 0}$.*

The δAS property implies that, asymptotically, the state trajectory of the system depends solely on the input sequence applied, as the effects of different initial condition vanish. Notice that δAS is weaker than δISS, i.e. if a system is δISS, then it is also δAS. This can be easily verified by taking $u_{a,0:k} = u_{b,0:k} = u_{0:k}$, for which specific case the δISS condition (2.6) boils down to the δAS condition (2.10).

A noteworthy case is when the condition of asymptotic stability (2.10) applies only to system's equilibria, as illustrated in the following definition.

Definition 2.11 (AS)**.** *Consider the equilibrium $(\bar{x}, \bar{u})$, such that $\bar{u} \in \mathcal{U}$, $\bar{x} \in \mathcal{X}$, and $\bar{x} = f_d(\bar{x}, \bar{u})$. Then, the equilibrium is said to be (ℓ_p) Asymptotically Stable (AS) with respect to $\mathcal{X}$ and $\mathcal{U}$ if there exists a function β of class $\mathcal{KL}$ such that*

$$\|\bar{x} - x_k(x_0, \bar{u})\|_p \leq \beta(\|\bar{x} - x_0\|_p, k). \tag{2.11}$$

for any initial state $x_0 \in \mathcal{X}$ and any $k \in \mathbb{Z}_{\geq 0}$.

Remark 2.1. *The stability properties stated in Section 2.1 and 2.2 feature a generic $\mathcal{KL}$ function β. When such function β is exponential with respect to its first argument, i.e., when there exist $\mu > 0$ and $\lambda \in (0, 1)$ such that*

$$\beta(s,t) \leq \mu \, s \, \lambda^k, \tag{2.12}$$

one can speak of exponential *ISS, ISPS, δISS, AS, and δAS.*

2.3 Implications of the introduced stability properties

Having introduced the stability notions used for the analysis of RNNs, we can now discuss the main implications of these properties that will be, in later chapters, exploited for the synthesis of theoretically-sound control laws based on stable RNN models. The following list, summarized in Table 2.1, should therefore not be interpreted as an exhaustive discussion of the implications of the above-mentioned stability notions, but rather as a summary of properties which may be exploited in the following chapters.

Boundedness of the output reachable set

As mentioned in Section 2.1.1, under the assumption that the output transformation of the system is Lipschitz-continuous, the ISPS property entails the boundedness of the system's state and output trajectories. This property is usually stated in terms of the output reachable set, defined as follows.

Definition 2.12 (Output reachable set)**.** *The set $\mathcal{Y} \subseteq \mathbb{R}^{n_y}$ is an output reachable set of system (2.1) if, for any $k \in \mathbb{Z}_{\geq 0}$, any initial state $x_0 \in \mathcal{X}$, and any input sequence $u_{0:k} \in \mathcal{U}_{0:k}$, it holds that*

$$y_\tau(x_0, u_{0:\tau}) \in \mathcal{Y}, \;\; \forall \tau \in \{0, ..., k\},$$

where, of course, $y_\tau(x_0, u_{0:\tau}) = g(x_\tau(x_0, u_{0:\tau}))$.

The following proposition can hence be stated.

Proposition 2.4. *If system (2.1) is ISPS and its output transformation is Lipschitz-continuous, then the system's output reachable set $\mathcal{Y}$ is bounded. Moreover, there exist a function γ_y of class $\mathcal{K}_\infty$ and a scalar $\varrho_y \geq 0$ such that, asymptotically,*

$$\mathcal{Y} \subseteq \bar{\mathcal{Y}} = \left\{ y \in \mathbb{R}^{n_y} : \|y\|_p \leq \gamma_y\Big(\sup_{u \in \mathcal{U}} \|u\|_p \Big) + \varrho_y \right\}. \tag{2.13}$$

Proof. See Appendix A.1.3. □

	ISPS	ISS	AS	δAS	δISS
Output reachable set boundedness	✓	✓			✓[†]
Vanishing of system initialization				✓	✓
Robustness to input perturbations					✓
Stability of linearized models					✓

Table 2.1: *Summary of the implications of the discussed stability properties.*

As discussed in [65], Proposition 2.4 implies that the ISPS property allows to compute a conservative bound of the output reachable set of the system. It is worth noting, however, that this bound may be overly conservative if the function γ_y is conservatively defined. On the other hand, the guaranteed boudnedness of the output reachable set provides a strong theoretical support for the strategy of computing a probabilistic bound via randomized approaches [65], which results in less conservative bounds. This idea is further discussed in Section 9.

Lastly, we point out that the boundedness of the output reachable set is also entailed by the ISS property, as it is a particular case of ISPS.

Vanishing of the system initialization

When the generic system (2.1) represents a black-box model of an unknown system, the problem suitably initializing its states arises. Indeed, an inaccurate initialization of the model could, in general, lead to a severe degradation of the modeling performances.

On the other hand, if the system model is δAS or δISS, the modeling performances are asymptotically independent from the model's initial conditions. Picking two initial states $x_{a,0} \in \mathcal{X}$ and $x_{b,0} \in \mathcal{X}$, for any control sequence $u_{0:k}$, one has that the bound on the maximum distance between the resulting state trajectories converge to zero

$$\|x_{a,k} - x_{b,k}\|_p \leq \beta(\|x_{a,0} - x_{b,0}\|_p, k) \xrightarrow{k\to\infty} 0.$$

The model's output is thus asymptotically independent of the initial state.

It is worth noting that whether this behavior is desirable in a model depends on whether the unknown plant displays, in turn, δAS-like or δISS-like stability properties[3], as discussed in Chapter 4.

[3] Since the plant from which the data are collected is not known, it is generally not possible to analytically verify that it is δAS or δISS. From the input-output trajectories it is however possible to numerically verify whether they are compatible with such stability properties.

[†] Under the mild assumption that the equilibrium manifold of the system is not empty.

Robustness to input perturbations

One of the well-known problems of neural networks is that of so-called robustness to adversarial attacks, that is, the ability to certify that small perturbations on inputs have limited effects on performances. Although this problem has been addressed for nonrecurrent NNs, see e.g. [74], it is beyond challenging for RNNs and, in general, for dynamical systems, in which perturbations can have effects that accumulate over the long term.

In this context, the robustness to input perturbations can be guaranteed by the δISS property. Indeed, letting $d_{u,0:k}$ be the input disturbance, such that $\tilde{u}_{0:k} = u_{0:k} + d_{u,0:k} \in \mathcal{U}_{0:k}$ is the perturbed input, from (2.10) it follows that

$$\|x_k - \tilde{x}_k\|_p \leq \gamma(\|d_{u,0:k}\|_{p,\infty}), \tag{2.14}$$

where $x_k = x_k(x_0, u_{0:k})$ and $\tilde{x}_k = \tilde{x}_k(x_0, \tilde{u}_{0:k})$ denote the unperturbed and the perturbed state trajectories, respectively. Equation (2.14) entails that the deviation of the state trajectories caused by an input perturbation is bounded by a function γ, which is strictly increasing with the maximum amplitude of the perturbation itself. Therefore, the smaller the amplitude of such perturbation, the tighter the bound on the state trajectories deviations.

It is worth noting that, although the γ function of the δISS definition (2.6) can be used for (2.14), it is often defined very conservatively, to the point that such bound is too conservative to be usable in practice. However, once the δISS of the system is assessed, one can numerically compute the functions β and γ, which provide a tighter (and more useful) bound [20].

Stability of linearized models

An implication of exponential δISS is that it guarantees the asymptotic stability of the system's linearization around any equilibrium point. The asymptotic stability of the linearized system is indeed very useful for the synthesis of many control laws, see [61].

Let us therefore consider a generic equilibrium $\bar{\Sigma} = (\bar{x}, \bar{u}, \bar{y})$ such that $\bar{x} = f(\bar{x}, \bar{u})$ and $\bar{y} = g(\bar{x})$, where $\bar{x} \in \text{Int}(\mathcal{X})$ and $\bar{u} \in \text{Int}(\mathcal{U})$. Letting

$$\delta x_k = x_k - \bar{x}, \qquad \delta u_k = u_k - \bar{u}, \qquad \delta y_k = y_k - \bar{y}, \tag{2.15a}$$

the linearized system reads as

$$\delta\Sigma(\bar{\Sigma}) : \begin{cases} \delta x_{k+1} = A_\delta \delta x_k + B_\delta \delta u_k \\ \delta y_k = C_\delta \delta x_k \end{cases}, \tag{2.15b}$$

where the matrices $A_\delta = A_\delta(\bar{\Sigma})$, $B_\delta = B_\delta(\bar{\Sigma})$, and $C_\delta = C_\delta(\bar{\Sigma})$ are computed as

$$A_\delta(\bar{\Sigma}) = \left.\frac{\partial f(x,u)}{\partial x}\right|_{\bar{x},\bar{u}}, \qquad B_\delta(\bar{\Sigma}) = \left.\frac{\partial f(x,u)}{\partial u}\right|_{\bar{x},\bar{u}}, \qquad C_\delta(\bar{\Sigma}) = \left.\frac{\partial g(x)}{\partial x}\right|_{\bar{x}}. \tag{2.15c}$$

The following theoretical contribution establishes a relationship between the exponential δISS property and the local asymptotic stability of the linearized system (2.15).

Theorem 2.1. *Let system* (2.1) *be exponentially δISS, and $\bar{\Sigma} = (\bar{x}, \bar{u}, \bar{y})$ be a generic equilibrium, such that $\bar{x} = f(\bar{x}, \bar{u})$ with $\bar{x} \in \mathrm{Int}(\mathcal{X})$ and $\bar{u} \in \mathrm{Int}(\mathcal{U})$, and $\bar{y} = g(\bar{x})$. Then, the linearization of* (2.1) *around $\bar{\Sigma}$, i.e.* (2.15)*, is locally asymptotically stable. That is, the matrix $A_\delta(\bar{\Sigma})$ is Schur stable.*

Proof. See Appendix A.1.4. □

2.4 Summary

In this chapter, the main stability notions that will be henceforth considered, i.e. ISPS, ISS, and δISS, have been formally introduced. The relationship among these properties have been discussed, and their association to other stability properties such as AS and δAS have been highlighted. Finally, the implications of these stability properties, which in future chapters will be leveraged to design theoretically-sound control strategies, have been explored. In the next chapter, the main RNN architectures are presented, and sufficient conditions to guarantee their ISPS, ISS, and δISS, are devised.

CHAPTER 3

Stable RNN architectures for system identification

As extensively discussed in Chapter 1, due to their modeling capabilities, neural networks have gathered an increasing research interest in the control systems domain, in particular concerning their use as black-box models for system identification purposes [2, 20].

Neural networks are nonlinear models, inspired by the operating principles of biological neural networks, and generally consist of a series of linear functions followed by nonlinear transformations. In light of their overparameterization, these models are extremely flexible and powerful, but they require a numerical optimization procedure – called *training procedure* – to determine the parameters, named *weights*, so as to minimize a loss function that quantifies how inaccurately the neural network is simulating the available data.

That of neural networks is a vast topic, and an exhaustive description of it is certainly beyond the scope of this work. The interested reader is therefore directed to specific works on the topic [14, 75]. Let it suffice to note that NNs can be divided into two broad categories: (*i*) Feed-Forward Neural Networks (FFNNs), which correspond to static transformations in

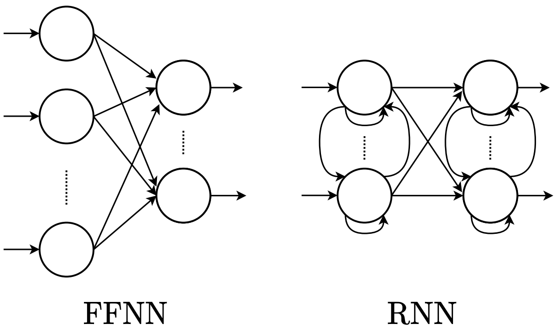

Figure 3.1: *Visual comparison of feed-forward and recurrent neural network architectures.*

which information flows from the inputs of the network to the outputs in an instantaneous and unidirectional fashion, with no memory retained of past data; (*ii*) Recurrent Neural Networks (RNNs), which feature internal loops, whose purpose is to maintain memory of past trajectories, that make these networks stateful, so that they actually are dynamical systems. A simple visual comparison between these two categories is proposed in Figure 3.1.

Being nonlinear dynamical systems, RNNs are particularly appropriate for system identification tasks [13,20]. They have been shown to potentially approximate any dynamical system, see [11]: this property is known as the universal approximation. However, traditional and simple RNN architectures, also called *vanilla* RNNs (see [14]), are affected by the so-called vanishing/exploding gradient problem, which in practice greatly limits the modeling performances that these architectures can achieve. Examples of vanilla RNN architectures and an analysis of the vanishing and exploding gradient issues are reported in [76] and [77]. To some extent, Neural NARX (NNARX) models [57], i.e. autoregressive models in which the nonlinear regression function consists of a FFNN, can be considered in this family. Because of vanishing and exploding gradient problems, these models have historically been trained by minimizing the one-step ahead prediction error [21], or the free-run simulation error over a short time horizon.

A first attempt to construct RNNs whose training is not plagued by gradient problems is that of Echo State Networks (ESNs) [78, 79]. This architecture is peculiar in that the network's states have fixed (i.e., untrainable) and randomly generated dynamics at the time of network creation. These fixed dynamics constitute the so-called dynamic reservoir. Training ESNs boils down to a linear regression problem, that is simple to solve and, unlike other RNNs, does not require iterative gradient descent-based algorithm.

Although their structure makes them particularly interesting, there are cases where the modeling capabilities of ESNs may prove insufficient, mainly because of their fixed dynamics. To achieve better performance while avoiding gradient problems, the *gated* RNNs have been proposed in the literature, in which the flow of information and the memory of past data are regulated by the so-called gates. In particular, the most popular gated architectures are Long Short-Term Memory networks (LSTMs), proposed in [80], and Gated Recurrent Units (GRUs), more recently proposed in [81]. In many applications, such as time series prediction [13, 82], natural language processing [83], and dynamical systems identification [84, 85], both architectures have demonstrated vastly superior performances than vanilla RNNs and ESNs.

In this thesis, we focus on three architectures: NNARXs, LSTMs, and GRUs. As we will discuss, this choice is due to the their complementary nature: NNARXs are characterized by the simplest structure, and are the easiest to train and use for control, but at the same time have more modest performances; LSTMs are the most complex to train, but have greater representational flexibility; finally, GRUs have intermediate performances but a simpler structure than LSTMs, which facilitates and speeds up the training procedure, often to the benefit of the modeling performances themselves.

Being nonlinear dynamical systems, we henceforth consider these architecture to fit the following (generic) state-space form

$$\Sigma(\Phi): \begin{cases} x_{k+1} = f(x_k, u_k; \Phi) \\ y_k = g(x_k; \Phi) \end{cases} \tag{3.1}$$

where $x_k \in \mathbb{R}^{n_x}$ denotes the state vector, $u_k \in \mathbb{R}^{n_u}$ the input vector, and $y_k \in \mathbb{R}^{n_y}$ the output vector, while Φ indicate the set of weights of the network.

Notice that the generic form (3.1) resembles system (2.1) for which the stability properties of interest have been described in Section 2, where however the dependence on the term d has been omitted for conciseness. Nonetheless, for each proposed architecture, it is shown that this term is represented by a subset of the network's weights Φ (i.e., $d = d(\Phi)$) whose nullity ensures that the origin is an equilibrium of $\Sigma(\Phi)$.

In the sections that follow, for each of the three mentioned architectures, the state-space form is described, and the conditions that guarantee its stability properties are derived. Before entering into the details of the proposed stability conditions, it is worth to briefly describe the existing literature on the stability properties of RNN architectures.

Existing literature on RNNs stability properties

The problem of ensuring the stability of RNN models has been only partially addressed in the existing literature, which mostly focused on the stability of continuous-time autonomous RNNs, see [86] and references therein. Nonetheless, these works are limited to continuous-time vanilla RNNs that are, on the one hand, continuous time – which turn out to be harder to train and to use for long-term prediction than their discrete-time counterpart –, and on the other hand, autonomous, meaning that input variables are not considered in the stability analysis. The AS and δAS properties of discrete-time vanilla RNNs have been analyzed in [87–89] and [90], respectively. In [91], sufficient conditions for the ISS of NNARXs have been proposed, but they are limited to continuous-time single-layer networks.

The stability properties of ESNs have been recently studied. In [92] a sufficient condition for the global δAS of ESNs has been proposed, while in [93] conditions guaranteeing the stronger δISS property have been devised and such property has been exploited for the synthesis of a nominally closed-loop stable NMPC law.

Concerning the gated RNNs, however, little results are available. In particular, the AS of the origin has been analyzed for single-layer autonomous LSTMs [94, 95]. Moreover, in [90] sufficient conditions for the δAS of single-layer LSTMs have been derived. Concerning GRUs, the equilibria of single-layer networks have been analyzed, and their local AS have been investigated in [96].

Contributions

Motivated by these gaps in the literature, we address the fundamental issue of guaranteeing the safety and the theoretical soundness of NNARXs, LSTMs, and GRUs architectures. This is obtained by analyzing the ISPS, ISS, and δISS properties of the considered architectures. In the sections that follow we propose, for each of these architectures, sufficient conditions on its weights that allow to guarantee its ISPS, ISS, and δISS. Such conditions boil down to a set of nonlinear inequalities in the form

$$\nu(\Phi) < 0, \tag{3.2}$$

where $\nu(\Phi)$ is a vector of suitable nonlinear transformations which depend both on the particular RNN architecture and on the specific stability property under consideration. Notably, condition (3.2) not only allows one to certify whether the RNN model (3.1) is ISPS, ISS, or δISS, but also allows to train provenly ISPS, ISS, δISS RNN models. Such suitable training procedure will be described in Chapter 4. To the best of our knowledge, this represents a novel and relevant contribution to the existing literature.

This chapter is structured as follows. In Section 3.1, the sufficient conditions for the ISS, ISPS, and δISS of NNARXs are devised, based on the paper [57]. Section 3.2 deals with the same stability properties for LSTMs. In particular, based on [22] sufficient conditions are initially devised for single-layer networks, and then extended to deep LSTMs [59]. Lastly, in Section 3.3 sufficient conditions for the ISPS, ISS, and δISS of GRUs are devised, based on [58].

3.1 NNARXs

In Neural NARX models, the output y_{k+1} is assumed to depend only on the previous H input and output data (H will be henceforth named *regression horizon*), as well as the current input u_k. The output y_{k+1} of the NNARX model is hence defined by

$$y_{k+1} = \eta(y_k, y_{k-1}, \ldots, y_{k-H+1}, u_k, u_{k-1}, \ldots, u_{k-H}; \Phi), \tag{3.3}$$

where η is a FFNN, parametrized by the weights Φ, representing the nonlinear regression function which establishes the relationship between the past data, the control variable, and the future output. For these models, η is assumed to be a vector of n_y Lipschitz-continuous functions. Model (3.3) can be recast as a discrete-time system in normal canonical form [97]. To this end, let us define $z_{h,k} = [y'_{k-H+h}, u'_{k-H-1+h}]' \in \mathbb{R}^{n_z}$, where $n_z = n_u + n_y$ and $h \in \{1, \ldots, H\}$ is an index spanning the regression horizon. Noticing that $z_{H,k+1} = [y'_{k+1}, u'_k]'$, (3.3) can be rewritten as

$$\begin{cases} z_{1,k+1} = z_{2,k} \\ \quad \vdots \\ z_{H-1,k+1} = z_{H,k} \\ z_{H,k+1} = \begin{bmatrix} \eta(z_{1,k}, z_{2,k}, \ldots, z_{H,k}, u_k; \Phi) \\ u_k \end{bmatrix} \\ y_k = [I \quad 0]\, z_{H,k} \end{cases} \tag{3.4}$$

Defining the state of the NNARX model as $x_k = [z'_{1,k}, \ldots, z'_{H,k}]' \in \mathbb{R}^{n_x}$, with $n_x = Hn_z$, system (3.4) can be compactly represented as

$$\begin{cases} x_{k+1} = Ax_k + B_u u_k + B_\eta \eta(x_k, u_k) \\ y_k = Cx_k \end{cases}, \tag{3.5a}$$

where, with a slight abuse of notation,

$$\eta(x_k, u_k) = \eta(y_k, y_{k-1}, ..., y_{k-H+1}, u_k, u_{k-1}, ..., u_{k-H}; \Phi),$$

and the fixed matrices A, B_u, B_η, and C are defined as follows

$$A = \begin{bmatrix} 0_{n_z,n_z} & I_{n_z,n_z} & 0_{n_z,n_z} & \cdots & 0_{n_z,n_z} \\ 0_{n_z,n_z} & 0_{n_z,n_z} & I_{n_z,n_z} & \cdots & 0_{n_z,n_z} \\ \vdots & & & \ddots & \vdots \\ 0_{n_z,n_z} & 0_{n_z,n_z} & 0_{n_z,n_z} & \cdots & I_{n_z,n_z} \\ 0_{n_z,n_z} & 0_{n_z,n_z} & 0_{n_z,n_z} & \cdots & 0_{n_z,n_z} \end{bmatrix}, \tag{3.5b}$$

$$B_u = \begin{bmatrix} 0_{n_z,n_u} \\ \vdots \\ 0_{n_z,n_u} \\ \tilde{B}_u \end{bmatrix} \text{ with } \tilde{B}_u = \begin{bmatrix} 0_{n_y,n_u} \\ I_{n_u,n_u} \end{bmatrix}, \tag{3.5c}$$

$$B_\eta = \begin{bmatrix} 0_{n_z,n_y} \\ \vdots \\ 0_{n_z,n_y} \\ \tilde{B}_\eta \end{bmatrix} \text{ with } \tilde{B}_\eta = \begin{bmatrix} I_{n_y,n_y} \\ 0_{n_u,n_y} \end{bmatrix}, \tag{3.5d}$$

$$C = \begin{bmatrix} 0_{n_y,n_z} & \cdots & 0_{n_y,n_z} & \tilde{C} \end{bmatrix} \text{ with } \tilde{C} = \begin{bmatrix} I_{n_y,n_y} & 0_{n_y,n_u} \end{bmatrix}. \tag{3.5e}$$

Being the function $\eta(x_k, u_k)$ a FFNN with L layers[1], it consists of a series of function combinations reading as follows

$$\begin{aligned} \eta_1 &= \psi_1\big(U^{(1)} x_k + W^{(1)} u_k + b^{(1)}\big), \\ \eta_2 &= \psi_2\big(U^{(2)} \eta_1 + W^{(2)} u_k + b^{(2)}\big), \\ &\vdots \\ \eta_L &= \psi_L\big(U^{(L)} \eta_{L-1} + W^{(L)} u_k + b^{(L)}\big), \\ \eta(x_k, u_k) &= U^{(0)} \eta_L + b^{(0)}, \end{aligned} \tag{3.5f}$$

where ψ_l indicates the activation function of the l-th layer which, for the sake of simplicity, is assumed to be Lipschitz-continuous with constant $L_\psi^{(l)}$ and zero-centered, i.e. $\psi_l(0) = 0$. For example, one may take $\psi_l = \phi$, i.e. the $\tanh$ activation function, having Lipschitz constant $L_\psi^{(l)} = 1$.

[1] It is reminded that, for multi-layer networks, the appendix $^{(l)}$ is used to denote a quantity associated to the l-th layer, $l \in \{1, ..., L\}$.

System (3.5) fully describes the state-space NNARX model and falls in the generic form (3.1), where the set of weights Φ reads as

$$\Phi = \big\{U^{(0)}, b^{(0)}, \{U^{(l)}, W^{(l)}, b^{(l)}\}_{\forall l \in \{1,...,L\}}\big\}.$$

Notice that $d(\Phi) = [b^{(0)\prime}, ..., b^{(L)\prime}]'$. Indeed, if d is null, it is easy to see that $\eta(0_{n_x,1}, 0_{n_u,1}) = 0_{n_y,1}$ and hence the origin is an equilibrium of $\Sigma(\Phi)$.

Moreover, it is worth emphasizing that in the NNARX model (3.5), the state vector x_k is a concatenation of the input and output vectors of the last H time-steps. When this model is used in closed-loop (e.g., in combination with a model-based control law), the current state of the model is known, since the past inputs and outputs are known. This means that, although it is a black-box model, there is no need to design a state observer, which greatly simplifies the control design.

3.1.1 Sufficient conditions for stability

Having described the NNARX models, we are now in the position to state their stability properties.

Theorem 3.1 (ISPS of NNARXs)**.** *A sufficient condition for the global exponential ISPS*[2] *of the NNARX* (3.5) *is that*

$$\prod_{l=0}^{L} \|U^{(l)}\|_2 - \frac{1}{\big(\prod_{l=1}^{L} L_\psi^{(l)}\big)\sqrt{H}} < 0. \tag{3.6}$$

Proof. See Appendix A.2.1. □

Corollary 3.1 (ISS of NNARXs)**.** *If* (3.6) *holds and the bias $b^{(l)}$ is null for any $l \in \{0, ..., L\}$, the NNARX model is globally exponentially ISS.*

Proof. As discussed above, if $b^{(l)}$ is null $\forall l \in \{0, ..., L\}$, i.e., $d(\Phi)$ is null, then the origin is an equilibrium of (3.5). Moreover, taking the same Lyapunov function candidate as in Appendix A.2.1, in light of (A.38e) and (A.38f), one gets $\varrho_1 = \varrho_2 = 0$, meaning that it also represents an ℓ_2-ISS Lyapunov function. Invoking Proposition 2.1, the ISS of the NNARX model can be guaranteed. □

Theorem 3.2 (δISS of NNARXs)**.** *If the NNARX model* (3.5) *is ISPS by Theorem 3.1, i.e., if* (3.6) *holds, then the system is also globally exponentially δISS.*

Proof. See Appendix A.2.2. □

[2] Global ISPS corresponds to the ISPS with respect to the sets $\mathcal{X} = \mathbb{R}^{n_x}$ and $\mathcal{U} = \mathbb{R}^{n_u}$.

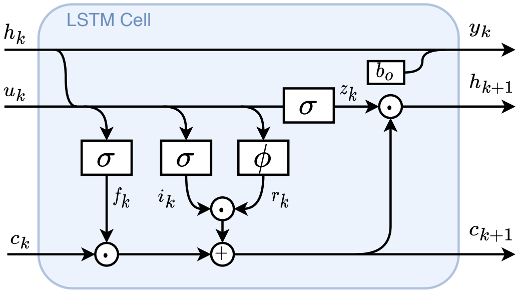

Figure 3.2: *Schematic of a shallow LSTM network.*

It is worth highlighting that the stability condition (3.6) is a nonlinear inequality on the weights of the network, which falls into the general form (3.2), since the terms $L_{\psi}^{(l)}$ are known constants that solely depend on the chosen activation functions.

3.2 LSTMs

In this section, LSTM models are formulated in the state-space form (3.1), and their stability properties are investigated. For simplicity of exposition, we first present the simple case of single-layer LSTMs, after which the more general case of deep LSTMs is analyzed.

To this end, we introduce a customary assumption when working with NN models, namely the unity-boundedness of the input variable.

Assumption 3.1. *The input is unity-bounded. That is, for any k, the input satisfies $u_k \in \mathcal{U}$, where*

$$\mathcal{U} = \{u \in \mathbb{R}^{n_u} : \|u\|_\infty \leq 1\}. \tag{3.7}$$

Notice that, as long as the control input is limited, Assumption 3.1 can always be satisfied by means of a suitable normalization of the input data.

3.2.1 Shallow LSTMs

The *shallow* (i.e., single-layer) LSTM model with input u and output y is described by the following state-space model [22]

$$\begin{cases} c_{k+1} = f_k \circ c_k + i_k \circ \phi(W_r u_k + U_r\, h_k + b_r) \\ h_{k+1} = z_k \circ \phi(c_{k+1}) \\ y_k = U_o h_k + b_o \end{cases}, \tag{3.8a}$$

where the states $c_k \in \mathbb{R}^{n_c}$ and $h_k \in \mathbb{R}^{n_c}$ are usually named *cell state* and *hidden state*, respectively, and n_c is the number of neurons of the layer, which is an hyperparameter (that is, a design knob) of the model. The state vector of the LSTM model is the concatenation of the cell and hidden states, i.e.

$$x_k = [c_k', h_k']'. \tag{3.8b}$$

Notice that the number of states of the layer is defined by the number of neurons, i.e., $n_x = 2n_c$. The terms f_k, i_k, and z_k appearing in (3.8a) are the gates of the LSTM, that rule the flow of information throughout the network, thus allowing to tackle the vanishing and exploding gradient problem and to retain long-term memory [77, 80]. These gates are described by the following equations

$$\begin{aligned} f_k &= \sigma(W_f u_k + U_f h_k + b_f), \\ i_k &= \sigma(W_i u_k + U_i h_k + b_i), \\ z_k &= \sigma(W_z u_k + U_z h_k + b_z), \end{aligned} \tag{3.8c}$$

where we recall that σ is the sigmoidal activation function, evaluated element-wise on its arguments. In the following, for the sake of compactness, we may also denote $r_k = \phi(W_r u_k + U_r\, h_k + b_r)$, referred to as *squashed input*, where ϕ is the $\tanh$ activation function.

In Figure 3.2, the shallow LSTM architecture is schematically represented adopting common deep learning conventions: merging arrows denote linear combination operations, whereas branching arrows represent vector copying operations.

At this stage, let us point out that, since $\sigma(\cdot) \in (0, 1)$, the gates are vectors of positive subunitary elements. The “aperture” of these gates depend on the cell input and on the hidden state: gates with values close to 0 are referred to as “closed”, since they block the flow of information, while gates with values close to 1 are referred to as “open”, since they let the information flow.

It is evident that the LSTM model (3.8) falls into the general form (3.1), where the set of weights reads as

$$\Phi = \{W_f, U_f, b_f, W_i, U_i, b_i, W_r, U_r, b_r, W_z, U_z, b_z, U_o, b_o\}$$

and $d(\Phi) = b_r$. Indeed, if b_r is null, (3.8) admits the origin as equilibrium.

Before analyzing the stability properties of (3.8), let us point out the following bounds.

Lemma 3.1. *The gates f_k, i_k, and z_k, and the squashed input r_k of the shallow LSTM* (3.8) *can be bounded, for each component $j \in \{1, ..., n_c\}$, as*

$$
\begin{aligned}
0 < 1 - \check{\sigma}_f \leq [f_k]_j \leq \check{\sigma}_f < 1, && \text{(3.9a)}\\
0 < 1 - \check{\sigma}_i \leq [i_k]_j \leq \check{\sigma}_i < 1, && \text{(3.9b)}\\
0 < 1 - \check{\sigma}_z \leq [z_k]_j \leq \check{\sigma}_z < 1, && \text{(3.9c)}\\
-1 < -\check{\phi}_r \leq [r_k]_j \leq \check{\phi}_r < 1. && \text{(3.9d)}
\end{aligned}
$$

where

$$
\begin{aligned}
\check{\sigma}_f = \sigma(\|W_f \quad U_f \quad b_f\|_\infty), && \text{(3.10a)}\\
\check{\sigma}_i = \sigma(\|W_i \quad U_i \quad b_i\|_\infty), && \text{(3.10b)}\\
\check{\sigma}_z = \sigma(\|W_z \quad U_z \quad b_z\|_\infty), && \text{(3.10c)}\\
\check{\phi}_r = \phi(\|W_r \quad U_r \quad b_r\|_\infty). && \text{(3.10d)}
\end{aligned}
$$

Proof. See Appendix A.2.3. □

The bounds discussed in Lemma 3.1 allow to establish a relationship between the weights and the minimum and maximum values of the gates.

3.2.2 Sufficient conditions for the stability of shallow LSTMs

To establish the stability properties of (3.8), we need to find an invariant set $\mathcal{X}$ within which ISPS, ISS, and δISS can be assessed. Such invariant set is therefore defined in the following.

Proposition 3.1 (Invariant set of shallow LSTMs)**.** *Given Assumption 3.1, an invariant set of the shallow LSTM* (3.8) *is*

$$\mathcal{X} = \mathcal{C} \times \mathcal{H}, \tag{3.11a}$$

where the sets $\mathcal{C}$ and $\mathcal{H}$ read as

$$
\begin{aligned}
\mathcal{C} = \{c \in \mathbb{R}^{n_c} : \|c\|_\infty \leq \check{c}\}, && \text{(3.11b)}\\
\mathcal{H} = \{h \in \mathbb{R}^{n_c} : \|h\|_\infty \leq \check{h}\}, && \text{(3.11c)}
\end{aligned}
$$

with the component-wise bounds $\check{c}$ and $\check{h}$ being defined as

$$
\begin{aligned}
\check{c} = \frac{\check{\sigma}_i \check{\phi}_r}{1 - \check{\sigma}_f}, && \text{(3.12a)}\\
\check{h} = \phi(\check{c}) < 1, && \text{(3.12b)}
\end{aligned}
$$

and $\check{\sigma}_f$, $\check{\sigma}_i$, $\check{\sigma}_z$, and $\check{\phi}_r$ being defined in (3.10).

Proof. See Appendix A.2.4. □

We are now in the position of stating the stability conditions of shallow LSTMs with respect to the invariant set $\mathcal{X}$ (3.11) and the input set $\mathcal{U}$ (3.7).

Theorem 3.3. *A sufficient condition for the exponential ISPS of the shallow LSTM model* (3.8) *with respect to the sets $\mathcal{X}$ and $\mathcal{U}$ is that the matrix $\mathfrak{A}$, defined below, is Schur stable*

$$\mathfrak{A} = \begin{bmatrix} \check{\sigma}_f & \check{\sigma}_i \|U_r\|_2 \\ \check{\sigma}_z \check{\sigma}_f & \check{\sigma}_z \check{\sigma}_i \|U_r\|_2, \end{bmatrix}, \tag{3.13}$$

where $\check{\sigma}_f$, $\check{\sigma}_i$, $\check{\sigma}_z$, and $\check{\phi}_r$ are defined in (3.10).

Proof. See Appendix A.2.6. □

While Theorem 3.3 represents a noteworthy result, the associated condition on Φ does not come in the general form (3.2). In the following, conditions for the Schur stability of matrix $\mathfrak{A}$ are therefore formulated is such form.

Proposition 3.2 (ISPS of shallow LSTMs)**.** *The matrix $\mathfrak{A}$ defined in* (3.13) *is Schur stable if and only if*

$$\check{\sigma}_f + \check{\sigma}_z \check{\sigma}_i \|U_r\|_2 - 1 < 0. \tag{3.14}$$

Proof. See Appendix A.2.6. □

In view of Theorem 3.3, (3.14) is a sufficient condition for the exponential ISPS of the shallow LSTM model with respect to $\mathcal{X}$ and $\mathcal{U}$. Moreover, condition (3.14) now falls into the general form (3.2).

Corollary 3.2 (ISS of shallow LSTMs)**.** *If* (3.14) *holds and the bias b_r is null, then the LSTM model* (3.8) *is exponentially ISS with respect to the sets $\mathcal{X}$ and $\mathcal{U}$.*

Proof. Let us first notice that if b_r is null, the origin is an equilibrium of (3.8). It is easy to verify that, since $\|b_r\|_2 = 0$, the ISPS term ϱ, defined in (A.64c), is null. Therefore, the ISPS condition also implies the (stronger) ISS. □

Theorem 3.4. *A sufficient condition for the δISS of the shallow LSTM model* (3.8) *with respect to the sets $\mathcal{X}$ and $\mathcal{U}$ is that the matrix $\mathfrak{A}_\delta$, defined below, is Schur stable*

$$\mathfrak{A}_\delta = \begin{bmatrix} \check{\sigma}_f & \check{\alpha} \\ \check{\sigma}_z \check{\sigma}_f & \check{\sigma}_z \check{\alpha} + \frac{1}{4} \check{h} \|U_z\|_2 \end{bmatrix}, \tag{3.15a}$$

where $\check{\sigma}_f$, $\check{\sigma}_i$, $\check{\sigma}_z$, and $\check{\phi}_r$ are defined in (3.10)*, $\check{c}$ and $\check{h}$ are defined in* (3.12)*, and*

$$\check{\alpha} = \frac{1}{4}\check{c}\|U_f\|_2 + \check{\sigma}_i\|U_r\|_2 + \frac{1}{4}\check{\phi}_r\|U_i\|_2. \tag{3.15b}$$

Proof. See Appendix A.2.7. □

It is worth noticing that condition (3.15) does not fall into the generic form (3.2). In the following proposition we thus provide a sufficient conditions in such form which ensures the Schur stability of the matrix $\mathfrak{A}_\delta$.

Proposition 3.3 (δISS of shallow LSTMs)**.** *The matrix $\mathfrak{A}_\delta$ defined in* (3.15a) *is Schur stable if and only if*

$$\check{\sigma}_f + \check{\sigma}_z\check{\alpha} + \frac{1}{4}\check{h}\|U_z\|_2 - \frac{1}{4}\check{\sigma}_f\check{h}\|U_z\|_2 - 1 < 0, \tag{3.16a}$$

$$\frac{1}{4}\check{\sigma}_f\check{h}\|U_z\|_2 - 1 < 0. \tag{3.16b}$$

Proof. See Appendix A.2.8. □

In view of Theorem 3.4, (3.16) represent sufficient conditions for the exponential δISS of the shallow LSTM model with respect to the sets $\mathcal{X}$ and $\mathcal{U}$. Such conditions consist of a pair of nonlinear nonconvex inequalities on the weights of the LSTM network. Hence, here $\nu(\Phi)$ is a vector function

$$\nu_1(\Phi) < 0,$$
$$\nu_2(\Phi) < 0,$$

where $\nu_1(\Phi)$ is (3.16a) and $\nu_2(\Phi)$ is (3.16b).

Notably, despite being only sufficient conditions, it can be shown that, if the δISS conditions specified in Proposition 3.3 hold, then the system is also ISPS by Proposition 3.2. This relationship is formalized in the following corollary.

Proposition 3.4. *The shallow LSTM's δISS condition* (3.16) *implies the ISPS condition* (3.14).

Proof. See Appendix A.2.9. □

3.2.3 Deep LSTMs

When the model to be learned by the RNN is particularly complex, it is well known that using deep NNs generally yields better results [98]. At the price of a slower training procedure, once can therefore resort to the *deep*

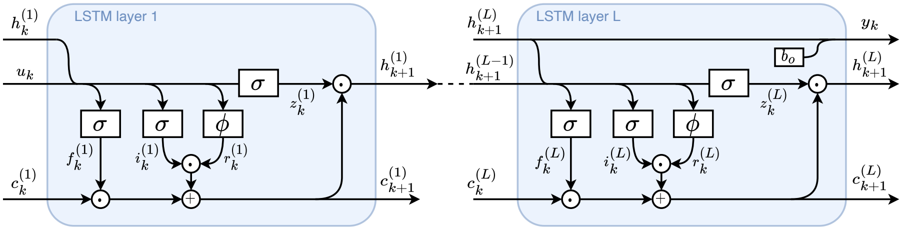

Figure 3.3: *Schematic of a deep LSTM network.*

(i.e., multi-layer) LSTMs. In this section, such architecture is described, and novel stability conditions for these models are devised.

Let us consider a deep LSTM with L layers, concatenated as illustrated in Figure 3.3. Each layer $l \in \{1, ..., L\}$ is of course described by the LSTM equation (3.8a), i.e.,

$$\begin{cases} c_{k+1}^{(l)} = f_k^{(l)} \circ c_k^{(l)} + i_k^{(l)} \circ r_k^{(l)} \\ h_{k+1}^{(l)} = z_k^{(l)} \circ \phi(c_{k+1}^{(l)}) \end{cases}, \tag{3.17a}$$

where, for the first layer ($l = 1$), the input $u_k^{(l)}$ is the network input u_k, while for the following layers (i.e., $l \in \{2, ..., L\}$), $u_k^{(l)}$ is the hidden state (at time $k+1$) of the previous layer. That is,

$$u_k^{(l)} = \begin{cases} u_k & \text{if } l = 1, \\ h_{k+1}^{(l-1)} & \text{if } l \in \{2, ..., L\}. \end{cases} \tag{3.17b}$$

Note that taking $u_k^{(l)} = h_{k+1}^{(l-1)} = z_k^{(l-1)} \circ \phi\big(f_k^{(l-1)} \circ c_k^{(l-1)} + i_k^{(l-1)} \circ r_k^{(l-1)}\big)$ instead of $u_k^{(l)} = h_k^{(l-1)}$ is a design choice which allows to avoid the accumulation of time delays throughout layers.

Each layer l has its own cell and hidden states, denoted as $c_k^{(l)} \in \mathbb{R}^{n_c^{(l)}}$ and $h_k^{(l)} \in \mathbb{R}^{n_c^{(l)}}$, respectively, where $n_c^{(l)}$ represents the number of neurons of the l-th layer. Therefore, each layer has a total of $n_x^{(l)} = 2n_c^{(l)}$ states. The l-th layer state vector reads as

$$x_k^{(l)} = \big[c_k^{(l)\prime}, h_k^{(l)\prime}\big]'. \tag{3.17c}$$

The output of the deep LSTM model is a linear combination of the hidden states of the last layer,

$$y_k = U_o h_k^{(l)} + b_o. \tag{3.17d}$$

The layers' gates $f_k^{(l)}$, $i_k^{(l)}$, and $z_k^{(l)}$, and the squashed input $r_k^{(l)}$ are defined, for each layer l, as

$$\begin{aligned} f_k^{(l)} &= \sigma(W_f^{(l)} u_k^{(l)} + U_f^{(l)} h_k^{(l)} + b_f^{(l)}), \\ i_k^{(l)} &= \sigma(W_i^{(l)} u_k^{(l)} + U_i^{(l)} h_k^{(l)} + b_i^{(l)}), \\ z_k^{(l)} &= \sigma(W_z^{(l)} u_k^{(l)} + U_z^{(l)} h_k^{(l)} + b_z^{(l)}), \\ r_k^{(l)} &= \phi(W_r^{(l)} u_k^{(l)} + U_r^{(l)} h_k^{(l)} + b_r^{(l)}). \end{aligned} \tag{3.17e}$$

It is worth noticing that, as evident from (3.17e), each layer features its own set of weights, denoted as

$$\Phi^{(l)} = \{W_f^{(l)}, U_f^{(l)}, b_f^{(l)}, W_i^{(l)}, U_i^{(l)}, b_i^{(l)}, W_r^{(l)}, U_r^{(l)}, b_r^{(l)}, W_z^{(l)}, U_z^{(l)}, b_z^{(l)}\}.$$

The set of weights of the deep LSTM includes the ones of all the layers of which it is composed and the matrices of the output transformation (3.17d), i.e.

$$\Phi = \bigcup_{l=1}^{L} \Phi^{(l)} \cup \{U_o, b_o\}.$$

The formulation of the deep LSTM (3.17) thus fits the general form (3.1), where the state vector is defined as

$$x_k = \left[x_k^{(1)\prime}, ..., x_k^{(L)\prime}\right]'. \tag{3.18}$$

Finally, let us point out that $d(\Phi)$ is here defined as $d = [b_r^{(1)\prime}, ..., b_r^{(L)\prime}]'$. Indeed, if d is null, $(0_{n_x^{(l)},1}, 0_{n_u^{(l)},1})$ is an equilibrium for each layer $l \in \{1, ..., L\}$, meaning that the origin is guaranteed to be an equilibrium of the deep LSTM system (3.17).

Remark 3.1. *The input of each layer, $u_k^{(l)}$, is unity-bounded. Indeed, recalling Assumption 3.1 and the input definition (3.17b), the hidden state satisfies $\|h_k^{(l)}\|_\infty < 1$, for any $l \in \{1, ..., L\}$.*

Remark 3.1 is justified by the fact that, for any $j \in \{1, ..., n_c^{(l)}\}$

$$[h_k^{(l)}]_j = [z_k^{(l)}]_j \, [\phi(c_{k+1}^{(l)})]_j.$$

Thus, in light of the boundedness of σ and ϕ, the two terms on the right-hand side are bounded in $(0, 1)$ and $(-1, 1)$, respectively, meaning that $[h_k^{(l)}]_j \in (-1, 1)$.

Thanks to Remark 3.1, the gates can be bounded in a similar way as done for shallow LSTMs.

Lemma 3.2. *Consider the deep LSTM model* (3.8)*. For any layer $l \in \{1, ..., L\}$, the gates $f_k^{(l)}$, $i_k^{(l)}$, and $z_k^{(l)}$, and the squashed input $r_k^{(l)}$, defined in* (3.17e)*, can be bounded component-wise $\forall j \in \{1, ..., n_c^{(l)}\}$ as follows*

$$
\begin{aligned}
0 < 1 - \check{\sigma}_f^{(l)} \leq [f_k^{(l)}]_j \leq \check{\sigma}_f^{(l)} < 1, && \text{(3.19a)}\\
0 < 1 - \check{\sigma}_i^{(l)} \leq [i_k^{(l)}]_j \leq \check{\sigma}_i^{(l)} < 1, && \text{(3.19b)}\\
0 < 1 - \check{\sigma}_z^{(l)} \leq [z_k^{(l)}]_j \leq \check{\sigma}_z^{(l)} < 1, && \text{(3.19c)}\\
-1 < -\check{\phi}_r^{(l)} \leq [r_k^{(l)}]_j \leq \check{\phi}_r^{(l)} < 1, && \text{(3.19d)}
\end{aligned}
$$

where

$$
\begin{aligned}
\check{\sigma}_f^{(l)} &= \sigma(\|W_f^{(l)} \quad U_f^{(l)} \quad b_f^{(l)}\|_\infty), && \text{(3.20a)}\\
\check{\sigma}_i^{(l)} &= \sigma(\|W_i^{(l)} \quad U_i^{(l)} \quad b_i^{(l)}\|_\infty), && \text{(3.20b)}\\
\check{\sigma}_z^{(l)} &= \sigma(\|W_z^{(l)} \quad U_z^{(l)} \quad b_z^{(l)}\|_\infty), && \text{(3.20c)}\\
\check{\phi}_r^{(l)} &= \phi(\|W_r^{(l)} \quad U_r^{(l)} \quad b_r^{(l)}\|_\infty). && \text{(3.20d)}
\end{aligned}
$$

Proof. See Appendix A.2.10. □

3.2.4 Sufficient conditions for the stability of deep LSTMs

We now define an invariant set for the full state vector (3.17c), with respect to which the stability properties can be defined.

Proposition 3.5 (Invariant set of deep LSTMs)**.** *An invariant set of the deep LSTM model* (3.17) *is*

$$
\mathcal{X} = \bigtimes_{l=1}^{L} \mathcal{X}^{(l)}, \qquad \text{(3.21a)}
$$

where

$$
\begin{aligned}
\mathcal{X}^{(l)} &= \mathcal{C}^{(l)} \times \mathcal{H}^{(l)}, && \text{(3.21b)}\\
\mathcal{C}^{(l)} &= \{c \in \mathbb{R}^{n_c^{(l)}} : \|c\|_\infty \leq \check{c}^{(l)}\}, && \text{(3.21c)}\\
\mathcal{H}^{(l)} &= \{h \in \mathbb{R}^{n_c^{(l)}} : \|h\|_\infty \leq \check{h}^{(l)}\}, && \text{(3.21d)}
\end{aligned}
$$

with the component-wise bounds $\check{c}^{(l)}$ and $\check{h}^{(l)}$ being defined as

$$
\begin{aligned}
\check{c}^{(l)} &= \frac{\check{\sigma}_i^{(l)} \check{\phi}_r^{(l)}}{1 - \check{\sigma}_f^{(l)}}, && \text{(3.22a)}\\
\check{h}^{(l)} &= \phi(\check{c}^{(l)}) < 1, && \text{(3.22b)}
\end{aligned}
$$

and $\check{\sigma}_f^{(l)}$, $\check{\sigma}_i^{(l)}$, $\check{\sigma}_z^{(l)}$, and $\check{\phi}_r^{(l)}$ being defined as in (3.20)*.*

Proof. See Appendix A.2.11. □

In view of the latter result, the sufficient conditions for the ISPS, ISS, and δISS of the deep LSTM architecture with respect to the invariant set $\mathcal{X}$ (3.21) and the input set $\mathcal{U}$ (3.7) can be stated.

Theorem 3.5 (ISPS of deep LSTMs)**.** *A sufficient condition for the exponential ISPS of the deep LSTM* (3.17) *with respect to the sets $\mathcal{X}$ and $\mathcal{U}$ is that each layer is ISPS by Proposition 3.2. That is,*

$$\check{\sigma}_f^{(l)} + \check{\sigma}_z^{(l)} \check{\sigma}_i^{(l)} \|U_r^{(l)}\|_2 - 1 < 0 \tag{3.23}$$

$\forall l \in \{1, ..., L\}$*, where* $\check{\sigma}_f^{(l)}$, $\check{\sigma}_i^{(l)}$, $\check{\sigma}_z^{(l)}$*, and* $\check{\phi}_r^{(l)}$ *are defined in* (3.20)

Proof. See Appendix A.2.12. □

From Theorem 3.5 it is evident that the proposed conditions for ISPS consist of L inequalities on weights of the network. In particular, for each layer l, the condition corresponds to an inequality in the form $\nu_l\big(\Phi^{(l)}\big) < 0$. By letting $\nu(\Phi) = \big[\nu_1\big(\Phi^{(1)}\big), ..., \nu_L\big(\Phi^{(L)}\big)\big]'$, one has that the ISPS condition given in Theorem 3.5 fits the general form (3.2).

Corollary 3.3 (ISS of deep LSTMs). *If, for each layer $l \in \{1, ..., L\}$, condition* (3.23) *holds and the bias $b_r^{(l)}$ is null, then the LSTM model* (3.17) *is ISS with respect to the sets $\mathcal{X}$ and $\mathcal{U}$.*

Proof. Let us first notice that if $b_r^{(l)}$ is null, the deep LSTM admits the origin as an equilibrium. Letting $b_r = \big[b_r^{(1)\prime}, ..., b_r^{(L)\prime}\big]'$, it is easy to notice that $\|b_r\|_2 = 0$. Therefore, the ISPS term ϱ, defined in (A.99c), is null, which means that the ISPS of the deep LSTM implies its ISS. □

Theorem 3.6 (δISS of deep LSTMs)**.** *A sufficient condition for the exponential δISS of the deep LSTM* (3.17) *with respect to the sets $\mathcal{X}$ and $\mathcal{U}$ is that each layer is δISS by Proposition 3.3. That is, $\forall l \in \{1, ..., L\}$, the following conditions are satisfied*

$$\check{\sigma}_f^{(l)} + \check{\sigma}_z^{(l)} \check{\alpha}^{(l)} + \frac{1}{4}\check{h}^{(l)} \|U_z^{(l)}\|_2 - \frac{1}{4}\check{\sigma}_f^{(l)} \check{h}^{(l)} \|U_z^{(l)}\|_2 - 1 < 0, \tag{3.24a}$$

$$\frac{1}{4}\check{\sigma}_f^{(l)} \check{h}^{(l)} \|U_z^{(l)}\|_2 - 1 < 0, \tag{3.24b}$$

where $\check{\alpha}^{(l)}$ is defined layer-wise as in (3.15b)*, i.e.*

$$\check{\alpha}^{(l)} = \frac{1}{4}\check{c}^{(l)} \|U_f^{(l)}\|_2 + \check{\sigma}_i^{(l)} \|U_r^{(l)}\|_2 + \frac{1}{4}\check{\phi}_r^{(l)} \|U_i^{(l)}\|_2, \tag{3.25}$$

with $\breve{\sigma}_f^{(l)}$, $\breve{\sigma}_i^{(l)}$, $\breve{\sigma}_z^{(l)}$, and $\check{\phi}_r^{(l)}$ being defined as in (3.20)*, and $\check{c}^{(l)}$ and $\check{h}^{(l)}$ as in* (3.22)*.*

Proof. See Appendix A.2.13. □

We point out that the conditions (3.24) fit the general form (3.2), since

$$\nu(\Phi) = \left[\nu_{1,1}(\Phi^{(1)}), \nu_{2,1}(\Phi^{(1)}), ..., \nu_{1,L}(\Phi^{(L)}), \nu_{2,L}(\Phi^{(L)})\right]',$$

where $\nu_{1,l}(\Phi^{(l)})$ and $\nu_{2,l}(\Phi^{(l)})$ read as (3.24a) and (3.24b), respectively. Hence $\nu(\Phi)$ consists of $2L$ nonlinear functions.

Remark 3.2. *As for shallow LSTMs, despite the proposed conditions are only sufficient, if the δISS conditions discussed in Theorem 3.6 hold, the deep LSTM is also ISPS by Theorem 3.5. This implication can be trivially proven applying Proposition 3.4 layer-wise.*

3.3 GRUs

In this section we present GRUs and analyze their stability properties. This RNN architecture has been conceived in [81] with the goal of combining the modeling capabilities allowed by the gating mechanism, with a simplification of the architecture, aimed at reducing the number of weights to be tuned and to speed up the training procedure. This balance makes GRUs suitable for learning dynamical system models to be used for control synthesis. This is reflected in this work, where the synthesis of model-based control laws will be conducted mainly on GRU models.

Following the same approach taken for LSTMs, we first describe shallow GRU models and discuss their stability properties. The proposed theoretical framework is then extended to the general case of deep GRUs.

Before entering into the details, we remind the reader of the unity-bounded input assumption (Assumption 3.1), which is still assumed to hold.

3.3.1 Shallow GRUs

Let us consider the following nonlinear discrete-time state-space system, which implements a shallow GRU model,

$$\begin{cases} x_{k+1} = z_k \circ x_k + (1 - z_k) \circ \phi\left(W_r\, u_k + U_r\, f_k \circ x + b_r\right) \\ y_k = U_o\, x_k + b_o \end{cases}. \tag{3.26a}$$

Such model consists of a single GRU layer, inspired to [81] yet strictly causal, and a linear output transformation. Note that $x_k \in \mathbb{R}^{n_x}$ denotes

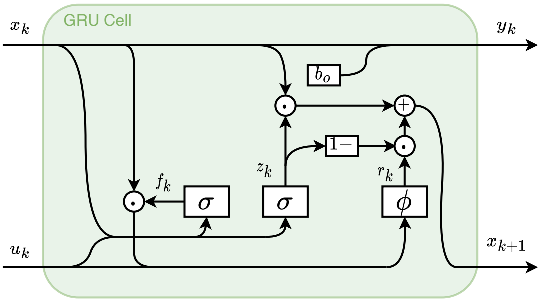

Figure 3.4: *Schematic of a shallow GRU network.*

the state of the shallow GRU, $u_k \in \mathbb{R}^{n_u}$ its input vector, and $y_k \in \mathbb{R}^{n_y}$ its output. The state dimensionality n_x matches the number of neurons n_c of the layer, i.e. $n_x = n_c$, which is a design choice of the model.

The terms z_k and f_k are named *update* and *forget* gates, respectively, and, analogously to LSTMs, they are functions of the inputs and states, squashed by the sigmoidal activation function:

$$\begin{aligned} z_k &= \sigma(W_z u_k + U_z x_k + b_z), \\ f_k &= \sigma(W_f u_k + U_f x_k + b_f). \end{aligned} \tag{3.26b}$$

For the sake of compactness, we may also denote $r_k = \phi(W_r\, u_k + U_r\, f_k \circ x + b_r)$, referred to as squashed input. The resulting architecture is schematically depicted in Figure 3.4. It is worth noticing that, compared to the shallow LSTM architecture (3.8), the GRU features one gate less, which explains why GRUs are deemed to have a simpler architecture[3].

The shallow GRU model (3.26) falls into the general form (3.1), where the set of weights to be trained is

$$\Phi = \{W_z, U_z, b_z, W_f, U_f, b_f, W_r, U_r, b_r, U_o, b_o\}$$

and $d(\Phi) = b_r$. Indeed, it is easy to notice that if b_r is null, the system (3.26) surely admits the origin as equilibrium.

At this point, let us introduce the following assumption concerning the boudnedness of the set of the model's initial states candidates.

Assumption 3.2. *The initial state of the shallow GRU* (3.26) *belongs to an arbitrarily large, but finite, set*

$$\mathcal{X} = \{x \in \mathbb{R}^{n_x} : \|x\|_\infty \leq \check{x}\}, \tag{3.27}$$

with $\check{x} \geq 1$.

[3] Even simpler GRU variants exist, such as minimal GRUs [99], where another gate is removed by setting $f_k = z_k$. This variant, however, is not here considered.

3.3.2 Characterization of shallow GRUs' state trajectories

To analyze the stability properties of GRUs, it is necessary to determine an invariant set from which the trajectories are confined. For this purpose, we show that $\mathcal{X}$ is an invariant set of the model. In addition, the trajectories of states within $\mathcal{X}$ are characterized.

Lemma 3.3 (Invariant set of shallow GRUs)**.** *The set $\mathcal{X}$, defined as in* (3.27) *with $\check{x} \geq 1$, is an invariant set for the shallow GRU model* (3.26)*. In particular, for any $u_k \in \mathbb{R}^{n_u}$, it holds that*

$$x_k \in \mathcal{X} \implies f(x_k, u_k) \in \mathcal{X}.$$

Proof. See Appendix A.2.14. □

It is worth highlighting that the invariant set described above is stated with respect to inputs extracted from $\mathbb{R}^{n_u} \supset \mathcal{U}$. Indeed, it is clear from the proof that the boundedness of the input is not required, so the lemma is formulated as generally as possible. This choice will prove useful when deep GRUs are considered.

As apparent from (3.27), the amplitude of the invariant set $\mathcal{X}$ depends on the bound on the initial state $\check{x}$. The minimal invariant set is henceforth denoted as

$$\tilde{\mathcal{X}} = \{x \in \mathbb{R}^{n_x} : \|x\|_\infty \leq 1\}, \tag{3.28}$$

and corresponds to $\check{x} = 1$. Later in this section we will show how such set enables significant simplifications of stability conditions. Before that, however, let us provide a fundamental lemma that allows us to characterize the trajectories of the GRU states.

Lemma 3.4. *Consider the shallow GRU* (3.26) *fed by any arbitrarily-bounded input sequence*[4]*. Then, for any initial state $x_0 \in \mathcal{X}$,*

i. if $x_0 \in \mathcal{X} \setminus \tilde{\mathcal{X}}$, then $\|x_k\|_\infty$ is strictly decreasing until $x_k \in \tilde{\mathcal{X}}$;

ii. there exists a finite $\bar{k} \geq 0$ such that $x_k \in \tilde{\mathcal{X}}$, $\forall k \geq \bar{k}$;

iii. each component of the state vector $[x_k]_j$ converges into the invariant set $\left[\tilde{\mathcal{X}}\right]_j = [-1, 1]$ in an exponential fashion.

Proof. See Appendix A.2.15. □

[4] By arbitrary-bounded input sequence we mean that there exists a finite $\check{u} \geq 1$, arbitrarily large, such that $\|u_\tau\|_\infty \leq \check{u}$ for any $\tau \in \{0, ..., k\}$. Moreover, note that in Lemma 3.4 we compactly denote the state vector as $x_k = x_k(x_0, u_{0:k})$.

Lemma 3.4 makes it clear that, since states converge in finite time into the invariant set $\tilde{\mathcal{X}}$, one could theoretically limit oneself to studying stability properties in such subset. Nonetheless, for the sake of generality, we will first present the results related to the invariant set $\mathcal{X}$, which leads to more conservative results. These results are then simplified by considering only the smaller invariant set $\tilde{\mathcal{X}} \subseteq \mathcal{X}$.

Lemma 3.5. *Consider the shallow GRU model* (3.26)*. If Assumption 3.1 holds, the gates z_k and f_k, and the squashed input r_k can be bounded over the invariant set $\mathcal{X}$ as*

$$0 < 1 - \check{\sigma}_z \leq [z_k]_j \leq \check{\sigma}_z < 1, \tag{3.29a}$$
$$0 < 1 - \check{\sigma}_f \leq [f_k]_j \leq \check{\sigma}_f < 1, \tag{3.29b}$$
$$-1 < -\check{\phi}_r \leq [r_k]_j \leq \check{\phi}_r < 1, \tag{3.29c}$$

for any component $j \in \{1, ..., n_c\}$, where

$$\check{\sigma}_z = \sigma(\|W_z \quad U_z \check{x} \quad b_z\|_\infty), \tag{3.30a}$$
$$\check{\sigma}_f = \sigma(\|W_f \quad U_f \check{x} \quad b_f\|_\infty), \tag{3.30b}$$
$$\check{\phi}_r = \phi(\|W_r \quad U_r \check{x} \quad b_r\|_\infty). \tag{3.30c}$$

Proof. See Appendix A.2.16. □

Remark 3.3. *The bounds of the gates described in Lemma 3.5 are conservative, as they are valid at any time instant, even when $x_0 \in \mathcal{X} \setminus \tilde{\mathcal{X}}$. However, Lemma 3.4 guarantees that the GRU states converge in finite time into the invariant set $\tilde{\mathcal{X}}$. Once inside such an invariant set, less conservative bounds to the gates can be computed as*

$$\tilde{\sigma}_z = \sigma(\|W_z \quad U_z \quad b_z\|_\infty), \tag{3.31a}$$
$$\tilde{\sigma}_f = \sigma(\|W_f \quad U_f \quad b_f\|_\infty), \tag{3.31b}$$
$$\tilde{\phi}_r = \phi(\|W_r \quad U_r \quad b_r\|_\infty), \tag{3.31c}$$

such that

$$0 < 1 - \tilde{\sigma}_z \leq [z_k]_j \leq \tilde{\sigma}_z < 1, \tag{3.32a}$$
$$0 < 1 - \tilde{\sigma}_f \leq [f_k]_j \leq \tilde{\sigma}_f < 1, \tag{3.32b}$$
$$-1 < -\tilde{\phi}_r \leq [r_k]_j \leq \tilde{\phi}_r < 1, \tag{3.32c}$$

is guaranteed $\forall j \in \{1, ..., n_c\}$ and for any $k \geq \bar{k}$, where $\bar{k} : x_{\bar{k}} \in \tilde{\mathcal{X}}$.

3.3.3 Sufficient conditions for the stability of shallow GRUs

Having defined an invariant set of the shallow GRU model, and having characterized its state trajectories, we are finally in the position to analyze its stability properties. Unless differently specified, these properties are intended to be stated with respect to the invariant set $\mathcal{X}$ (3.27) and the input set $\mathcal{U}$ (3.7).

Theorem 3.7 (ISPS of shallow GRUs)**.** *A sufficient condition for the exponential ISPS of the shallow GRU* (3.26) *with respect to the sets $\mathcal{X}$ and $\mathcal{U}$ is that*

$$\tilde{\sigma}_f \|U_r\|_\infty - 1 < 0, \tag{3.33}$$

where $\tilde{\sigma}_f$ is defined in (3.31b).

Proof. See Appendix A.2.17. □

Let us highlight that the ISPS sufficient condition (3.33) falls into the general form (3.2).

Corollary 3.4 (ISS of shallow GRUs)**.** *Under condition* (3.33)*, if the bias b_r is null, then the GRU* (3.26) *is exponentially ISS over the sets $\mathcal{X}$ and $\mathcal{U}$.*

Proof. First, we point out that if b_r is null, the origin is an equilibrium of (3.26). Then, since $\|b_r\|_\infty = 0$, the ISPS term ϱ is null, see (A.131c), which implies that the system is also ISS. □

Theorem 3.8 (δISS of shallow GRUs)**.** *A sufficient condition for the exponential δISS of the shallow GRU* (3.26) *with respect to the sets $\mathcal{X}$ and $\mathcal{U}$ is that*

$$\left(\frac{1}{4}\check{x}\|U_f\|_\infty + \check{\sigma}_f\right)\|U_r\|_\infty + \frac{1}{4}\frac{\check{x} + \check{\phi}_r}{1 - \check{\sigma}_z}\|U_z\|_\infty - 1 < 0, \tag{3.34}$$

where $\check{\sigma}_z$, $\check{\sigma}_f$, and $\check{\phi}_r$ are defined as in (3.30).

Proof. See Appendix A.2.18. □

It is worth noticing that condition stated in Theorem 3.8 guarantees the δISS of the system only inside the invariant set, and – depending on the scale $\check{x}$ of the invariant set – it might be conservative. Since Lemma 3.4 implies that the state trajectories converge into the (smaller) invariant set $\tilde{\mathcal{X}}$, one may think to relax condition (3.34) by assuming that the system is initialized within $\tilde{\mathcal{X}} \subseteq \mathcal{X}$. This observation leads to the following corollary.

Corollary 3.5 (Relaxed δISS of shallow GRUs)**.** *A relaxed condition for the δISS of the shallow GRU* (3.26) *with respect to the sets $\tilde{\mathcal{X}}$ and $\mathcal{U}$ is that*

$$\left(\frac{1}{4}\|U_f\|_\infty + \tilde{\sigma}_f\right)\|U_r\|_\infty + \frac{1}{4}\frac{1+\tilde{\phi}_r}{1-\tilde{\sigma}_z}\|U_z\|_\infty - 1 < 0, \tag{3.35}$$

where $\tilde{\sigma}_z$, $\tilde{\sigma}_f$, and $\tilde{\phi}_r$ are defined as in (3.31).

Proof. The corollary follows straightforwardly from Theorem (3.8), where $\check{x} = 1$. □

Notice that both (3.34) and (3.35) fit the generic form presented in (3.2). In the following remark, the relationship between these two sufficient conditions is discussed.

Remark 3.4. *The condition* (3.35) *involved by Corollary 3.5 is less conservative than the condition* (3.34) *required by Theorem 3.8. While Corollary 3.5 ensures the δISS with respect to $\tilde{\mathcal{X}} \subseteq \mathcal{X}$, it also allows to attain a similar (but weaker) δISS-related property also outside $\tilde{\mathcal{X}}$. In fact, when $x_{a,0} \in \mathcal{X} \setminus \tilde{\mathcal{X}}$ and/or $x_{b,0} \in \mathcal{X} \setminus \tilde{\mathcal{X}}$, it is not possible to show that, during the exponential convergence of $x_{a,k}$ and $x_{b,k}$ into $\tilde{\mathcal{X}}$ (Lemma 3.4), the condition* (3.35) *implies that the δISS relation* (2.6) *holds. However, as soon as both $x_{a,k} \in \tilde{\mathcal{X}}$ and $x_{b,k} \in \tilde{\mathcal{X}}$ – which is guaranteed by Lemma 3.4 to happen in finite time – the δISS property regularly applies.*

Finally, we point out that the δISS sufficient conditions proposed in Theorem 3.8 (and in Corollary 3.5) also imply the ISPS of the GRU. This relationship is formalized in the following proposition.

Proposition 3.6. *Both the δISS sufficient condition* (3.35) *and the relaxed condition* (3.34) *imply the ISPS condition* (3.33).

Proof. See Appendix A.2.19. □

3.3.4 Deep GRUs

In this section we describe the deep GRU models. As for LSTMs, the underlying idea is that adding layers generally lead to significant improvements of the modeling capabilities. Therefore, after having described the deep GRU architecture, the nontrivial problem of finding an invariant set is addressed, and sufficient conditions for the stability of deep GRU models are devised.

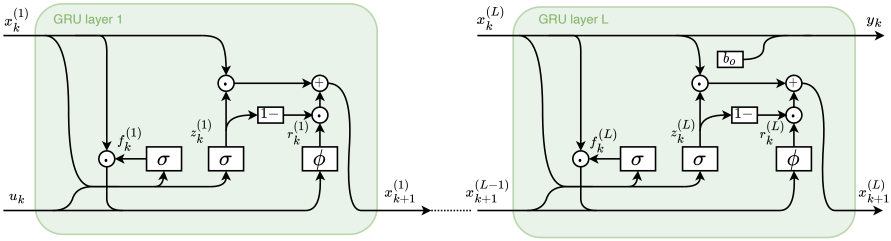

Figure 3.5: *Schematic of a deep GRU network.*

Consider a deep GRU composed of L GRU layers, concatenated as shown in Figure 3.5. Therefore, each layer $l \in \{1, ..., L\}$ is described by equation (3.26a), that is

$$x_{k+1}^{(l)} = z_k^{(l)} \circ x_k^{(l)} + (1 - z_k^{(l)}) \circ r_k^{(l)}, \tag{3.36a}$$

where $x_k^{(l)} \in \mathbb{R}^{n_c^{(l)}}$ denotes the state vector of the l-th layer with $n_c^{(l)}$ neurons. The state of the deep GRU is therefore a vector of dimension $n_x = \sum_{l=1}^{L} n_c^{(l)}$, defined as the concatenation of the states of all the layers, i.e.

$$x_k = \left[x_k^{(1)\prime}, x_k^{(2)\prime}, ..., x_k^{(L)\prime}\right]^\prime. \tag{3.36b}$$

The input of the l-th layer, denoted by $u_k^{(l)}$, is represented by model input itself u_k for the first layer ($l = 1$), while for the following layers it is defined as the updated state of the preceding layer. That is,

$$u_k^{(l)} = \begin{cases} u_k & \text{if } l = 1, \\ x_{k+1}^{(l-1)} & \text{if } l \in \{2, ..., L\}. \end{cases} \tag{3.36c}$$

The gates $z_k^{(l)}$ and $f_k^{(l)}$, and the squashed input $r_k^{(l)}$ of layer l are defined as

$$\begin{aligned} z_k^{(l)} &= \sigma\big(W_z^{(l)} u_k^{(l)} + U_z^{(l)} x_k^{(l)} + b_z^{(l)}\big), \\ f_k^{(l)} &= \sigma\big(W_f^{(l)} u_k^{(l)} + U_f^{(l)} x_k^{(l)} + b_f^{(l)}\big), \\ r_k^{(l)} &= \phi\big(W_r^{(l)} u_k^{(l)} + U_r^{(l)} f_k^{(l)} \circ x_k^{(l)} + b_r^{(l)}\big). \end{aligned} \tag{3.36d}$$

Finally, the output of the model y_k is defined as a linear combination of the states of the last layer only, i.e.

$$y_k = U_o x_k^{(L)} + b_o. \tag{3.36e}$$

Note that the deep GRU model described in (3.36) fits the generic form (3.1), and that it is characterized by the set of weights

$$\Phi = \bigcup_{l=1}^{L} \Phi^{(l)} \cup \{U_o, b_o\},$$

where $\Phi^{(l)}$ indicates the weights of the l-th layer, i.e.

$$\Phi^{(l)} = \big\{ W_z^{(l)}, U_z^{(l)}, b_z^{(l)}, W_f^{(l)}, U_f^{(l)}, b_f^{(l)}, W_r^{(l)}, U_r^{(l)}, b_r^{(l)} \big\}.$$

Finally, letting $d(\Phi) = \big[b_r^{(1)\prime}, ..., b_r^{(L)\prime} \big]'$, it holds that, if d is null, the origin is an equilibrium of the deep GRU (3.36).

As for shallow GRUs, we now introduce an assumption concerning the boundedness of the set of initial states candidates.

Assumption 3.3. *The initial state of the deep GRU* (3.36) *belongs to the set $\mathcal{X}$, defined as*

$$\mathcal{X} = \bigtimes_{l=1}^{L} \mathcal{X}^{(l)}, \tag{3.37a}$$

where, letting $\check{x}^{(l)} \geq 1$,

$$\mathcal{X}^{(l)} = \big\{ x^{(l)} \in \mathbb{R}^{n_c^{(l)}} : \| x^{(l)} \|_\infty \leq \check{x}^{(l)} \big\}. \tag{3.37b}$$

3.3.5 Characterization of deep GRUs' state trajectories

At this point, in order to analyze the stability properties of interest, an invariant set of the deep GRU model needs to be formulated.

Lemma 3.6 (Invariant set of deep GRUs)**.** *The set $\mathcal{X}$, defined as in* (3.37)*, is an invariant set of the deep GRU model* (3.36)*. That is, for any $u_k \in \mathbb{R}^{n_u}$, it holds that*

$$x_k \in \mathcal{X} \implies f(x_k, u_k) \in \mathcal{X}.$$

Proof. See Appendix A.2.20. □

A fundamental consequence of Lemma 3.6, which will be extensively exploited in the results that follow, is that, in light of (3.36c), for any u_k, the input of layers $l \in \{2, ..., L\}$ is bounded as

$$\| u_k^{(l)} \|_\infty \leq \check{x}^{(l-1)}. \tag{3.38}$$

A further observation is that, as for shallow GRUs, the amplitude of the invariant set $\mathcal{X}$ defined in (3.37) depends on the bounds $\check{x}^{(1)}, ..., \check{x}^{(L)}$. The smallest invariant set is denoted as

$$\tilde{\mathcal{X}} = \{x \in \mathbb{R}^{n_x} : \|x\|_\infty \leq 1\}, \tag{3.39}$$

which corresponds to (3.37) with $\check{x}^{(1)} = ... = \check{x}^{(L)} = 1$. We now show an extension of Lemma 3.4 to deep GRUs, which allows to characterize the trajectories of the deep GRUs' states.

Lemma 3.7. *Consider the deep GRU* (3.36) *fed by any arbitrarily-bounded input sequence. Then, for any initial state* $x_0 = [x_0^{(1)\prime}, ..., x_0^{(L)\prime}]' \in \mathcal{X}$*,*

i. for each layer $l \in \{1, ..., L\}$*, if* $x_0^{(l)} \in \mathcal{X}^{(l)} \backslash \tilde{\mathcal{X}}^{(l)}$*, then* $\|x_k^{(l)}\|_\infty$ *is strictly decreasing until* $x_k^{(l)} \in \tilde{\mathcal{X}}^{(l)}$*, and hence* $\|x_k\|_\infty$ *is strictly decreasing until* $x_k \in \tilde{\mathcal{X}}$*;*

ii. there exists a finite $\bar{k} \geq 0$ *such that* $x_k \in \tilde{\mathcal{X}}$, $\forall k \geq \bar{k}$*;*

iii. each component of the state vector $[x_k]_j$ *converges into its invariant set* $\big[\tilde{\mathcal{X}}\big]_j = [-1, 1]$ *in an exponential fashion.*

Proof. See Appendix A.2.21. □

In view of Lemma 3.7, we can now provide bounds on the gates of the network, which will prove useful for devising the stability conditions.

Lemma 3.8. *Consider the deep GRU* (3.36)*. If Assumption* 3.1 *holds, for each layer* $l \in \{1, ..., L\}$*, the gates* $z_k^{(l)}$ *and* $f_k^{(l)}$*, and the squashed input* $r_k^{(l)}$*, can be bounded over the invariant set* $\mathcal{X}$ *as*

$$0 < 1 - \check{\sigma}_z^{(l)} \leq [z_k^{(l)}]_j \leq \check{\sigma}_z^{(l)} < 1, \tag{3.40a}$$

$$0 < 1 - \check{\sigma}_f^{(l)} \leq [f_k^{(l)}]_j \leq \check{\sigma}_f^{(l)} < 1, \tag{3.40b}$$

$$-1 < -\check{\phi}_r^{(l)} \leq [r_k^{(l)}]_j \leq \check{\phi}_r^{(l)} < 1, \tag{3.40c}$$

for any component $j \in \{1, ..., n_c^{(l)}\}$*. Recalling* (3.37b)*, these bounds can be computed as*

$$\check{\sigma}_z^{(l)} = \sigma(\|W_z^{(l)}\check{x}^{(l-1)} \quad U_z^{(l)}\check{x}^{(l)} \quad b_z^{(l)}\|_\infty), \tag{3.41a}$$

$$\check{\sigma}_f^{(l)} = \sigma(\|W_f^{(l)}\check{x}^{(l-1)} \quad U_f^{(l)}\check{x}^{(l)} \quad b_f^{(l)}\|_\infty), \tag{3.41b}$$

$$\check{\phi}_r^{(l)} = \phi(\|W_r^{(l)}\check{x}^{(l-1)} \quad U_r^{(l)}\check{x}^{(l)} \quad b_r^{(l)}\|_\infty), \tag{3.41c}$$

where $\check{x}^{(0)} = \sup \|u_k\|_\infty = 1$.

Proof. See Appendix A.2.22. □

Remark 3.5. *The bounds of the gates proposed in Lemma 3.8 are conservative, as they are valid at any time instant, even when $x_0 \in \mathcal{X} \setminus \tilde{\mathcal{X}}$. On the other hand, Lemma 3.7 ensures the finite-time convergence of x_k into the (smaller) invariant set $\tilde{\mathcal{X}}$ defined in* (3.39)*. Once x_k enters such set, less conservative bounds hold. In particular, for each $l \in \{1, ..., L\}$, these relaxed bounds can be computed as*

$$\tilde{\sigma}_z^{(l)} = \sigma(\|W_z^{(l)} \quad U_z^{(l)} \quad b_z^{(l)}\|_\infty), \tag{3.42a}$$

$$\tilde{\sigma}_f = \sigma(\|W_f^{(l)} \quad U_f^{(l)} \quad b_f^{(l)}\|_\infty), \tag{3.42b}$$

$$\tilde{\phi}_r = \phi(\|W_r^{(l)} \quad U_r^{(l)} \quad b_r^{(l)}\|_\infty), \tag{3.42c}$$

such that

$$0 < 1 - \tilde{\sigma}_z^{(l)} \leq [z_k^{(l)}]_j \leq \tilde{\sigma}_z^{(l)} < 1, \tag{3.43a}$$

$$0 < 1 - \tilde{\sigma}_f^{(l)} \leq [f_k^{(l)}]_j \leq \tilde{\sigma}_f^{(l)} < 1, \tag{3.43b}$$

$$-1 < -\tilde{\phi}_r^{(l)} \leq [r_k^{(l)}]_j \leq \tilde{\phi}_r^{(l)} < 1, \tag{3.43c}$$

$\forall j \in \{1, ..., n_c^{(l)}\}$ *and for any $k \geq \bar{k}$, where $\bar{k}$ is such that $x_{\bar{k}} \in \tilde{\mathcal{X}}$.*

3.3.6 Sufficient conditions for the stability of deep GRUs

At this point we are in the position to introduce the conditions that ensure the stability of deep GRUs. Notice that, unless differently specified, such stability properties are stated with respect to the invariant set $\mathcal{X}$ (3.37) and the input set $\mathcal{U}$ (3.7).

Theorem 3.9 (ISPS of deep GRUs)**.** *A sufficient condition for the exponential ISPS of the deep GRU* (3.36) *with respect to the sets $\mathcal{X}$ and $\mathcal{U}$ is that*

$$\tilde{\sigma}_f^{(l)} \|U_r^{(l)}\|_\infty - 1 < 0 \tag{3.44}$$

$\forall l \in \{1, ..., L\}$, *where $\tilde{\sigma}_f^{(l)}$ is defined as* (3.42b)*.*

Proof. See Appendix A.2.23. □

It is worth pointing out that condition (3.44) corresponds to requiring that every layer satisfies the ISPS sufficient condition (3.33). Moreover, as stated in the following corollary, these conditions also imply the ISS when the bias is null.

Corollary 3.6 (ISS of deep GRUs)**.** *If, for each layer $l \in \{1, ..., L\}$, condition* (3.44) *holds and the bias $b_r^{(l)}$ is null, then the GRU is exponentially ISS with respect to the sets $\mathcal{X}$ and $\mathcal{U}$.*

Proof. Since $b_r^{(l)}$ is null, the deep GRU model (3.36) admits the origin as equilibrium. Moreover, letting $b_r = \left[b_r^{(1)\prime}, ..., b_r^{(L)\prime}\right]'$, and observing that $\|b_r\|_\infty = 0$, the ISPS term ϱ is null, see (A.160c), meaning that the system is also ISS. □

Theorem 3.10 (δISS of deep GRUs)**.** *A sufficient condition for the exponential δISS of the shallow GRU* (3.36) *with respect to the sets $\mathcal{X}$ and $\mathcal{U}$ is that*

$$\left(\frac{1}{4}\check{x}^{(l)}\|U_f^{(l)}\|_\infty + \check{\sigma}_f^{(l)}\right)\|U_r^{(l)}\|_\infty + \frac{1}{4}\frac{\check{x}^{(l)} + \check{\phi}_r^{(l)}}{1 - \check{\sigma}_z^{(l)}}\|U_z^{(l)}\|_\infty - 1 < 0, \tag{3.45}$$

$\forall l \in \{1, ..., L\}$*, where* $\check{\sigma}_z^{(l)}$, $\check{\sigma}_f^{(l)}$, *and* $\check{\phi}_r^{(l)}$ *are defined as in* (3.41).

Proof. See Appendix A.2.24. □

Note that condition (3.45) can be overly conservative, especially when the set $\mathcal{X}$ is large, since the bounds on the gates increase with the amplitude of such set, see (3.41). Less conservative conditions, however, can be provided if the δISS is stated with respect to the smaller invariant set $\tilde{\mathcal{X}} \subseteq \mathcal{X}$ defined in (3.39).

Corollary 3.7 (Relaxed δISS of deep GRUs)**.** *A relaxed condition for the exponential δISS of the deep GRU* (3.36) *with respect to the sets $\tilde{\mathcal{X}}$ and $\mathcal{U}$ is that*

$$\left(\frac{1}{4}\|U_f^{(l)}\|_\infty + \tilde{\sigma}_f^{(l)}\right)\|U_r^{(l)}\|_\infty + \frac{1}{4}\frac{1 + \tilde{\phi}_r^{(l)}}{1 - \tilde{\sigma}_z^{(l)}}\|U_z^{(l)}\|_\infty - 1 < 0, \tag{3.46}$$

$\forall l \in \{1, ..., L\}$*, where* $\tilde{\sigma}_z^{(l)}$, $\tilde{\sigma}_f^{(l)}$, *and* $\tilde{\phi}_r^{(l)}$ *are defined as in* (3.42).

Proof. The corollary straightforwardly follows from Theorem 3.10. Indeed, by taking $\mathcal{X} = \tilde{\mathcal{X}}$, i.e. $\check{x}^{(1)} = ... = \check{x}^{(L)} = 1$, condition (3.45) is equivalent to (3.46). □

Similarly to what was pointed out in Remark 3.4, the condition proposed in Corollary 3.7 is weaker than the one proposed in Theorem 3.10. While the latter condition implies that the definition of δISS holds throughout the set $\mathcal{X}$, the former condition merely ensures that the definition of δISS holds within the set $\tilde{\mathcal{X}} \subseteq \mathcal{X}$. In any case, when the deep GRU is initialized outside $\tilde{\mathcal{X}}$, the relaxed condition (3.46) allows to establish a kind of weaker δISS, that is, a δISS that holds only after both state trajectories have entered the set $\tilde{\mathcal{X}}$. This convergence is guaranteed to happen exponentially and in finite time by Lemma 3.7.

Note also that the sufficient conditions reported in Theorem 3.9, Theorem 3.10, and Corollary 3.7 fall in the general form (3.2). More specifically, each of these sufficient conditions consists of L nonlinear inequalities, and hence

$$\nu(\Phi) = \big[\nu_1(\Phi^{(1)}), ..., \nu_L(\Phi^{(L)})\big]'$$

where $\nu_l(\Phi^{(l)})$ is described by (3.44), (3.45), and (3.46), respectively.

Remark 3.6. *The δISS sufficient conditions proposed in Theorem 3.10, as well as the conditions proposed in Corollary 3.7, can be easily shown to imply the ISPS condition illustrated in Theorem 3.9. This relationship can be proven by applying Proposition 3.6 layer-wise.*

3.4 Summary

In this chapter the RNN architectures considered in this thesis have been described. In particular, the NNARX, LSTM, and GRU architectures have been formulated as strictly proper discrete-time nonlinear dynamical systems. For each of these architectures, theoretically-sound sufficient conditions guaranteeing their ISPS, ISS, and δISS have been proposed.

As highlighted, these properties are critical to ensure the safety, robustness and generalizability of trained RNN models, yet limited contributions on this topic are available in the literature to date. The goal of this chapter has been to fill these gaps with a unified framework and consistent notation, both for single-layer and multi-layer architectures. All the proposed sufficient conditions boil down to a vector of nonlinear inequalities on the RNN's weights Φ, i.e.

$$\nu(\Phi) < 0.$$

These sufficient conditions can be used a-posteriori to check whether the ISPS, ISS, or δISS of an already trained RNN can be guaranteed, or – as discussed in the next chapter – they can be imposed during the training procedure to a-priori guarantee the stability of the RNNs being trained.

CHAPTER 4

Training procedure

In Section 3, the RNN architectures used in this thesis for black-box system identification and control have been discussed. In order for them to approximate dynamical systems, RNNs must undergo the so-called training procedure, in which the weights leading to the “best” modeling performances are estimated. It is widely known that the training procedure can be time-consuming, computationally-intensive, and experience-demanding. Indeed, the training approach and its design parameters, i.e. the so-called hyperparameters, are generally crucial for a satisfactory outcome of the procedure. While an exhaustive description of the available training procedures is beyond the scope of this thesis, the aim of this chapter is threefold: (*i*) to provide a general overview of the goals to be pursued and the issues to be addressed by the training procedure, (*ii*) to detail our proposed procedure for training provenly-stable RNNs, and (*iii*) to provide guidelines for the choice of the hyperparameters for such procedure.

4.1 Introduction to the training of RNNs

The general purpose of the training procedure is to make the RNN approximate a specific unknown dynamical system, of which only input-output data are available. These data are generally collected during an ad-hoc experiment campaign, or can be collected during online system operations, and in either case they must carry sufficient information on the plant. The data consist of pairs of sequences, namely the applied input sequence and the measured output sequence. For the sake of simplicity of exposition, let us here consider a single experiment, in which the input $u_{0:T}$ is applied and the output sequence $y_{0:T}$ is correspondingly measured.

The training problem consists in finding the set of optimal weights Φ^o such that the output of the free-run simulation[1] of the RNN (3.1), i.e. $y_{0:T}(x_0, u_{0:T})$, is as close as possible to the measured output sequence $y_{0:T}$, without incurring in overfitting. In order to conduct the training procedure, it is necessary to define a metrics that quantifies the goodness of the RNN simulation with respect to the measured output.

A further well-known challenge is the choice of the optimization algorithm, as well as its hyperparameters, to solve the underlying fitting problem. Such problem is indeed deeply nonlinear, non-convex and large-scale, since the dimensionality of the weights grows rapidly with the number of layers and neurons per layer. This optimization procedure has been historically carried out numerically via the so-called Gradient Descent (GD) methods [14]. However, its application to RNNs, which also features the time domain, is not straightforward and thus motivated DL researchers to develop several training strategies, briefly summarized below.

Historically, NNARX models have been trained by minimizing the one-step ahead prediction error via GD, see e.g. [16,100]. This was made possible by the fact that, while in general for RNN models the state x_k (needed to predict the future state y_{k+1}) is not known, in NNARX models it consists of a collection of past inputs and outputs, and is therefore known. Being based on one-step ahead prediction techniques, however, this method could lead to models with inadequate long-term prediction capabilities. Moreover, it can not be used with other RNN architectures, such as LSTM and GRU, for which the state vector is not known or defined.

To train these architectures, instead, the most common approach is that of Back-Propagation Through Time (BPTT) [13]. This approach consists, roughly speaking, in an iterative numerical procedure in which, at each iteration, the following computations are carried out

i. *Forward Pass* – the free-run simulation of the RNN (3.1), denoted as $y_{0:T}(x_0, u_{0:T}) = y_{0:T}(x_0, u_{0:T}; \Phi)$, is first evaluated;

[1] The term *free-run* simulation is a deep learning jargon for the open-loop simulation of the system (3.1), fed by some known input sequence, in this case $u_{0:T}$.

ii. *Gradient Computation* – the gradient of the error between the RNN simulation $y_{0:T}(x_0, u_{0:T}; \Phi)$ and the ground truth $y_{0:T}$ with respect to Φ is computed, e.g. using Automatic Differentiation software tools;

iii. *Backward Pass* – a gradient descent step is taken towards the direction minimizing the gradient, and Φ is updated.

The procedure described above is repeated until convergence.

Although BPTT was originally thought to be an inefficient option because of the exploding and vanishing gradient problems that plague vanilla RNNs [77], the advent of GRUs and LSTMs, designed precisely to avoid such problems, has proven the capabilities of the method. In particular, a variant of BPTT known as Truncated Back-Propagation Through Time (TBPTT) is now widely used to train RNNs for long-term time-series forecasting and system identification, see [13] and references therein.

As described in Section 4.2, TBPTT consists of splitting the training sequences $u_{0:T}$ and $y_{0:T}$ into many shorter partially-overlapping subsequences, on which a batched BPTT is applied. This approach yields several advantages. First, it better scales with the length of the input-output data, as the forward pass and gradient computation are carried out on shorter subsequences and in a parallel fashion. Secondly, the inherently stochastic[2] nature of TBPTT generally allows to better escape local minima, making the training algorithm more robust with respect to the random initial weights, and to mitigate for the overfitting phenomenon, at the price of a generally longer training time.

Before going into the details of the adopted TBPTT-based training procedure, let us point out that several other approaches have been proposed in the literature. For example, in [101] the authors propose an Extended Kalman Filter-based training procedure for single-layer vanilla RNNs, while in [102, 103] the idea of training RNNs via evolutionary algorithms is explored. A novel algorithm named Forward Propagation Through Time algorithm has been recently proposed in [104], with the aim of speeding up the training procedure and obtaining networks with better generalizability and performance. However, the TBPTT approach is herein considered since it is currently the standard for RNN training, particularly for GRUs and LSTMs. There is consensus that, in the majority of cases, this training approach provides satisfactory outcomes, and that there is considerable potential to further improve performances with targeted adjustments. To mention a few of them, accelerated optimization algorithms such as Adam [105] and

[2] The term “stochastic” is here used in its deep learning acception, and indicates the fact that the training procedure is performed on randomized batches, see [14].

RMSProp [106] (as well as their variants) can be used to speed up the convergence; a suitable weights initialization can robustify the training procedure [107]; dropout can drastically improve the generalization capabilities and avoid overfitting [108]; architecture adjustments can be taken reduce the number of weights and the parallelizability of the forward pass [109].

4.2 Training provenly stable RNNs via TBPTT

In this section we describe the adopted training procedure, which is a standard TBPTT procedure with the addition of a suitable regularization term that enforces the stability of the RNN to be trained. Before continuing, it is however fundamental to point out that enforcing the stability of the RNN being trained only makes sense if the system generating the input-output data enjoys the very same property. The stability of the system may be known from physical insights, or numerically assessed on the input-output trajectories, as shown in [58]. In what follows, as also discussed in Remark 4.2, the stability (in a sense better specified below) of the underlying system generating the data is assumed.

We now consider the problem of training the RNN $\Sigma(\Phi)$, described by the the state-space form (3.1). Recalling the Notation Addendum 2.1, for clarity we denote with

$$y_k(x_0, u_{0:k}; \Phi) = y_k(x_0, u_{0:k}; \Sigma(\Phi))$$

the output at time k of the RNN initialized in the state x_0, and fed by the input sequence $u_{0:k}$, so as to highlight the dependence of the output on the weights Φ. Moreover, we slightly stretch the Notation Addendum 2.1 and denote by

$$y_{k_1:k_2}(x_0, u_{0:k_2}; \Phi) = \big\{ y_{k_1}(x_0, u_{0:k_1}; \Phi), ..., y_{k_2}(x_0, u_{0:k_2}; \Phi) \big\} \tag{4.1}$$

the output sequence of the RNN model between instant k_1 and k_2.

The proposed training procedure assumes that three pairs of input-output sequences[3] are available, namely

i. One pair for *training*, indicated by $\mathcal{D}_{\text{tr}} = (u_{\text{tr},0:T_{\text{tr}}}, y_{\text{tr},0:T_{\text{tr}}})$;

ii. One pair for *validation*, indicated by $\mathcal{D}_{\text{val}} = (u_{\text{val},0:T_{\text{val}}}, y_{\text{val},0:T_{\text{val}}})$;

iii. One pair for the *independent test*, indicated by $\mathcal{D}_{\text{te}} = (u_{\text{te},0:T_{\text{te}}}, y_{\text{te},0:T_{\text{te}}})$.

[3] This standard convention allows for simpler notation in the following, but can be readily extended to the case of multiple input-output sequences (e.g. collected through a test-campaign), or to the case of a single one.

As mentioned, it is assumed that these sequences are sufficiently informative, that is, they adequately explore the operational region of interest and excite the relevant frequencies of the system to be learned[4].

According to the TBPTT principle, we then randomly extract shorter and partially overlapping subsequences from the above-described sequences. Specifically, we extract N_{tr} training subsequences from $\mathcal{D}_{\text{tr}}$ and N_{val} subsequences from $\mathcal{D}_{\text{val}}$, all having the same fixed length $T_s < T_{\text{tr}}, T_{\text{val}}$. Note that N_{tr} and T_s are hyperparameters that need to be carefully selected, as later discussed. The extracted subsequences are denoted as

$$\begin{aligned} u_{0:T_s}^{\{i\}} &= u_{\text{tr},k_i:k_i+T_s} \\ y_{0:T_s}^{\{i\}} &= y_{\text{tr},k_i:k_i+T_s} \end{aligned} \tag{4.2a}$$

for the training dataset, where $i \in \mathcal{I}_{\text{tr}}$ is the index associated to each training subsequence, and k_i is generated randomly such that $k_i + T_s \leq T_{\text{tr}}$. The set of training subsequences is indexed by

$$\mathcal{I}_{\text{tr}} = \{1, ..., N_{\text{tr}}\}. \tag{4.2b}$$

Analogously, the validation subsequences are defined as

$$\begin{aligned} u_{0:T_s}^{\{i\}} &= u_{\text{val},k_i:k_i+T_s} \\ y_{0:T_s}^{\{i\}} &= y_{\text{val},k_i:k_i+T_s} \end{aligned} \tag{4.3a}$$

where again each subsequence is indexed by $i \in \mathcal{I}_{\text{val}}$, with

$$\mathcal{I}_{\text{val}} = \{N_{\text{tr}} + 1, ..., N_{\text{tr}} + N_{\text{val}}\}, \tag{4.3b}$$

and the index k_i is randomly generated so that $k_i + T_s \leq T_{\text{val}}$. The independent test set, instead, is not divided into shorter subsequences as its only purpose is to objectively assess the performances of the RNN after the training procedure.

The ultimate goal of the training procedure is to minimize an index quantifying the fitting quality of the RNN over the training set $\mathcal{I}_{\text{tr}}$, subject to the stability of the RNN model, i.e. $\nu(\Phi) < 0$, see (3.2). The training procedure could thus be naively stated as an optimization problem

$$\begin{aligned} \Phi^o = \arg\min_{\Phi} & \operatorname{MSE}(\mathcal{I}_{\text{tr}}; \Phi) \\ \text{s.t.} \quad & \nu(\Phi) < 0 \end{aligned} \tag{4.4}$$

[4] Often the pair $(u_{0:T}, y_{0:T})$ is obtained through an open-loop experiment on the system to be identified. In this case, the design of an input sequence $u_{0:T}$ that satisfies the criteria of informativeness and persistence of excitation is known as design of experiment problem [2].

where $\mathrm{MSE}(\mathcal{I}_{\mathrm{tr}}; \Phi)$ is the commonly used Mean Square Error (MSE) of the RNN over $\mathcal{I}_{\mathrm{tr}}$, defined in Section 4.2.1. The optimization problem (4.4) cannot be solved directly, mainly due to the large number of optimization variables, to the presence of the nonlinear and non-convex stability constraint, and – especially – to the generally large number of training sequences in the set $\mathcal{I}_{\mathrm{tr}}$.

For this reason, an iterative method is applied to solve (4.4). Hence, at each iteration – referred to as *epoch* in the deep learning jargon – the training set $\mathcal{I}_{\mathrm{tr}}$ is firstly randomly partitioned into batches, i.e., into B random and non overlapping subsets $\mathcal{I}_{\mathrm{tr}}^{\{1\}}$, ..., $\mathcal{I}_{\mathrm{tr}}^{\{B\}}$. Then, for each of these batches, the loss function $\mathcal{L}$ is evaluated as

$$\mathcal{L}(\mathcal{I}_{\mathrm{tr}}^{\{b\}}, \Phi) = \mathrm{MSE}(\mathcal{I}_{\mathrm{tr}}^{\{b\}}; \Phi) + \rho(\nu(\Phi)), \tag{4.5}$$

where $\mathrm{MSE}(\mathcal{I}_{\mathrm{tr}}^{\{b\}}; \Phi)$ measures the fitting quality of the RNN onto the b-th training batch, and $\rho(\nu(\Phi))$ is a regularization term whose purpose is to steer $\nu(\Phi)$ towards negative values, i.e. towards the RNN guaranteed-stability region[5]. Suitable candidate functions $\rho(\cdot)$ are reported later this chapter in Section 4.2.2. Then, in the spirit of gradient descent-based minimization algorithms, the gradient of $\mathcal{L}$ with respect to Φ is computed via Automatic Differentiation softwares, and Φ is updated towards the minimum gradient direction[6]. At the end of each epoch, the performances of the RNN are evaluated on the validation set $\mathcal{I}_{\mathrm{val}}$, e.g. by computing $\mathrm{MSE}(\mathcal{I}_{\mathrm{val}}; \Phi)$. The training is stopped when the RNN satisfies the desired stability condition, i.e. $\nu(\Phi) < 0$, and the performance metrics over the validation set stops improving. This procedure yields the weights of the trained network, indicated by $\Phi^\star$.

At the end of the training procedure, the performances of the trained RNN are quantified on the independent test dataset $\mathcal{D}_{\mathrm{te}}$. Unlike the validation dataset, the test dataset is not used during the training procedure, so it allows RNN's performances to be objectively evaluated on new data, obtaining a figure of merit less prone to overfitting. Therefore, the trained RNN model is simulated using the random initial state x_0 and the test input sequence $u_{\mathrm{te},0:T_{\mathrm{te}}}$, and the quality of fit is evaluated using the FIT performance index [%], defined as[7]

$$\mathrm{FIT} = 100 \left(1 - \frac{\sum_{k=\tau_w}^{T_{\mathrm{te}}} \| y_k(x_0, u_{\mathrm{te},0:k}; \Phi^\star) - y_{\mathrm{te},k} \|_2}{\sum_{k=\tau_w}^{T_{\mathrm{te}}} \| y_{\mathrm{te},k} - y_{\mathrm{te,avg}} \|_2} \right), \tag{4.6}$$

[5] We recall that, according to (3.2), $\nu(\Phi) < 0$ implies the ISPS, ISS, or δISS of the RNN.

[6] To this end, different GD-based strategies could be adopted, such as traditional Gradient Descent [14], or the more efficient Adam [105] and RMSProp [106].

[7] The term τ_w denotes the washout period, an hyperparameter introduced later this chapter.

Algorithm 1 Training Algorithm

```
Require: Training subsequences (4.2) and validation subsequences (4.3)
  Initialize the weights $\Phi$
  for epoch $e = 1, ..., E$ do
      Randomly partition $\mathcal{I}_{\text{tr}}$ into batches $\mathcal{I}_{\text{tr}}^{\{1\}}, ..., \mathcal{I}_{\text{tr}}^{\{B\}}$
      for batch $b = 1, ..., B$ do
          Compute the loss $\mathcal{L}(\mathcal{I}_{\text{tr}}^{\{b\}}, \Phi)$ using random initial states      ▷ Forward pass
          Compute its gradient w.r.t. $\Phi$, i.e. $\nabla_\Phi \mathcal{L}(\mathcal{I}_{\text{tr}}^{\{b\}}, \Phi)$      ▷ Gradient computation
          Update $\Phi$ using gradient descent algorithms      ▷ Backward pass
      end for
      Compute the validation metrics $\text{MSE}(\mathcal{I}_{\text{val}}; \Phi)$
      if $\nu(\Phi) < 0$ and the validation metrics stops improving then
          Stop the training procedure
      end if
  end for
  Assess the performances on the test dataset computing FIT
```

where $y_{\text{te,avg}}$ is the empirical average of the output sequence $y_{\text{te},0:k}$. A FIT index close to 100% indicates a good accuracy of the RNN on the test dataset.

The resulting training procedure is summarized in Algorithm 1. To conclude the description of the training procedure, it is necessary to describe the two terms appearing in the loss function (4.5).

4.2.1 Mean Square Error

To define the MSE of the RNN $\Sigma(\Phi)$ over a batch, let us introduce the MSE over the generic input-output sequences $(u_{0:T}^{\{i\}}, y_{0:T}^{\{i\}})$ of length T. To this end, the RNN is initialized in a random initial state $x_0 \in \mathcal{X}$, and it is fed by the input sequence $u_{0:T}^{\{i\}}$, thus obtaining the output sequence $y_{0:T}(x_0, u_{0:T}^{\{i\}}; \Phi)$. The MSE is then defined as

$$\begin{aligned}\text{MSE}(y_{\tau_w:T}(x_0, u_{0:T}^{\{i\}}; \Phi), y_{\tau_w:T}^{\{i\}}) &= \frac{1}{n_y} \frac{1}{T - \tau_w + 1} \sum_{k=\tau_w}^{T} \left\| y_k(x_0, u_{0:k}^{\{i\}}; \Phi) - y_k^{\{i\}} \right\|_2^2 \\ &= \frac{1}{n_y} \left\| y_{\tau_w:T}(x_0, u_{0:T}^{\{i\}}; \Phi) - y_{\tau_w:T}^{\{i\}} \right\|_{2,2}^2,\end{aligned} \tag{4.7}$$

where the initial τ_w steps, known as *washout* period, are discarded to accommodate the model transient due to the random initialization [14].

The MSE over a generic batch $\mathcal{I}$ is therefore defined as

$$\mathrm{MSE}(\mathcal{I};\Phi) = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \mathrm{MSE}(y_{\tau_w:T}(x_0, u_{0:T}^{\{i\}};\Phi), y_{\tau_w:T}^{\{i\}}), \tag{4.8}$$

where $|\mathcal{I}|$ indicates the cardinality of the batch $\mathcal{I}$.

4.2.2 Stability-enforcing regularization term

The regularization term $\rho(\nu(\Phi))$ arises from a relaxation of the constraint $\nu(\Phi) < 0$ in (4.4), which is motivated by the fact that gradient descent-based algorithms do not support explicit constraints. In the spirit of a Lagrangian relaxation of such constraint, we thus introduce the term $\rho(\nu(\Phi))$ in the loss function, designed to penalize the violation of the stability constraint. Letting n_ν indicate the cardinality of $\nu(\Phi)$, we point out that $\rho : \mathbb{R}^{n_\nu} \to \mathbb{R}$ can be any real-valued function that is strictly increasing with its argument. Moreover, this function should be designed not to encourage unnecessarily low values of $\nu(\Phi)$.

Two function candidates are herein proposed for $\rho(\cdot)$. The former is more straightforward and simple, but can make the training procedure more oscillatory due to a gradient discontinuity. The latter allows instead to limit such oscillations, at the price of an additional hyperparameter to be tuned.

Piecewise-linear function

A straightforward candidate for the function $\rho(\cdot)$ is a piecewise-linear function. Let us indicate by $\varepsilon_\nu > 0$ the constraint clearance. The regularization function can hence be stated as

$$\rho(\nu(\Phi)) = \sum_{i=1}^{n_\nu} \bar{\pi}\max([\nu(\Phi)]_i + \varepsilon_\nu, 0) + \underline{\pi}\min([\nu(\Phi)]_i + \varepsilon_\nu, 0), \tag{4.9}$$

where $\bar{\pi} > 0$ and $\underline{\pi} > 0$ denote the slopes of the piecewise-linear function. Note that by taking $\underline{\pi} \ll \bar{\pi}$ one can prevent excessively negative values of $\nu(\Phi)$ from being rewarded.

Generalized piecewise-linear function

Although the piecewise-linear function introduced in (4.9) allows, if its coefficients are properly selected, to steer the RNN model towards its stability region, it is plagued by the fact that its gradient is, by construction, discontinuous in $[\nu(\Phi)]_i = -\varepsilon_\nu$. This gradient discontinuity, unfortunately, may lead to oscillations of the gradient descent algorithms used to perform

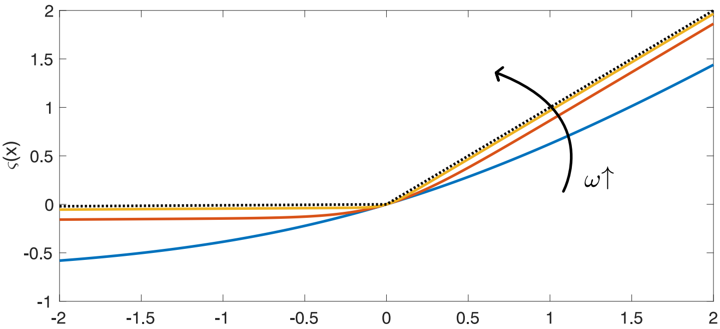

Figure 4.1: *Visual comparison of the generalized piecewise-linear function proposed in* (4.10a) *with $\underline{\pi} = 0.01$ and $\bar{\pi} = 1$, for several values of ω (blue line: $\omega = 1$, red line: $\omega = 5$, yellow line: $\omega = 10$). Notice that as ω increases $\varsigma(x)$ approaches the piecewise-linear function depicted as a black-dotted line.*

the training procedure. We therefore propose a generalized version of the piecewise-linear regularization term that features a continuous gradient.

Let us define the function $\varsigma(x)$ as the primitive of a sigmoidal function ranging from $\underline{\pi}$ to $\bar{\pi}$ with steepness $\omega > 0$, i.e.

$$\begin{aligned} \varsigma(x) &= \int \underline{\pi} + (\bar{\pi} - \underline{\pi})\sigma(\omega x) dx \\ &= \underline{\pi} x + \frac{\bar{\pi} - \underline{\pi}}{\omega} \Big[\ln\big(1 + e^{\omega x}\big) + c \Big] \\ &= \underline{\pi} x + \frac{\bar{\pi} - \underline{\pi}}{\omega} \Big[\ln\big(1 + e^{\omega x}\big) - \ln 2 \Big], \end{aligned} \tag{4.10a}$$

where c is computed to guarantee that $\varsigma(0) = 0$. The function $\varsigma(x)$ has been plotted in Figure 4.1 and compared to a piecewise-linear function. Notice that as the steepness coefficient ω increases, $\varsigma(x)$ approaches its piecewise-linear counterpart. Thus, while the smoothness of $\varsigma(x)$ comes at the price of an additional hyperparameter, it can be easily selected to be sufficiently large to ensure that $\varsigma(x)$ approximates the piecewise-linear function[8].

At last, the proposed smoothed regularization term reads as

$$\rho(\nu(\Phi)) = \sum_{i=1}^{n_\nu} \varsigma\big([\nu(\Phi)]_i + \varepsilon_\nu\big), \tag{4.10b}$$

where $\varepsilon_\nu \geq 0$ is the constraint clearance.

[8] If $\omega \to \infty$, $\sigma(\omega x) \to \text{step}(x)$, whose primitive is a piecewise-linear function with coefficients $\underline{\pi}$ for $x \leq 0$ and $\bar{\pi}$ for $x > 0$.

4.2.3 Challenges and common practices

In the description made above, some of the common practices and challenges of the training procedure have been neglected for the sake of simplicity. We hence discuss these common practices and provide guidelines on how to choose some of the many hyperparameters of the proposed training procedure.

First, it is worth emphasizing how critical it is to normalize the input-output data used for training, validation, and testing [14]. This holds true for three reason: (*i*) different scales of the outputs would mean an unbalanced training procedure, as the gradient of $\mathcal{L}$ would be mostly determined by the outputs with the largest scale; (*ii*) large inputs would easily lead to the saturation of the activation functions, leading to very small gradients and hence to a slow and ineffective gradient descent; (*iii*) most of the introduced stability conditions rely upon the common unity-bounded input assumption (Assumption 3.1). The datasets $\mathcal{D}_{\text{tr}}$, $\mathcal{D}_{\text{val}}$, and $\mathcal{D}_{\text{te}}$ are therefore assumed to be normalized.

Remark 4.1. *If $\mathcal{D}_{\text{tr}}$, $\mathcal{D}_{\text{val}}$, and $\mathcal{D}_{\text{te}}$ are not normalized one can easily normalize the data as follows. The input sequences can be normalized component wise with respect to the generally known saturation values $[\bar{u}]_i$ and $[\underline{u}]_i$. Letting $[m_u]_i = \frac{[\bar{u}]_i + [\underline{u}]_i}{2}$ be the bias of the i-th input component and $[s_u]_i = [\bar{u}]_i - [m_u]_i$ its scale one can compute the normalized input by applying, for any $i \in \{1, ..., n_u\}$, the transformation*

$$[u_{\alpha,k}]_i \leftarrow \frac{[u_{\alpha,k}]_i - [m_u]_i}{[s_u]_i}, \tag{4.11}$$

$\forall k \in \{0, ..., T_\alpha\}$, *where* $\alpha \in \{\text{tr}, \text{val}, \text{te}\}$. *The output sequences are instead normalized component-wise with respect to the empirical mean and scale computed over the training set. That is, for any output component $i \in \{1, ..., n_y\}$, we compute*

$$[m_y]_i = \frac{1}{T_{\text{tr}} + 1} \sum_{k=0}^{T_{\text{tr}}} [y_{\text{tr},k}]_i,$$
$$[s_y]_i = \max_k |[y_{\text{tr},k}]_i - [m_y]_i|,$$

and the output sequences are then normalized applying the following transformation

$$[y_{\alpha,k}]_i \leftarrow \frac{[y_{\alpha,k}]_i - [m_y]_i}{[s_y]_i}, \tag{4.12}$$

$\forall k \in \{0, ..., T_\alpha\}$, *where* $\alpha \in \{\text{tr}, \text{val}, \text{te}\}$.

Secondly, we would like to steer the attention of the reader to the challenge of selecting the hyperparameters of the training procedure. These hyperparameters include

- The length of the subsequences T_s: large T_s generally allow for long-term modeling performances, but excessively large values lead to cumbersome gradient computation and thus a significantly slower training.
- The number of training and validation subsequences, i.e. N_{tr} and N_{val}: the number of validation subsequences should be enough to reasonably cover the whole validation dataset $\mathcal{D}_{\text{val}}$, hence $N_{\text{val}} T_s > T_{\text{val}}$, while the number of training subsequences is generally taken so that $N_{\text{tr}} T_s \gg T_{\text{tr}}$.
- The number of batches B, which is inversely proportional to the so-called *batch size* $|\mathcal{I}_{\text{tr}}^{\{b\}}|$: as discussed in [14], large batch sizes allow to take advantage of parallel computing and to improve the smoothness and stability of the gradient descent, but increase the likeliness of getting stuck in local minima. On the other hand, small batches result in less exploitation of parallelization and hence in longer training times, and may even undermine the convergence of the gradient descent algorithm.
- The washout period τ_w, which should as low as possible while accommodating the initialization transient [13].
- The GD-based optimization algorithm (Adam, RMSProp, etc.) and its hyperparameters, which need to be carefully tuned to ensure the convergence of the training procedure while escaping the local minima [14].
- The stability-related regularization term $\rho(\cdot)$, which should be designed following the guidelines provided in Section 4.2.2, and its coefficients, i.e. $\underline{\pi}$ and $\bar{\pi}$. Large values of these coefficients could make this regularization term of the loss function predominant, introducing a “stability bias” that could harm the effectiveness of the training procedure, yielding unsatisfactory modeling performances. On the other hand, values that are too small may be unable to enforce the satisfaction of the stability conditions.

Finding appropriate values of these hyperparameters is usually a result of the designer’s knowledge, and often several trial-and-error attempts are required to obtain satisfactory results. In particular, with respect to the last point, it is worth noting that the introduction of the stability-related regularization term, if not designed appropriately, may lead to stable models with worse performance than those that would be attained without such stability guarantee.

Finally, as mentioned at the beginning of Chapter 4.2, we point out that training provably stable RNNs is generally desirable only if the system that generates the input-output training data enjoys the same stability proprieties, as discussed in the following remark.

Remark 4.2. *The input-output datasets $\mathcal{D}_{\text{tr}}$, $\mathcal{D}_{\text{val}}$, and $\mathcal{D}_{\text{te}}$ are assumed to be generated by a system which enjoys the same stability property that is being enforced by the regularization term $\rho(\nu(\Phi))$. This assumption may be known from physical arguments, or they could be assessed numerically on the measured sequences, see e.g. [58]. Obviously, in this latter case, since only the input-output trajectories are measured, one needs to resort to ISS, ISPS, and δISS-like properties referred to output variables, such as the input-to-output stability [73]. On the other hand, enforcing stability conditions when learning an unsuitable system may lead to poor performances of the RNN model, as the regularization term biases the training procedure towards the space of provenly-stable networks.*

4.3 Numerical example

4.3.1 Benchmark system description

The benchmark system here adopted to test the proposed training procedure is the pH neutralization process described in [110, 111], and depicted in Figure 4.2. The system is composed of two tanks, i.e. Tank 1 and Tank 2.

Tank 2 is fed by the acid flow rate q_1, and its output is the flow rate q_{1e}. The hydraulic dynamics of Tank 2, being much faster than the others involved, is neglected, i.e. it is assumed that $q_1 = q_{1e}$.

Tank 1, referred to as *reactor tank*, is fed by three flows, namely the acid flow rate q_{1e}, the buffer flow rate q_2, and the alkaline base flow rate q_3. The terms q_1 and q_2 can not be manipulated and represent disturbances. The alkaline flow rate q_3 can be instead modulated by means of a controllable valve, and thus represent a control variable. The output of the reactor tank is the fixed flow rate q_4, from which the pH is measured. The objective of the control scheme is to control the pH of the output flow to a (piecewise) constant setpoints.

A simplified model [111] of this system is described by a third order continuous-time dynamical system having one input (i.e. $u = q_3$), one

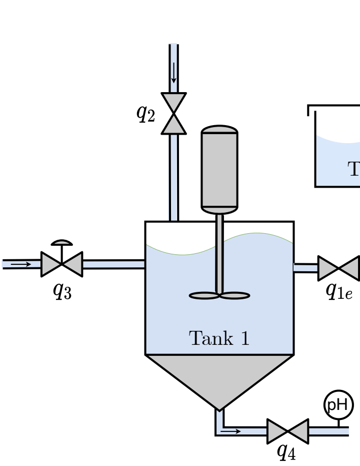

Figure 4.2: *Schematic of the pH neutralization process benchmark system.*

output (i.e. $y = pH$), and two exogenous disturbances (i.e. $d = [q_1, q_2]'$). The model is described by the following constrained differential equations:

$$\begin{aligned} \dot{x}(t) &= f_p(x(t), u(t), d(t)) \\ c(x(t), y(t)) &= 0, \end{aligned} \tag{4.13a}$$

where function $f_p(x(t), u(t), d(t))$ describes by the following dynamics

$$\dot{x}_1 = \frac{q_1}{A_1 x_3}(W_{a1} - x_1) + \frac{q_2}{A_1 x_3}(W_{a2} - x_1) + \frac{q_3}{A_1 x_3}(W_{a3} - x_1), \tag{4.13b}$$

$$\dot{x}_2 = \frac{q_1}{A_1 x_3}(W_{b1} - x_2) + \frac{q_2}{A_1 x_3}(W_{b2} - x_2) + \frac{q_3}{A_1 x_3}(W_{b3} - x_2), \tag{4.13c}$$

$$\dot{x}_3 = \frac{1}{A_1}(q_1 - C_{v4}(x_3 + z)^n) + \frac{q_2}{A_1} + \frac{q_3}{A_1}, \tag{4.13d}$$

in which the time index has been omitted for the sake of compactness. The output y is instead determined by the implicit equation $c(x(t), y(t)) = 0$, defined as

$$c(x, y) = x_1 + 10^{y-14} + 10^{-y} + x_2 \frac{1 + 2 \cdot 10^{y-pK_2}}{1 + 10^{pK_1 - y} + 10^{y-pK_2}}, \tag{4.13e}$$

where the parameters pK_1 and pK_2 are the first and second dissociation constants of the weak acid H_2CO_3.

$A_1 = 207$	cm^2	$W_{a1} = 3$	$^{mol}/_{mL}$	$q_1 = 16.6$	$^{mL}/_{s}$
$z = 11.5$	cm	$W_{a2} = -3$	$^{mol}/_{mL}$	$q_2 = 0.55$	$^{mL}/_{s}$
$C_{v4} = 4.59$		$W_{a3} = -3.05$	$^{mol}/_{mL}$	$q_3 = 15.6$	$^{mL}/_{s}$
$n = 0.607$		$W_{b1} = 0$	$^{mol}/_{mL}$	$q_4 = 32.8$	$^{mL}/_{s}$
$pK_1 = 6.35$		$W_{b2} = 30$	$^{mol}/_{mL}$		
$pK_2 = 10.25$		$W_{b3} = 0.05$	$^{mol}/_{mL}$		

Table 4.1: *Nominal parameters and operating conditions of the pH neutralization process*

The nominal parameters of the model are reported in Table 4.1, and the nominal operating point is

$$\begin{aligned} \bar{x} &= [-0.432, 0.528, 14]' & \bar{u} &= 15.6, \\ \bar{d} &= [16.6, 0.55]' & \bar{y} &= 7.0. \end{aligned}$$

Moreover, the alkaline flow q_3 is subject to a following saturation constraint, i.e.

$$u(t) \in [12.5, 17]. \tag{4.14}$$

A simulator of the plant equations (4.13) has been implemented in Simulink in order to collect the necessary data for the RNN training procedure. Notably, it has been numerically verified that the system displays δISS-like properties through the inspection of input-output trajectories. The proposed benchmark system is therefore a good candidate to be learned by a δISS RNN model.

4.3.2 Training procedure

To collect an informative dataset spanning the operating regions and frequencies of interest, the plant simulator has been fed with Multilevel Pseudo-Random Signals (MPRS). These signals consist of trains of steps of random amplitude and duration, to which a white Gaussian noise with standard deviation 0.0125 has been superimposed. Note that, before running the simulation for data collection, it should be verified that the randomly generated input sequences for the validation and test datasets span the entire input set spanned by the training dataset, so that the network performances are validated and tested on suitably informative datasets.

The input and output data of the system have been recorded with a sampling time $\tau_s = 15$s: in this way, enough datapoints are collected during transients. Output measurements have also been corrupted by a white Gaussian noise with standard deviation 0.01. Overall, the training dataset $\mathcal{D}_{\text{tr}}$ consists of a single experiment of duration $T_{\text{tr}} = 5000$ steps, while both

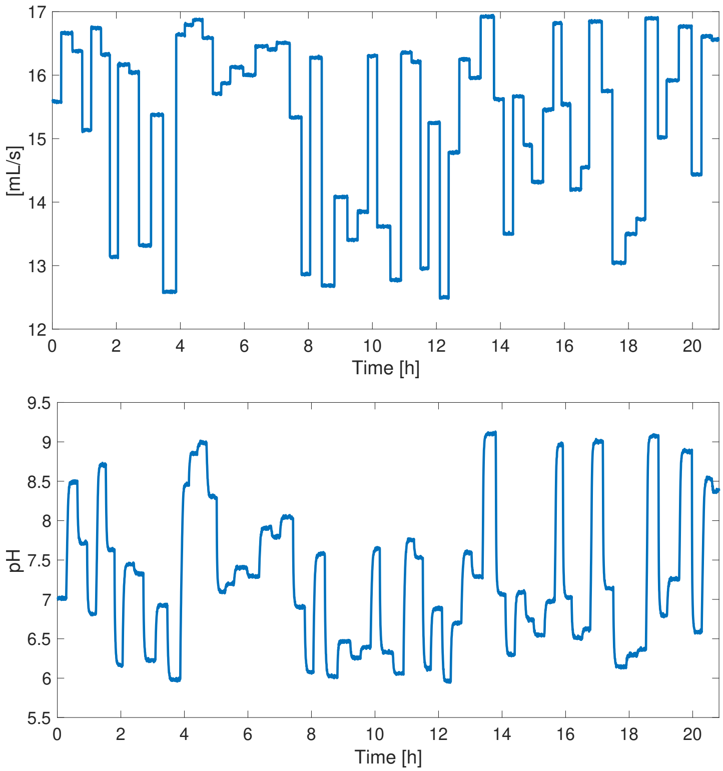

Figure 4.3: *Training set: applied input sequence (top) and measured pH (bottom).*

the validation and test input-output sequences, i.e. $\mathcal{D}_{\text{val}}$ and $\mathcal{D}_{\text{te}}$, consist of one experiment of duration $T_{\text{val}} = T_{\text{te}} = 1500$ steps each. In Figure 4.3 the input and output sequences of the training dataset are depicted.

It is worth stressing that before these input-output sequences can be used to carry out the training procedure of the RNN model, they must be normalized according to the instructions provided in Remark 4.1. For the sake of interpretability, however, denormalized trajectories (i.e., in their original scale) are represented in the figures reported below. After normalization, according to the TBPTT paradigm, partially overlapping subsequences are randomly extracted from $\mathcal{D}_{\text{tr}}$ and $\mathcal{D}_{\text{val}}$. The training and validation datasets are defined by extracting $N_{\text{tr}} = 300$ training subsequences and $N_{\text{val}} = 50$ validation subsequences, respectively, of the same length $T_s = 250$.

The training procedure described in Algorithm 1 has been implemented in PyTorch 1.10 [112], using RMSProp as optimization algorithm and an early stopping criterion, i.e., stopping training after 250 epochs without performance improvement on the validation dataset. The training procedure

RNN arch.	Layer units	δISS regulariz.	FIT [%]	Average $\bar{\epsilon}$ $[10^{-3}]$	Average $\bar{\epsilon}_{ss}$ $[10^{-3}]$	Average epochs
GRU	$[7]$	No	96.83	14.16	12.33	925
		Yes	96.28	16.92	14.10	2662
	$[5,5]'$	No	97.45	11.30	9.80	1098
		Yes	96.97	12.03	10.56	2043
	$[7,7]'$	No	97.09	11.20	9.95	1355
		Yes	97.82	8.89	7.96	1985
LSTM	$[7]$	No	97.36	10.97	9.89	1137
		Yes	96.78	14.74	12.17	2707
	$[5,5]'$	No	97.76	9.35	8.25	1293
		Yes	96.70	13.52	12.18	2532
	$[7,7]'$	No	97.37	11.31	10.45	1080
		Yes	97.05	13.51	12.42	1396

Table 4.2: *Results of the training procedure*

has been tested for a variety of RNN models. Specifically, in Table 4.2 the configurations of the considered architectures and their performance (averaged over different training instances, characterized by different batch sizes) are reported. The table is explained below.

- Both shallow and deep architectures have been tested for GRUs and LSTMs. The column “Layer units” thus denotes the number of units for each layer, i.e. $\left[n_c^{(1)}, ..., n_c^{(L)}\right]'$. Scalar entries denote the number of units of shallow architectures.

- The above-mentioned architectures have been trained both with and without including the corresponding δISS regularization term, so as to assess the impact of the stability conditions on the training procedure. The inclusion of the δISS condition is reported in the “δISS regulariz.” column. Where included, the adopted regularization function is the Generalized Piecewise Function (4.10), with the fixed parameters $\overline{\pi} = 2 \cdot 10^{-3}$, $\underline{\pi} = 10^{-6}$, $\omega = 10$, and $\varepsilon_\nu = 0.04$.

- For each configuration, the average FIT index achieved and the average training epochs required are reported, alongside other performance indexes computed on the independent test set $\mathcal{D}_{\text{te}}$. The first index, $\bar{\epsilon}$, is the mean simulation error.

$$\bar{\epsilon} = \frac{1}{T_{\text{te}} - \tau_w + 1} \sum_{k=\tau_w}^{T_{\text{te}}} \| y_k(x_0, u_{\text{te},0:k}; \Phi^\star) - y_{\text{te},k} \|_2 \tag{4.15a}$$

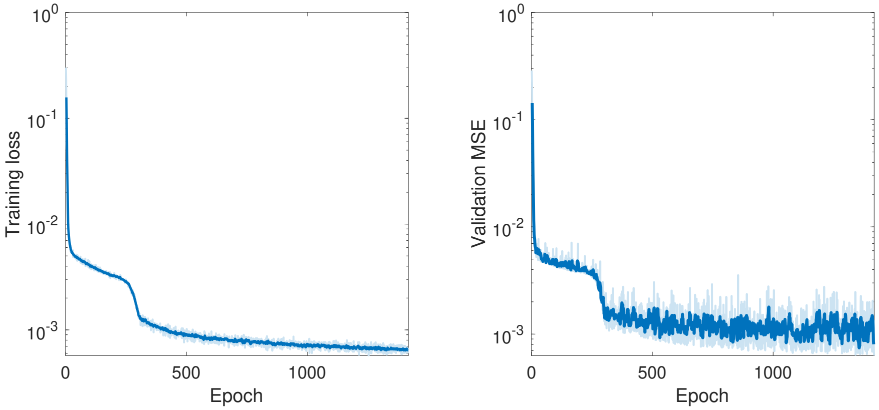

Figure 4.4: *Training procedure of a shallow GRU with* $n_c = 7$*. On the left, the evolution of the training loss function is depicted in logarithmic scale. On the right, the evolution of the* MSE *metrics on the validation dataset is reported. In both plots, the actual curves (light-blue lines) and their moving average (bold blue lines) are illustrated.*

The second index, $\bar{\epsilon}_{ss}$, denotes the mean post-transient simulation error. Denoting by $\mathcal{K}_{ss}$ the set of time-steps at which the RNN model is assumed to be settled[9], i.e.

$$\bar{\epsilon}_{ss} = \frac{1}{|\mathcal{K}_{ss}|} \sum_{k \in \mathcal{K}_{ss}} \| y_k(x_0, u_{\text{te},0:k}; \Phi^\star) - y_{\text{te},k} \|_2. \tag{4.15b}$$

Note that the NNARX architecture has not been considered here, as it will be discussed in Section 7.

The results reported in Table 4.2 demonstrate that, save for minor differences, the plant can be learned fairly well by the various architectures considered. Indeed, in all cases, the FIT index is around 97%, and the average error indexes are in the range of 1% of the operating range (which, in terms of denormalized output, correspond to an error on the pH prediction in the range of 10^{-2}). In light of the results it is possible to provide insights that may be useful when designing and training RNN models.

i. Shallow RNNs often have enough representational power to identify dynamical systems. Although one might be tempted to directly resort to deep RNNs, they generally call for a larger number of training epochs, an increased computational burden (i.e., a longer time-per-

[9] $\mathcal{K}_{ss}$ is determined empirically. In the proposed example, $u_{\text{te},0:T_{\text{te}}}$ is a train of steps with small input noise superimposed. Having verified that these steps have sufficient duration for the RNN model to converge in the absence of input noise, the set $\mathcal{K}_{ss}$ is hence considered as the last 10 samples of each step.

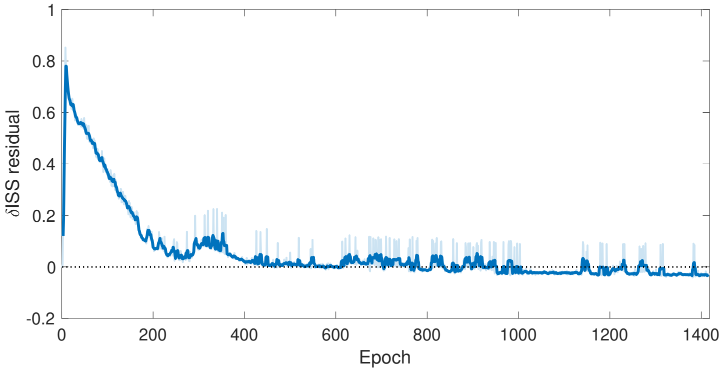

Figure 4.5: *Evolution of the δISS residual $\nu(\Phi)$ throughout the training. The light-blue line indicates the actual residual, while the bold-blue line is its moving average.*

epoch), a greater dependence of the attained performances on the initialization of Φ, see [14], and a greater risk of overfitting.

ii. Despite the conservativeness of the proposed δISS conditions, consistently with the results discussed in [58] the results suggest that the enforcement of the RNNs' δISS property via regularization does not significantly harm the RNNs' modeling performances, but rather it affects the number of training epochs required. This behavior is explained by the fact that, if the training hyperparameters are properly designed, the order of the regularization term $\rho(\nu(\Phi))$ is comparable to that of the MSE. Thus, at the beginning of the training procedure, the optimizer trades modeling performances with the reduction of the δISS residual.

iii. Albeit the δISS residuals (i.e., $\nu(\Phi^\star)$) have been omitted for simplicity, it has been verified that the RNNs trained without enforcing the δISS conditions are characterized by positive δISS residuals[10]. This means that these networks can not be guaranteed to be δISS. This is not surprising, as the proposed conditions are conservative and only sufficient.

In order to provide a more detailed example of the training procedure, let us consider a shallow GRU with $n_c = 7$ units. Such model is trained including the δISS-related regularization term in the loss function, in order to ensure its δISS. The batch size here adopted is 20.

Figure 4.4 shows the evolution of the loss function and of the validation MSE performance index is reported. Note that the convergence of the training procedure is rather smooth. According to the early stopping rule,

[10] The order of magnitude of these residuals depends on the specific architecture. For GRUs, it generally ranges from 10^5 to 10^8, whereas for LSTMs it is tipically smaller, in the order of 10^1.

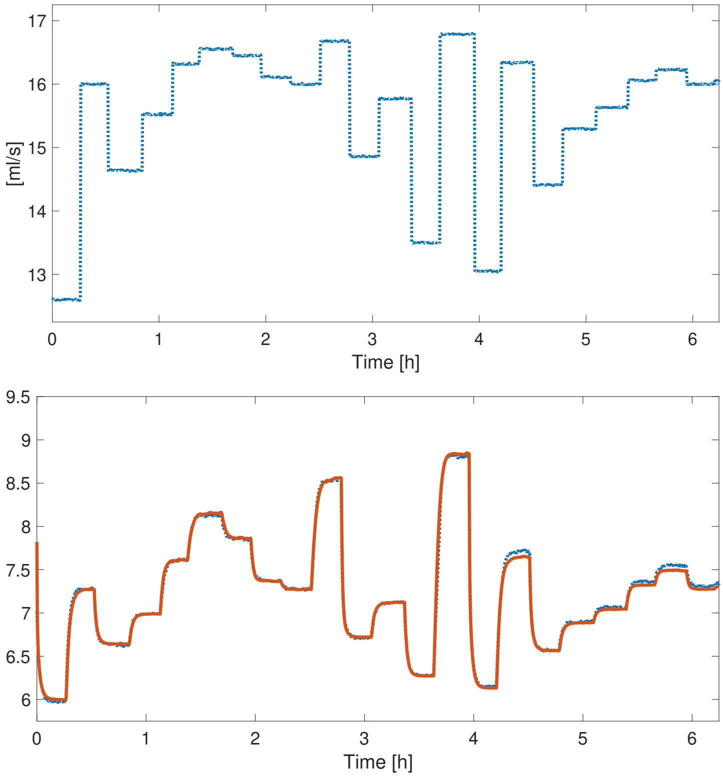

Figure 4.6: *Performances of a trained shallow GRU model. In the top plot the test dataset's input sequence is depicted, while on the bottom the open-loop simulation of the model (red solid line) is compared to the actual output (blue dotted line).*

the training procedure is halted after 1414 training epochs, since the validation MSE stops improving. Each epoch takes approximately $0.6s$ to be executed on an i7 2.6 GHz processor.

In Figure 4.5 the evolution of the δISS residual $\nu(\Phi)$ is depicted. Notice that after epoch 1000 the residual oscillates around $-\varepsilon_\nu$, for which values the network δISS is ensured. Note that these oscillations can be dampened by decreasing the value of the coefficients $\underline{\pi}$ and $\bar{\pi}$, at the price of a possibly slower convergence of the training procedure.

Finally, the modeling performances of the trained network are assessed on the independent test set $\mathcal{D}_{\mathrm{te}}$. In Figure 4.6 the test set input sequence $u_{\mathrm{te},0:T_{\mathrm{te}}}$ is depicted, and open-loop simulation of the trained model, i.e. $y_{0:k}(x_0, u_{\mathrm{te},0:T_{\mathrm{te}}}; \Phi^\star)$, is compared to the measured output sequence $y_{\mathrm{te},0:T_{\mathrm{te}}}$. The FIT index scored by the GRU model is 96.44%, while $\bar{\epsilon} = 16.33 \cdot 10^{-3}$ and $\bar{\epsilon}_{ss} = 13.95 \cdot 10^{-3}$, indicating satisfactory modeling performances.

4.4 Summary

In this chapter a training procedure of the previously discussed RNN models has been proposed. This procedure is based on the method of truncated back-propagation through time and allows RNN models to be trained using input-output trajectories collected from the plant via experiments. A suitably designed regularization term is also included so that the network meets the sufficient conditions that guarantee its ISPS, ISS or δISS. Finally, the effectiveness of the proposed training procedure has been assessed on a pH neutralization process benchmark system, and guidelines on the selection of the hyperparameters have been discussed.

Part II

Control design

CHAPTER 5

Introduction to the control problem

In the previous part of this thesis, the use of stable RNN models for the identification of nonlinear dynamical systems has been investigated. Special attention has been devoted to training provenly ISPS, ISS, and δISS RNN models, with the pledge of exploiting these stability properties for the synthesis of control systems with closed-loop stability guarantees. In the spirit of the indirect data-driven control synthesis approach, model-based control laws are generally synthesized based on the stable identified model by relying on the so-called Certainty Equivalence Principle (CEP). This principle consists of assuming that the unknown plant and its RNN model coincide.

Under this premise we henceforth assume that a stable RNN model $\Sigma(\Phi^\star)$ of the plant has been trained by means of the procedure shown in Chapter 4 using suitably collected input-output data. By invoking the CEP, the plant is assumed to be described by the same state-space system equations as $\Sigma(\Phi^\star)$. The control synthesis problem thus boils down to designing a control law based on the identified RNN model $\Sigma(\Phi^\star)$, for which many techniques are potentially available [16].

5.1 Problem Statement

Let us consider a generic RNN model of the plant characterized by the state-space form

$$\Sigma(\Phi^\star) : \begin{cases} x_{k+1} = f(x_k, u_k; \Phi^\star) \\ y_k = g(x_k; \Phi^\star) \end{cases}, \tag{5.1}$$

where the functions f and g depend on the chosen RNN architecture, see Chapter 3. Model (5.1) is here assumed to be exponentially δISS with respect to its invariant set $\mathcal{X}$ and the input set $\mathcal{U}$. Such stability property can be guaranteed, e.g., by ensuring that the corresponding δISS sufficient conditions $\nu(\Phi^\star) < 0$ hold true.

The main goal of the controller is to steer the output of the plant to a reference value $\bar{y}$, while fulfilling the input constraint $u_k \in \tilde{\mathcal{U}}$, where $\tilde{\mathcal{U}} \subseteq \mathcal{U}$ represents a potentially tighter input constraint than the input set $\mathcal{U}$ on which the δISS property has been derived. To address the regulation problem, the existence of a feasible equilibrium corresponding to the output reference is assumed.

Assumption 5.1. *Given the output reference $\bar{y}$, there exists $\bar{x} \in \mathrm{Int}(\mathcal{X})$ and $\bar{u} \in \mathrm{Int}(\tilde{\mathcal{U}})$ such that the triplet $\bar{\Sigma} = (\bar{x}, \bar{u}, \bar{y})$ is a feasible equilibrium, i.e.*

$$\begin{cases} \bar{x} = f(\bar{x}, \bar{u}; \Phi^\star) \\ \bar{y} = g(\bar{x}; \Phi^\star) \end{cases} \tag{5.2}$$

In the following, the linearization of $\Sigma(\Phi^\star)$ around the equilibrium $\bar{\Sigma}$, denoted by $\delta\Sigma(\bar{\Sigma})$, is characterized by the matrices $A_\delta(\bar{\Sigma})$, $B_\delta(\bar{\Sigma})$, and $C_\delta(\bar{\Sigma})$, computed as described in (2.15). The following customary assumptions are therefore taken on $\delta\Sigma(\bar{\Sigma})$.

Assumption 5.2. *The linearized system $\delta\Sigma(\bar{\Sigma})$, described by the tuple*

$$\big(A_\delta(\bar{\Sigma}), B_\delta(\bar{\Sigma}), C_\delta(\bar{\Sigma})\big),$$

is reachable, observable, and does not have any invariant zero in $z = 1$.

Remark 5.1. *Owing to Theorem 1 of [113], Assumption 5.2 entails the existence of an open neighborhood of $\bar{y}$, denoted by $\Gamma(\bar{y}) \subseteq R^{n_y}$ where, for any $\tilde{y} \in \Gamma(\bar{y})$, there exists a feasible equilibrium triplet $(\tilde{x}(\tilde{y}), \tilde{u}(\tilde{y}), \tilde{y})$ such that $\tilde{x}(\tilde{y}) = f(\tilde{x}(\tilde{y}), \tilde{u}(\tilde{y}); \Phi^\star)$ and $\tilde{y} = g(\tilde{x}(\tilde{y}); \Phi^\star)$, with $\tilde{x}(\tilde{y}) \in \mathrm{Int}(\mathcal{X})$ and $\tilde{u}(\tilde{y}) \in \mathrm{Int}(\tilde{\mathcal{U}})$. This local result allows to conclude that it is possible to move the output setpoint in a neighborhood of $\bar{y}$ and still guarantee that a feasible solution to the control problem exists.*

At this stage, assume that we have a setpoint $\bar{y}$ and that Assumptions 5.1 and 5.2 are satisfied.

Problem 5.1. *(Regulation problem) Given the model $\Sigma(\Phi^\star)$ and the output setpoint $\bar{y}$, steer the system to the equilibrium $\bar{\Sigma}$ by means of a control action that satisfies the input constraint $u_k \in \tilde{\mathcal{U}}$.*

The main control strategy considered in this thesis to address Problem 5.1 is Nonlinear Model Predictive Control (NMPC), discussed below. An alternative strategy, based on Internal Model Control, is also considered.

5.2 Model Predictive Control

In the framework of nonlinear systems' regulation, nonlinear model predictive control represents a solid and mature control strategy [114–116], as it allows to straightforwardly handle the nonlinearity of the model and the constraints on input, state, and output variables. Many variants of the traditional NMPC scheme have been proposed, to mention a few, to ensure outputs' offset-free tracking, see [117, 118], robustness against uncertainty, see [119–121], chance-constrained constraint satisfaction, see [122]. The broad applicability of NMPC, the possibility to naturally enforce constraints, and its often-superior closed-loop performances have thus led to its tremendous diffusion in both academia and industry, see [123].

Below, the working principle of NMPC is concisely summarized. Note that the provided description is not intended to cover the wide variety of NMPC architectures available in the literature, but is sufficiently abstract to fit the different NMPC strategies discussed in this thesis.

That of NMPC is a model-based implicit state-feedback control law. The control action applied to the plant is determined, at each instant k, by solving a nonlinear Finite Horizon Optimal Control Problem (FHOCP). In the FHOCP, the availability of the model (i.e., (5.1)) is exploited to relate the future evolution of the state and output trajectories, along the so-called prediction horizon, to any given control sequence. Solving the FHOCP amounts to computing the control sequence which leads to the lowest possible value of some cost function, which expresses and quantifies the control objectives, subject to constraints on the future state and input trajectories. The first optimal control action is applied and then, in accordance to the receding horizon principle, the entire procedure is repeated at step $k + 1$.

Note that, in order to use (5.1) to predict future state and output trajectories, at each time step k, it is necessary to know the actual state of such system. Therefore, since black-box systems are characterized by unmeasurable states, they call for suitably designed state observers.

5.2.1 State of the art of NN-based MPC

Given its peculiarities, in the realm of indirect data-driven approaches, that of NMPC based on identified neural models has been a popular choice since the early days of NN. In [124–126], the authors proposed NMPC laws to control systems learned by FFNN-based one-step ahead predictor models. Implementations of this approach on real systems have demonstrated satisfactory performances [23].

Because MPC performance is related to the accuracy of the underlying model, recent works have considered the use of RNNs as identified plant models for the synthesis of predictive control laws. This proposal is motivated by the modeling power of RNNs, which are known to outperform FFNN for learning dynamic systems, see e.g. [13]. For example, in [54] the authors proposed to learn the model of a complex chemical system using continuous-time RNNs, which has been then used to design an NMPC law; in [24] and [25] GRU and LSTM models have been used to design NMPCs for continuous process industries; in [55] an NMPC law based on an LSTM model of a business center's cooling system has been proposed, showing that the modeling performances of LSTMs allow such control scheme to outperform a predictive control law based on a grey-box model; in [127] predictive control laws based on continuous-time RNNs have been proposed for chemical systems.

While the above contributions are relevant, as they illustrate the potential of RNN-based NMPC laws, there is a need for a theoretical framework that sheds light on the closed-loop stability guarantees of this kind of scheme. A first step towards this aim is [128], in which the author has shown that, for specific two-layer FFNN structures, by suitably designing an NMPC scheme one can provide closed-loop stability guarantees. Similarly, in [19] the authors have shown that single-layer continuous-time vanilla RNNs can be employed to design NMPC laws with nominal closed-loop stability guarantees, under assumptions on the RNN model's Lipschitzianity and on the boundedness of the modeling error.

A different approach to ensure nominal closed-loop stability has been proposed in [93], which relies on the δISS property of the considered ESN model, and on the design of a provenly convergent state observer for such model. Note that a state observer is required to operate the RNN predictive model in closed-loop since, save for the NNARX architecture, RNNs are characterized by unmeasurables states, that therefore need to be estimated. To attain theoretical closed-loop guarantees, such observer must yield a state estimation converging to the true, unknown, state of the plant.

In general, it is well known that the static performances of an NMPC law are closely related to the accuracy of the predictive model. This also applies to the control scheme mentioned above, which unfortunately does not guarantee zero-error output regulation to constant references [22]. That is, a static plant-model mismatch, however small, can lead to a mismatch between the closed-loop plant's output and its constant reference value.

The problem of designing *offset-free* NMPC laws, i.e. that guarantee both closed-loop asymptotic stability and zero asymptotic output regulation error to constant references, is well known in the MPC literature, see [129] for an in-depth survey on the topic. There are two main strategies by which the problem has been addressed.

A first strategy has been proposed by Morari and Maeder [118], where the authors propose to enlarge the model of the system with the dynamics of a fictitious disturbance, generally assumed to be constant and additive to the inputs or outputs of the system. Such enlarged model is then used both to design a state observer providing an estimate of the disturbance and as a predictive model for an NMPC control law which, embedding the dynamics of the disturbance, is able to reject it. As pointed out in [130], crucial to the success of this approach is the possibility to suitable model this fictitious disturbance and to design an appropriate algorithm for its estimation, which is not always straightforward.

A second strategy is the one devised by Magni et al. [117], in which the authors propose to enlarge the system model with integrators on the output tracking error, and then to use such augmented model both to design a weak detector (i.e., a state observer with nominal convergence guarantees) and to synthesize a suitable NMPC law. This scheme results in a nominally closed-loop stable offset-free control architecture. Moreover, the asymptotic zero-error output regulation is robust to model uncertainty, as long as the closed-loop stability is preserved [117].

While many offset-free strategies have been proposed based on the two aforementioned approaches, to the best of the author's knowledge few works deal specifically with their synthesis for systems identified by RNN models. In particular, a disturbance estimation-based NMPC has been proposed for NNARXs in [131], while in [132] RNN models are used to learn the mismatch (to be compensated) between the plant and its linear model.

5.3 Internal Model Control

Albeit NMPC-based strategies are capable of achieving remarkable performance, guaranteeing constraint satisfaction and, if properly designed (e.g., following the procedures proposed in the next chapters), even nominal closed-loop stability, they call for the solution of an online nonlinear optimization problem at every time-step. This computational burden may be, in some cases, prohibitive. Therefore, in these cases, the control problem can be addressed resorting to a completely different control architecture, i.e., the Internal Model Control (IMC) strategy [133].

In short, this strategy consists of synthesizing a controller that approximates the inverse of the system model. This means that, ideally, the controller, fed with the output reference, should then generate the control sequence that steers the model's output as close as possible to such reference. In this scheme, to avoid to have a purely open-loop scheme, a negative feedback of the plant-model mismatch is also included [134], which allows to improve the static performances and the robustness of the scheme.

The remarkable peculiarity of this scheme is that the controller is synthesized offline based on the system model only, and requires limited online computations. This architecture is therefore extremely lightweight from the computational standpoint, which makes it a good complementary alternative to NMPC.

5.3.1 State of the art of NN-based IMC

Despite the potential of this control strategy, it has mostly been used with linear systems [135] since, for that class of systems it is easy to guarantee the existence of the model inverse and to compute it. Extensions to nonlinear systems have also been provided, see [133, 134]. However, attention is devoted on ensuring the existence of the exact inverse of the model, and hence the classes of nonlinear systems considered are input-affine ones, which can nonetheless lead to reasonable closed-loop performance in some practical cases [136].

In this context, [15] proposed to adopt static FFNN as system's model and model's inverse approximators, relying on their universal approximation guarantees. A similar approach is undertaken by [137], in which FFNN networks are used in an autoregressive configuration to approximate the model and synthesize the controller.

In these approaches, however, the model's and controller's dependence on the past trajectories is encoded by embedding a sufficiently long regression horizon of past input-output regressors. On the other hand, as exten-

Architecture	NMPC	Offset-free NMPC	IMC
NNARX	-	Chapter 7, [61]	-
LSTM	[22, 63]	[138]	Chapter 8
GRU	Chapter 6.1, [63]	Chapter 6.2, [60]	Chapter 8, [62]

Table 5.1: *Summary of the control schemes proposed by the Author.*

sively discussed, RNNs would generally be better candidates for learning and approximating dynamic systems. Despite their potential, however, to the best of the author's knowledge, IMC schemes based on gated RNNs have not been proposed so far.

5.4 Contributions

Several control schemes, based on the RNN models discussed in Section 3, have been proposed, see Table 5.1.

In [22] it has been shown that, when the system is learned by shallow LSTMs, the δISS of the model allows to design nominally converging state observers which, in turn, allow to synthesize a nominally closed-loop stable NMPC law. Along this line, in Chapter 6.1 a theoretically-sound NMPC architecture has been devised for δISS shallow GRUs. Such scheme is able to guarantee nominal closed-loop stability under sufficient conditions on the NMPC's prediction horizon and weights. The synthesis of this control law consists of two distinct design steps, i.e. (*i*) proposing a state observer with nominal convergence guarantees and (*ii*) formulating an NMPC law which exploits such convergent state observation to guarantee nominal stability of the closed-loop. The proposed strategy can also be extended to other nonlinear δISS systems, provided exponentially converging state observers can be synthesized – such as in the case of shallow LSTMs –, and will be object of a future publication [63].

The problem of designing an offset-free NMPC scheme with closed-loop stability guarantees has been addressed in [138] and [60] for δISS shallow LSTMs and GRUs, respectively. These control schemes, based on the control strategy originally introduced in [117], rely upon the model's δISS to design the ingredients of the control law, i.e. the gain of the integral action, as well as a weak detector and a stabilizing control law for the augmented system model. Albeit being slightly more complex to be synthesized, we show that this second control scheme is more robust to disturbances affecting the plant. In Chapter 6.2, the synthesis of this control scheme for δISS shallow GRUs is described.

A similar control strategy, proposed for δISS NNARX models in [57], is described in Section 7. In this scheme, the control variable is designed as the sum of two components, i.e., an integral action related to the tracking error, which allows to attain asymptotic zero-error output tracking of constant references, and a derivative action, which instead allows to improve the closed-loop dynamic performances.

Finally, in [62] an IMC scheme based on δISS deep GRUs has been proposed. Since this approach readily extends to LSTMs, in Chapter 8 it is described considering any general δISS gated RNN. In particular, a strategy is proposed to train the controller, described by a RNN, to approximate the inverse of the model, also described by a RNN. Relying on the δISS of both networks, the closed-loop properties are discussed.

5.5 Summary

In this chapter the control problem to be addressed in the reminder of this part has been formulated. Two main control approaches that can be profitably employed for this purpose have been briefly presented, namely the nonlinear model predictive control and the internal model control. The available literature on the use of these control architectures on the considered RNN models has been discussed, and the necessity of leveraging the devised theoretical stability framework to design control laws with closed-loop stability guarantees has emerged.

CHAPTER 6

Model Predictive Control design for shallow GRU models

In this chapter, two control strategies based on NMPC are proposed for shallow GRU models. The trained model of the plant, i.e. (5.1), is here considered to be a shallow GRU characterized by the equations (3.26). Moreover, this model is assumed to be exponentially ℓ_∞-δISS with respect to the invariant set $\mathcal{X}$ (3.27) and the input set $\mathcal{U}$ (3.7). Such property, which can be guaranteed, e.g., by means of Theorem 3.8, for GRU models amounts of assuming the existence of $\lambda_\delta \in (0, 1)$ and $\mu_\delta > 0$ such that Definition 2.6 applies with the function β in the form

$$\beta(\|x_{a,0} - x_{b,0}\|_\infty, k) = \mu_\delta \lambda_\delta^k \|x_{a,0} - x_{b,0}\|_\infty, \tag{6.1}$$

and for some function γ of class $\mathcal{K}_\infty$.

Remark 6.1. *Theorem 3.8 allows to conservatively estimate μ_δ and λ_δ. Indeed, from (A.146) it holds that*

$$\mu_\delta = 1, \tag{6.2}$$

while, from (A.140), one can derive that

$$\lambda_\delta \leq \max\big(\kappa_x(\check{\sigma}_z), \kappa_x(1 - \check{\sigma}_z)\big) \tag{6.3a}$$

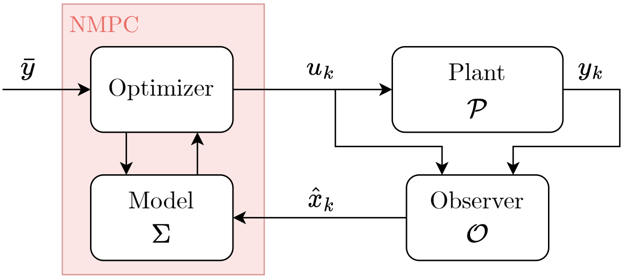

Figure 6.1: *Schematic of the proposed NMPC control architecture.*

where $\kappa_x(z)$ *is*

$$\kappa_x(z) = z + (1-z)\Big(\frac{1}{4}\check{x}\|U_f\|_\infty + \check{\sigma}_f\Big)\|U_r\|_\infty + \frac{1}{4}(\check{\phi}_r + \check{x})\|U_z\|_\infty. \tag{6.3b}$$

Regardless of how these constants are calculated, numerically or conservatively by (6.2) and (6.3), we assume that they are known. We are now in the position to describe the proposed control strategies to address Problem 5.1 for the shallow GRU model under consideration.

6.1 Closed-loop stable NMPC design

In this section, the control scheme depicted in Figure 6.1 is formally introduced. Such a scheme is a rather standard one, typically adopted when NMPC is used for the regulation of systems with unmeasurable states.

Since the state vector x_k is not measured, a state observer is needed to retrieve a state estimate $\hat{x}_k$. Such state estimate is then used to initialize the predictive model (i.e., the shallow GRU model $\Sigma(\Phi^\star)$) of the NMPC's FHOCP, whose solution leads to the optimal control action u_k to be applied.

The synthesis of the control scheme, therefore, boils down to two steps: (*i*) the design of a nominally convergent state observer $\mathcal{O}$, i.e., an observer yielding a state estimate nominally converging to the real, unknown, state; (*ii*) the synthesis of an FHOCP that, exploiting this convergent state estimate, allows to guarantee the nominal closed-loop stability.

In the following, these two tasks are therefore described.

6.1.1 State observer design

In the spirit of [22,60], to define a state estimate $\hat{x}_k$ with convergence guarantees, we propose to adopt a Luenberger-like structure closely resembling that of the shallow GRU system (3.26) whose states are to be estimated.

Therefore, the candidate state observer reads as

$$\mathcal{O}(\Phi_o): \begin{cases} \hat{x}_{k+1} = \hat{z}_k \circ \hat{x}_k + (1 - \hat{z}_k) \circ \hat{r}_k \\ \hat{y}_k = U_o \hat{x}_k + b_o \end{cases} \tag{6.4a}$$

where $\hat{x}_k \in \mathbb{R}^{n_x}$ denotes the observer's state. The observer's gates $\hat{z}_k$ and $\hat{f}_k$, and the squashed input $\hat{r}_k$ are defined as

$$\begin{aligned} \hat{z}_k &= \sigma(W_z u_k + U_z \hat{x}_k + b_z + L_z(y_k - \hat{y}_k)), \\ \hat{f}_k &= \sigma(W_f u_k + U_f \hat{x}_k + b_f + L_f(y_k - \hat{y}_k)), \\ \hat{r}_k &= \phi(W_r u_k + U_f \hat{f}_k \circ \hat{x}_k + b_f). \end{aligned} \tag{6.4b}$$

The overall set of weights of the GRU observer (6.4) is $\Phi_o = \Phi^\star \cup \{L_f, L_Z\}$, where $\Phi^\star$ matches the set of weights of the GRU system to be observed, and is therefore fixed. The tuning parameters of the observer are the gains L_f and L_z, that allow to exploit the known innovation $y_k - \hat{y}_k$ to improve the future state estimation $\hat{x}_{k+1}$.

Notation Addendum 6.1. *In the following, the observer $\mathcal{O}(\Phi_o)$ may be represented in the compact form*

$$\mathcal{O}(\Phi_o): \begin{cases} \hat{x}_{k+1} = f_o(\hat{x}_k, u_k, y_k; \Phi_o) \\ \hat{y}_k = g(\hat{x}_k; \Phi_o) \end{cases}. \tag{6.5}$$

Moreover, in accordance to Notation Addendum 2.1, we may denote by $\hat{x}_k(\hat{x}_0, u_{0:k}, y_{0:k}; \Phi_o)$ the observer state at time k, when (6.5) is initialized in $\hat{x}_0$ and it is fed by the input sequence $u_{0:k}$ and output measurement $y_{0:k}$.

We assume that the observer (6.4) satisfies Assumption 3.2. That is, we assume that the observer's initial state $\hat{x}_0$ is bounded in the same set $\mathcal{X}$ (3.27) that bounds the initial state of the observed GRU system. This ensures that Lemma 3.3 can be applied to (6.4) to show that $\mathcal{X}$ is an invariant set of the observer itself[1].

The observer design problem amounts to finding the gains L_z and L_f that guarantee the convergence of the state estimate to the true state, in the sense specified by the following definition.

Definition 6.1. *The observer $\mathcal{O}(\Phi_o)$ is said to be convergent to the GRU system $\Sigma(\Phi^\star)$ if, for any unknown initial state of the GRU system $x_0 \in \mathcal{X}$, given the sequence of applied inputs $u_{0:k} \in \mathcal{U}_{0:k}$ and the sequence of*

[1] The fact that $\mathcal{X}$ represents an invariant set for the observer can be easily proven by noticing that the innovation enters the gates as an additive argument, and can thus be regarded as an additive term to the input u_k. Accordingly, the proof of Lemma 3.3 can be straightforwardly applied to $\mathcal{O}$.

measured output $y_{0:k}(x_0, u_{0:k}; \Phi^\star)$*, the observed state* $\hat{x}_k(\hat{x}_0, u_{0:k}, y_{0:k}; \Phi_o)$ *converges to the true state* $x_k(x_0, u_{0:k}; \Phi^\star)$ *for any initial guess* $\hat{x}_0 \in \mathcal{X}$*, i.e.*

$$\|\hat{x}_k(\hat{x}_0, u_{0:k}, y_{0:k}; \Phi_o) - x_k(x_0, u_{0:k}; \Phi^\star)\|_2 \leq \beta_o(\|\hat{x}_0 - x_0\|_2, k) \tag{6.6}$$

where $\beta_o \in \mathcal{KL}$*.*

If β_o takes an exponential form, the observer is said to be *exponentially convergent*. In the following theorem we propose a sufficient condition for the nominal exponential convergence of the observer.

Theorem 6.1 (State observer convergence)**.** *Consider the observer gain* L_z *and* L_f*, and assume that there exists* $\lambda_o \in (0, 1)$ *such that,* $\forall z \in [1-\check{\sigma}_z, \check{\sigma}_z]$*,*

$$\kappa_o(z, L_z, L_f) < \lambda_o \tag{6.7a}$$

where

$$\begin{aligned}\kappa_o(z, L_z, L_f) =& z + (1-z)\Big(\frac{1}{4}\check{x}\|U_f - L_f U_o\|_\infty + \check{\sigma}_f\Big)\|U_r\|_\infty \\ &+ \frac{1}{4}(\check{\phi}_r + \check{x})\|U_z - L_z U_o\|_\infty\end{aligned} \tag{6.7b}$$

and $\check{\sigma}_z$*,* $\check{\sigma}_f$*, and* $\check{\phi}_r$ *are defined as in* (3.30)*. Then, the state observer* (6.4) *is nominally exponentially convergent in the sense specified by Definition 6.1, with function* β_o *in the form*

$$\beta_o(\|\hat{x}_0 - x_0\|_2, k) = \mu_o \lambda_o^k \|\hat{x}_0 - x_0\|_2, \tag{6.8}$$

for some $\mu_o > 0$.

Proof. See Appendix A.3.1. □

While Theorem 6.1 allows to assess whether the state observer can be guaranteed to be convergent for specific gains L_z and L_f, it does not provide any guidelines on how to select them. It is worth noticing that, according to (6.8), λ_o represents a bound on the worst-case convergence rate of the observer. Therefore, in the following proposition we recast the observer design problem as a convex optimization program, in order to retrieve the gains that ensure the smallest possible bound on the convergence rate λ_o.

Proposition 6.1 (Optimal tuning of state observer)**.** *The gains* L_z *and* L_f *of the state observer* (6.4) *that allow to fulfill Theorem 6.1 while ensuring* *the fastest worst-case convergence rate λ_o, can be computed by solving the following convex optimization problem*

$$\begin{aligned} \lambda_o, L_z^\star, L_f^\star = \arg \min_{\lambda, L_z, L_f} \ & \lambda \\ \textit{s.t.} \quad & \kappa_o(\check{\sigma}_z, L_z, L_f) \leq \lambda \\ & \kappa_o(1 - \check{\sigma}_z, L_z, L_f) \leq \lambda \\ & 0 < \lambda < 1 \end{aligned}, \tag{6.9}$$

where $\kappa_o(z, L_z, L_f)$ is defined as in (6.7b).

Proof. See Appendix A.3.2. □

Remark 6.2. *As discussed in the proof of Proposition 6.1, the δISS of $\Sigma(\Phi^\star)$ allows to guarantee that the optimization problem* (6.9) *admits a feasible solution. This is expected, since the suboptimal open-loop observer ($L_z = L_f = 0_{n_x, n_y}$) corresponds to the GRU model $\Sigma(\Phi^\star)$ itself, with convergent state trajectories in view of the δISS definition.*

6.1.2 FHOCP formulation

Having synthesized a state observer that yields a convergent observation $\hat{x}_k$, we now propose a formulation of the FHCOP, to be solved at time instant k, which allows to steer the system to the target equilibrium $\bar{\Sigma}$. The FHOCP is formulated relying on the predictive model of the system, $\Sigma(\Phi^\star)$, which allows to predict the future state trajectories throughout the prediction horizon N, given the current state estimate and the input sequences applied along the control horizon $N_c < N$. This prediction mechanism not only allows to enforce constraints on future inputs and states, but also to encode the desired closed-loop behavior in terms of a cost function that depends on future inputs and states.

For clarity, at time k, we denote by

$$u_{k:k+N_c-1|k} = \{u_{k|k}, ..., u_{k+N_c-1|k}\}, \tag{6.10}$$

the sequence of future inputs applied throughout the control horizon $\mathcal{N}_c = \{0, ..., N_c - 1\}$, after which the constant input $\bar{u}$ is applied. Similarly, we denote by

$$x_{k:k+N|k} = \{x_{k|k}, ..., x_{k+N|k}\}, \tag{6.11}$$

the predicted state trajectory throughout the prediction horizon $\mathcal{N} = \{0, ..., N - 1\}$, where $x_{k|k} = \hat{x}_k$. Note that, in (6.11), the term $x_{k+t|k}$ denotes the predicted state at time $k+t$ given the input sequence[2] $u_{k:k+t|k}$, i.e. $x_{k+t|k} = x_{k+t}(x_{k|k}, u_{k:k+t|k}; \Sigma(\Phi^\star))$. Under this notation, the nonlinear FHOCP can be therefore stated as follows.

$$\min_{u_{k:k+N_c-1|k}} \Bigg\{ J_k = \sum_{\tau=0}^{N_c-1} \Big(\|x_{k+\tau|k} - \bar{x}\|_Q^2 + \|u_{k+\tau|k} - \bar{u}\|_R^2 \Big) \tag{6.12a}$$

$$+ \sum_{\tau=N_c}^{N} \|x_{k+\tau|k} - \bar{x}\|_S^2 \Bigg\}$$

$$\text{s.t.} \quad x_{k|k} = \hat{x}_k \tag{6.12b}$$

$$x_{k+\tau+1|k} = f(x_{k+\tau|k}, u_{k+\tau|k}; \Sigma(\Phi^\star)) \qquad \forall \tau \in \mathcal{N}_c \tag{6.12c}$$

$$x_{k+\tau+1|k} = f(x_{k+\tau|k}, \bar{u}; \Sigma(\Phi^\star)) \qquad \forall \tau \in \mathcal{N} \setminus \mathcal{N}_c \tag{6.12d}$$

$$u_{k+\tau|k} \in \tilde{\mathcal{U}} \qquad \forall \tau \in \mathcal{N}_c \tag{6.12e}$$

In the above formulation, the dynamics of the shallow GRU predictive model $\Sigma(\Phi^\star)$ is embedded by means of the constraints (6.12c) and (6.12d). In particular, throughout the control horizon $\mathcal{N}_c$, the state evolution is ruled by (6.12c), after which the constant control action $\bar{u}$ is applied, see (6.12d). With constraint (6.12b), the predictive model is initialized in the observed state $\hat{x}_k$, produced by the state observer $\mathcal{O}(\Phi_o)$ designed in the previous section. The input saturation constraint is enforced by (6.12e). It is worth noticing that, since $\tilde{\mathcal{U}} \subseteq \mathcal{U}$, this input constraint also ensures that the model is operated in accordance with Assumption 3.1.

Moreover, while the states are guaranteed to lie within the invariant set $\mathcal{X}$, it is not possible to further constrain the states in a tighter set. In any case, this limitation is not overly restrictive: indeed, since the system is a black-box model, states have no physical meaning, therefore imposing constraints on the state is generally not of interest.

The cost function J_k in (6.12a) is a quadratic cost function, which penalizes the deviations (throughout the control horizon) of the predicted state trajectories from the equilibrium $\bar{x}$, and the deviations of the input sequence from the equilibrium $\bar{u}$, weighted by the positive-definite weight matrices Q and R, respectively. Quadratic terms also penalize the displacement of the state from the target equilibrium $\bar{x}$ after the control horizon, and they are weighted by the positive-definite matrix S. The minimum and maximum singular values of these matrices are respectively denoted by $0 < \underline{\varsigma}_Q \leq \bar{\varsigma}_Q$, $0 < \underline{\varsigma}_R \leq \bar{\varsigma}_R$, and $0 < \underline{\varsigma}_S \leq \bar{\varsigma}_S$.

[2] For the sake of consistency of notation, $u_{k:k+N|k}$ is here used to denote the control sequence (6.10) concatenated with the constant control action $\bar{u}$.

According to the receding horizon principle, at any instant k the optimal control sequence $u^{\star}_{k:k+N_c-1|k}$ is computed by solving the FHOCP (6.12), and the fist optimal control input $u_k = u^{\star}_{k|k} = \kappa_{\text{MPC}}(\hat{x}_k)$ is applied.

In the following theorem sufficient conditions for the closed-loop stability of the proposed NMPC law are provided.

Theorem 6.2. *A sufficient condition for the closed-loop stability of the NMPC law $u_k = \kappa_{MPC}(\hat{x}_k)$ associated to the FHOCP (6.12) is that the weight matrices Q and S are designed such that*

$$\bar{\varsigma}_Q < \underline{\varsigma}_S \tag{6.13a}$$

and that the prediction horizon $N - N_c$ is large enough to satisfy

$$N - N_c > \frac{1}{2} \log_{\lambda_\delta} \left(\frac{\underline{\varsigma}_S - \bar{\varsigma}_Q}{n_x \mu_\delta^2 \bar{\varsigma}_S} \right) - 1. \tag{6.13b}$$

Proof. See Appendix A.3.3. □

At this stage, it is worth to highlight the peculiarities of the proposed NMPC strategy. First, the weight matrix S, associated with the penalty of states beyond the control horizon, is arbitrary, as long as it is positive definite and satisfies condition (6.13a). This is rather different from the usual formulations, where the terminal cost is computed as an approximation of the so-called cost-to-go, i.e., the cost required to steer the system from the terminal state $x_{k+N|k}$ to equilibrium under a suitably designed auxiliary control law [139]. Similarly, no terminal constraint is required in (6.12). Such constraint generally involves the computation of a terminal set, i.e., a set within which an auxiliary control law able to steer the system to the equilibrium is guaranteed to exist. The absence of these two terminal ingredient allows to simplify the control design phase and to avoid an overly conservative control. Finally, it can be observed that condition (6.13b) implies that the prediction horizon must be sufficiently longer than the control horizon. In this sense, the control horizon can be designed so as to improve the dynamic performances of the control scheme, while the prediction horizon, taken sufficiently long, allows the nominal closed-loop stability to be guaranteed.

Remark 6.3. *The attainment of nominal closed-loop stability guarantees via the adoption of a sufficiently long prediction horizon is reminiscent of closed-loop stability framework devised in [140], in which the authors proved that terminal cost and terminal constraint can be removed if quasi-infinite horizons are adopted. Such strategy conveniently makes it unnecessary to include in the NMPC formulation the "terminal ingredients," which*

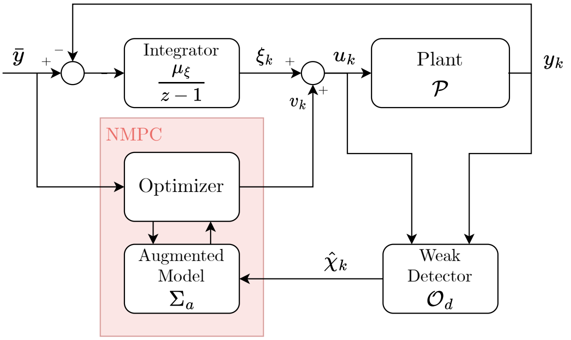

Figure 6.2: *Proposed offset-free NMPC architecture for shallow GRU models.*

are not only difficult and onerous to compute, especially online, but can also lead to overly conservative control laws [139]. On the other hand, it should be pointed out that if $\lambda_\delta \approx 1$, condition (6.13b) might imply an excessively long prediction horizon which, in turn, may yield an high computational burden.

6.2 Offset-free NMPC design based on augmented system

In the following, a second approach to address Problem 5.1 is proposed. In particular, we consider an offset-free NMPC law based on the control architecture devised in [117], whose theoretical framework allows to guarantee the nominal closed-loop stability and the asymptotic zero-error output regulation for square models, i.e. $n_u = n_y$.

The control scheme here proposed, depicted in Figure 6.2, has been discussed in [60]. The design of this control scheme relies on the inclusion of an integrator on the output tracking error guarantees, in line with the Internal Model Principle [141]. Namely, if the stability of the augmented system, consisting of the system model and the integrator, can be ensured, the control scheme attains asymptotic zero-error output tracking of constant references. According to the strategy proposed in [117], the closed-loop nominal asymptotic stability and constraint satisfaction can be guaranteed through two ingredients, i.e., a weak detector of the augmented system (that is, as discussed later, a state observer enjoying nominal convergence guarantees) and an NMPC scheme stabilizing such augmented system.

Note that, unlike the approach discussed in Section 6.1, the control scheme discussed here relies on existing results, i.e. the design of stabilizing NMPC laws, to attain the desired stability and performance guarantees.

The control architecture is detailed in three distinct steps: (*i*) model enlargement with integrators on the output tracking error, (*ii*) design of a weak detector for the augmented system, and (*iii*) synthesis of a suitable NMPC law stabilizing the augmented system.

6.2.1 System augmentation

Consider the model (5.1) which, we recall, is here assumed to be described by a shallow GRU architecture, see (3.26). According to Assumption 5.1, the reference equilibrium triplet is denoted by $\bar{\Sigma} = (\bar{x}, \bar{u}, \bar{y})$. As discussed in Remark 5.1, under Assumption 5.2 the local solvability of the output tracking problem can be guaranteed, see also [117]. Therefore, integrators can be placed on the output tracking error $\bar{y} - y_k$ by augmenting the system model with the integrator dynamics, i.e.

$$\xi_{k+1} = \xi_k + \mu_\xi(\bar{y} - y_k). \tag{6.14a}$$

Here $\xi_k \in \mathbb{R}^{n_u}$ denotes the state of the integral action, characterized by the gain μ_ξ, which is a design parameter of the control scheme whose choice is discussed in the following. As shown in Figure 6.2, the control action applied to the system is given by the sum of the integral action ξ_k and the exogenous control variable v_k, i.e.

$$u_k = \xi_k + v_k. \tag{6.14b}$$

The rationale behind these two control components is that the integral action is intended to achieve the desired static performances, i.e asymptotic zero-error output tracking, while v_k is the NMPC control action which allows to improve the dynamic performances and to ensure constraint satisfaction during transients.

By combining the components (6.14) with the system model (5.1), the augmented model is

$$\Sigma_a(\Phi^\star) : \begin{cases} x_{k+1} = f(x_k, u_k; \Phi^\star) \\ \xi_{k+1} = \xi_k + \mu_\xi(\bar{y} - y_k) \\ u_k = v_k + \xi_k \\ y_k = g(x_k; \Phi^\star) \end{cases}, \tag{6.15}$$

In the following, such augmented model is compactly described by the system

$$\Sigma_a(\Phi^\star) : \begin{cases} \chi_{k+1} = \varphi(\chi_k, v_k, \bar{y}; \Phi^\star) \\ \zeta_k = \psi(\chi_k; \Phi^\star) \end{cases}, \tag{6.16}$$

whose state and output vectors are

$$\begin{aligned} \chi_k &= [x_k', \xi_k']', \\ \zeta_k &= [y_k', \xi_k']'. \end{aligned} \tag{6.17}$$

The target equilibrium of the augmented system, indicated by $\bar{\Sigma}_a = (\bar{\chi}, \bar{v}, \bar{\zeta})$, can be straightforwardly defined from $\bar{\Sigma}$ by means of the following equations

$$\begin{aligned} \bar{\chi} &= [\bar{x}', \bar{u}']', \\ \bar{v} &= 0_{n_u,1}, \\ \bar{\zeta} &= [\bar{y}', \bar{u}']'. \end{aligned} \tag{6.18}$$

Note that, consistently with the definition of the control action given in (6.14b), the steady-state constant input is assumed to be entirely supplied by the integral action, i.e. $\bar{\xi} = \bar{u}$ and $\bar{v} = 0_{n_u,1}$.

Choice of the integrator gain

Before describing the remaining ingredients of the control architecture, note that the choice of integral action gain μ_ξ is, in practice, crucial to achieve satisfactory closed-loop performance. Therefore, it is useful to provide a range of values within which this gain should be chosen.

Since the model $\Sigma(\Phi^\star)$ is described by a shallow GRU that is exponentially δISS with respect to the sets $\mathcal{X}$ and $\mathcal{U}$, the Schur stability of the linearized system matrix $A_\delta(\bar{\Sigma})$ is guaranteed by Theorem 2.1. This allows the following corollary to be formulated.

Corollary 6.1. *There exists a scalar $\check{\kappa}_\mu > 0$ such that, for any $\kappa_\mu \in (0, \check{\kappa}_\mu)$, the integrator gain*

$$\mu_\xi = \kappa_\mu \big[C_\delta(\bar{\Sigma}) \big(I_{n_x,n_x} - A_\delta(\bar{\Sigma}) \big)^{-1} B_\delta(\bar{\Sigma}) \big]^{-1} \tag{6.19}$$

makes the augmented system (6.16) *locally asymptotically stable around the equilibrium $\bar{\Sigma}_a$.*

Proof. Since $A_\delta(\bar{\Sigma})$ is Schur, and in light of Assumption 5.2, the corollary immediately follows from the results reported in [142]. □

Corollary 6.1 provides a useful guideline for the tuning of the gain μ_ξ. As illustrated in [142], large values of κ_μ lead to a faster convergence of ξ_k to its equilibrium, at the price, however, of possible overshoots and of a reduction in the stability margins of the augmented system. On the other hand, smaller values of κ_μ lead to a slower convergence of the output to its reference value. In the following, we assume that μ_ξ has been suitably tuned in accordance with Corollary 6.1.

6.2.2 Weak detector design

Following the steps of [117], a state observer must be designed for the augmented system. This state observer, here labeled as $\mathcal{O}_d$, takes the following generic form

$$\mathcal{O}_d(\Phi_d) : \begin{cases} \hat{\chi}_{k+1} = \varphi_d(\hat{\chi}_k, v_k, \zeta_k, \bar{y}; \Phi_d) \\ \hat{\zeta}_k = \psi(\hat{\chi}_k; \Phi_d) \end{cases}, \tag{6.20}$$

where Φ_d is the set of weights of the observer, later detailed. This state observer must be a weak detector of the augmented system with respect to the equilibrium $\bar{\Sigma}_a = (\bar{\chi}, \bar{v}, \bar{\zeta})$, in the sense specified by the following definition [143].

Definition 6.2 (Weak detector). *The state observer* (6.20) *is said to be a weak detector of the augmented system* (6.16) *with respect to the equilibrium $\bar{\Sigma}_a = (\bar{\chi}, \bar{v}, \bar{\zeta})$ and to the set $\mathcal{Z}$ if*

i. $\bar{\chi} = \varphi_d(\bar{\chi}, \bar{v}, \bar{\zeta}, \bar{y}; \Phi_d)$

ii. there exists a continuous function V_d, and functions α_{d1}, α_{d2}, and α_{d3} of class $\mathcal{K}_\infty$ such that, for $(\chi_k, v_k) \in \mathcal{Z}$ and $(\hat{\chi}_k, v_k) \in \mathcal{Z}$, it holds that

$$\begin{aligned} \alpha_{d1}(\|\hat{\chi}_k - \chi_k\|_2) \leq V_d(\hat{\chi}_k, \chi_k) \leq \alpha_{d2}(\|\hat{\chi}_k - \chi_k\|_2), \\ V_d(\hat{\chi}_{k+1}, \chi_{k+1}) - V_d(\hat{\chi}_k, \chi_k) \leq -\alpha_{d3}(\|\hat{\chi}_k - \chi_k\|_2), \end{aligned} \tag{6.21}$$

where $\hat{\chi}_{k+1} = \varphi_d(\hat{\chi}_k, v_k, \zeta_k, \bar{y}; \Phi_d)$ and $\chi_{k+1} = \varphi(\chi_k, v_k, \bar{y}; \Phi^\star)$

In line with [60], the candidate weak detector here proposed is a Luenberger-type state observer described by the following equations

$$\mathcal{O}_d(\Phi_d) : \begin{cases} \hat{x}_{k+1} = \hat{z}_k \circ \hat{x}_k + (1 - \hat{z}_k) \circ \hat{r}_k \\ \hat{\xi}_{k+1} = \hat{\xi}_k + \mu_\xi(\bar{y} - \hat{y}_k) + L_{\xi y}(y_k - \hat{y}_k) + L_{\xi\xi}(\xi_k - \hat{\xi}_k) \\ \hat{y}_k = U_o \hat{x}_k + b_o \end{cases} \tag{6.22a}$$

where $\hat{z}_k$, $\hat{f}_k$, and $\hat{r}_k$ are defined as

$$\begin{aligned} \hat{z}_k &= \sigma\big(W_z(v_k + \hat{\xi}_k) + U_z \hat{x}_k + b_z + L_{zy}(y_k - \hat{y}_k) + L_{z\xi}(\xi_k - \hat{\xi}_k)\big), \\ \hat{f}_k &= \sigma\big(W_f(v_k + \hat{\xi}_k) + U_f \hat{x}_k + b_f + L_{fy}(y_k - \hat{y}_k) + L_{f\xi}(\xi_k - \hat{\xi}_k)\big), \\ \hat{r}_k &= \phi\big(W_r(v_k + \hat{\xi}_k) + U_r \hat{f}_k \circ \hat{x}_k + b_r\big). \end{aligned} \tag{6.22b}$$

The observer gains L_{zy}, $L_{z\xi}$, L_{fy}, $L_{f\xi}$, $L_{\xi y}$, and $L_{\xi\xi}$ are matrices of suitable dimensions, and need to be designed to ensure that the observer meets Definition 6.2. Overall, the set of weights of the observer is $\Phi_d = \Phi^\star \cup \Phi_L$, where Φ_L denotes the set of gains, i.e.

$$\Phi_L = \{L_{zy}, L_{z\xi}, L_{fy}, L_{f\xi}, L_{\xi y}, L_{\xi\xi}\}.$$

The following theoretical result thus establish sufficient conditions on Φ_L under which (6.22) is guaranteed to be a weak detector of the augmented system.

Theorem 6.3. *Consider the augmented system* (6.16)*, and the state observer* (6.22)*. Let*

$$\begin{aligned} \check{\kappa}_{dx}(L_{zy}, L_{fy}) &= \max\big(\kappa_{dx}(\check{\sigma}_z, L_{zy}, L_{fy}), \kappa_{dx}(1-\check{\sigma}_z, L_{zy}, L_{fy})\big), \\ \check{\kappa}_{d\xi}(L_{z\xi}, L_{f\xi}) &= \kappa_{d\xi}(1-\check{\sigma}_z, L_{z\xi}, L_{f\xi}), \end{aligned} \tag{6.23}$$

where the functions κ_{dx} and $\kappa_{d\xi}$ are defined as

$$\begin{aligned} \kappa_{dx}(z, L_{zy}, L_{fy}) = z + (1-z)\|U_r\|_\infty \Big(\frac{1}{4}\check{x}\|U_f - L_{fy}U_o\|_\infty + \check{\sigma}_f\Big) \\ + \frac{1}{4}(\check{x} + \check{\phi}_r)\|U_z - L_{zy}U_o\|_\infty \end{aligned} \tag{6.24a}$$

$$\begin{aligned} \kappa_{d\xi}(z, L_{z\xi}, L_{f\xi}) =&(1-z)\Big(\|W_r\|_\infty + \frac{1}{4}\check{x}\|U_r\|_\infty\|W_f - L_{f\xi}\|_\infty\Big) \\ &+ \frac{1}{4}(\check{x} + \check{\phi}_r)\|W_z - L_{z\xi}\|_\infty \end{aligned} \tag{6.24b}$$

and $\check{\sigma}_z$, $\check{\sigma}_f$, and $\check{\phi}_r$ are defined as in (3.30)*. Then, if the gains of the observer are such that the matrix*

$$\mathfrak{A}_d = \begin{bmatrix} \check{\kappa}_{dx}(L_{zy}, L_{fy}) & \check{\kappa}_{d\xi}(L_{z\xi}, L_{f\xi}) \\ \|U_o\|_\infty \|\mu_\xi + L_{\xi y}\|_\infty & \|I_{n_y,n_y} - L_{\xi\xi}\|_\infty \end{bmatrix} \tag{6.25}$$

is Schur stable, the state observer is a weak detector of the augmented system with respect to the equilibrium $(\bar{\chi}, \bar{v}, \bar{\zeta})$ and to the set

$$\mathcal{Z} = \{(\chi, v) : \chi = [x', \xi']' \wedge x \in \mathcal{X} \wedge (\xi + v) \in \tilde{\mathcal{U}}\}, \tag{6.26}$$

in the sense specified by Definition 6.2.

Proof. See Appendix A.3.4. □

We now provide a necessary and sufficient condition for the Schur stability of matrix $\mathfrak{A}_d$, in the form of a pair of explicit inequalities on the observer gains Φ_L.

Proposition 6.2. *A necessary and sufficient condition for the Schur stability for the matrix $\mathfrak{A}_d$ defined in* (6.25) *is that the gains satisfy the following conditions*

$$\big(1 - \breve{\kappa}_{dx}(L_{zy}, L_{fy})\big)\big(1 - \|I_{n_y,n_y} - L_{\xi\xi}\|_\infty\big) \geq \breve{\kappa}_{d\xi}(L_{z\xi}, L_{f\xi})\|U_o\|_\infty\|\mu_\xi + L_{\xi y}\|_\infty \tag{6.27a}$$

and

$$\breve{\kappa}_{dx}(L_{zy}, L_{fy})\|I_{n_y,n_y} - L_{\xi\xi}\|_\infty \leq \breve{\kappa}_{d\xi}(L_{z\xi}, L_{f\xi})\|U_o\|_\infty\|\mu_\xi + L_{\xi y}\|_\infty + 1. \tag{6.27b}$$

That is, if (6.27) *are fulfilled, the state observer* (6.22) *is a weak detector of the augmented system* (6.16) *over $\mathcal{Z}$.*

Proof. By applying Lemma A.1, one gets that (6.27) are necessary and sufficient conditions for the Schur stability of matrix $\mathfrak{A}_d$. Therefore, if conditions (6.27) are fulfilled, Theorem 6.3 can be applied to guarantee that (6.22) is a weak detector of the augmented system. □

In light of Proposition 6.2, it is possible to setup a nonlinear optimization problem to compute the observer gains, which reads as follows

$$\begin{aligned} \min_{\Phi_L} \quad & \|\mathfrak{A}_d\|_2 \\ \text{s.t.} \quad & (6.27) \end{aligned} \tag{6.28}$$

In this optimization problem, the gains satisfying Proposition 6.2 that minimize the induced 2-norm of the matrix $\mathfrak{A}_d$ are retrieved. The idea behind this choice is that the induced 2-norm is an upper bound of the spectral radius of $\mathfrak{A}_d$, and thus minimizing $\|\mathfrak{A}_d\|_2$ likely yields a faster detector. As formalized in the following corollary, the exponential δISS of the GRU model $\Sigma(\Phi^\star)$ allows to guarantee the existence of a feasible solution to the optimization problem (6.28), and hence the existence of a weak detector of the augmented system, which is a requirement of the control architecture here proposed.

Corollary 6.2. *The exponential δISS of the GRU model $\Sigma(\Phi^\star)$ guarantees the existence of a feasible solution to the conditions* (6.27).

Proof. A feasible set of gains satisfying (6.27) is $L_{zy} = L_{fy} = 0_{n_x,n_y}$, $L_{\xi y} = -\mu_\xi$, $L_{\xi\xi} = \varsigma_\xi I_{n_y,n_y}$ with $\varsigma_\xi \in (0, 1)$, and any $L_{z\xi}$ and $L_{f\xi}$. Indeed, under this choice $\kappa_{dx}(L_{zy}, L_{fy}) = \kappa_x(z)$, see (6.3a) which, in view of Remark 6.1, is guaranteed to be sub-unitary. Then, since the left-hand side of (6.27a) is non-negative and the right hand-size is zero, condition (6.27a) is surely fulfilled. Similarly, the left-hand side of (6.27b) is sub-unitary, while the right hand side is 1, which implies that (6.27b) is fulfilled. □

6.2.3 FHOCP formulation

The last ingredient of the control architecture that needs to be designed is an NMPC law that stabilizes the augmented system, see again Figure 6.2. While several NMPC laws can be adopted to this end, here we adopt the strategy proposed in [139].

Therefore, at the generic time-step k, the state observation $\hat{\chi}_k$ produced by the weak detector (6.22) is sampled, as well as the state of the integrator ξ_k, and an FHOCP is solved minimizing the cost function over the prediction horizon $\mathcal{N} = \{0, ..., N-1\}$ to retrieve the optimal control sequence. In particular, the following standard stabilizing FHOCP formulation is considered

$$
\begin{aligned}
&\min_{v_{k:k+N-1|k}} \sum_{\tau=0}^{N-1} \Big(\|\chi_{k+\tau|k} - \bar{\chi}\|_Q^2 + \|v_{k+\tau|k}\|_R^2 \Big) + V_f(\chi_{k+N|k}, \bar{\chi}) && \text{(6.29a)} \\
&\text{s.t.} \quad \chi_{k|k} = \hat{\chi}_k && \text{(6.29b)} \\
&\qquad \tilde{\xi}_{k|k} = \xi_k && \text{(6.29c)} \\
&\qquad \chi_{k+\tau+1|k} = \varphi(\chi_{k+\tau|k}, v_{k+\tau|k}, \bar{y}) \qquad \forall \tau \in \mathcal{N} && \text{(6.29d)} \\
&\qquad \tilde{\xi}_{k+\tau+1|k} = \tilde{\xi}_{k+\tau|k} + \mu_\xi(\bar{y} - E_y \psi(\chi_{k+\tau|k})) \qquad \forall \tau \in \mathcal{N} && \text{(6.29e)} \\
&\qquad (\chi_{k+\tau|k}, v_{k+\tau|k}) \in \mathcal{Z} \qquad \forall \tau \in \mathcal{N} && \text{(6.29f)} \\
&\qquad v_{k+\tau|k} + \tilde{\xi}_{k+\tau|k} \in \tilde{\mathcal{U}} \qquad \forall \tau \in \mathcal{N} && \text{(6.29g)} \\
&\qquad \chi_{k+N|k} \in \Omega_{\bar{\chi}} && \text{(6.29h)}
\end{aligned}
$$

In the proposed FHOCP, the augmented system model (6.16) is embedded as predictive model via (6.29d). This model is initialized in the weak detector's state $\hat{\chi}_k$ by means of constraint (6.29b). As illustrated in [117], the integrator dynamics are also emulated in order to enforce the input constraint satisfaction. In (6.29c) the emulated integrator state $\tilde{\xi}_{k|k}$ is therefore initialized in the actual integrator state ξ_k. The emulated integrator state is then propagated throughout the prediction horizon by means of constraint (6.29e), where E_y is a selection matrix that extracts the component $y_{k+\tau|k}$ from $\psi(\chi_{k+\tau|k})$. The constraint satisfaction is thus enforced throughout the prediction horizon via (6.29f) and (6.29g).

The cost function to be minimized, stated in (6.29a), penalizes the deviation of the augmented state $\chi_{k+\tau|k}$ from its equilibrium $\bar{\chi}$, weighted by the positive-definite matrix Q, as well as the input component $v_{k+\tau|k}$, weighted by the positive-definite matrix R.

The term $V_f(\chi_{k+N|k}, \bar{\chi})$ appearing in the cost function (6.29a), as well as the set $\Omega_{\bar{\chi}}$ appearing in (6.29h), are the so-called terminal cost and terminal set, respectively. The former is defined as the cost-to-go from the terminal state $\chi_{k+N|k}$ to the equilibrium $\bar{\chi}$ under some auxiliary control law; the latter is defined as a set (ideally, as large as possible) within which the auxiliary control law is guaranteed to steer the system state $\chi_{k+N|k}$ to the equilibrium $\bar{\chi}$ while always respecting the input and state constraints. For formal definitions of these terminal ingredients and practical methods to numerically approximate them, the reader is addressed to [139, 144].

Lastly, we denote by $v^{\star}_{k:k+N-1|k}$ the optimal control sequence obtained by solving the FHOCP (6.29). In light of the RH principle, the input $v^{\star}_{k|k} = \kappa_{\text{MPC}}(\hat{\chi}_k, \xi_k)$ is applied to the augmented system, and hence $u_k = \xi_k + v^{\star}_{k|k}$. At the following time instant the entire procedure is then repeated.

Remark 6.4. *Under standard assumptions on the Lipschitz continuity of κ_{MPC}, see [117, 145], Theorem 3 of [117] implies that the proposed control architecture guarantees offset-free output regulation as well as nominal closed-loop stability in a neighborhood of the equilibrium $\bar{\Sigma}_a$.*

Remark 6.5. *A special case of FHOCP* (6.29) *occurs the terminal set coincides with the equilibrium point $\bar{\chi}$, i.e. $\Omega_{\bar{\chi}} = \{\bar{\chi}\}$. In this case, the terminal cost $V_f(\chi_{k+N|k}, \bar{\chi})$ is also not necessary, since the $\chi_{k+N|k} = \bar{\chi}$. This simplification, however, comes at the price of a significant increase of the computational burden, since a longer prediction horizon is generally needed to maintain the feasibility of the resulting FHOCP.*

6.3 Numerical example

To evaluate the performance of the proposed control schemes, let us again consider the pH neutralization process presented in Chapter 4.3. As a model of this plant, we consider the shallow GRU with $n_c = 7$ units illustrated in Section 4.3.2. Moreover, the untightned input set $\tilde{\mathcal{U}} = \mathcal{U}$ is here considered for the sake of simplicity.

Below, we first discuss the design of the NMPC scheme presented in Section 6.1. We then focus on the offset-free NMPC scheme proposed in Section 6.2, and finally the performances of the two schemes are compared.

6.3.1 NMPC synthesis

The synthesis of the NMPC scheme proposed in Section 6.1 boils down to two tasks, i.e. (*i*) tuning the state observer (6.5) so as to guarantee its nominal convergence, and (*ii*) choosing the design parameters of the FHOCP

$Q = I_{7,7}$	$R = 1$	$S = 2Q$	$N = 50$	$N_c = 20$

Table 6.1: *Design parameters of the NMPC law*

(6.12), such as the prediction horizon, the control horizon, and the weight matrices of the cost function.

As for the state observer design, the Luenberger-like observer (6.4) is adopted, and its gains are selected as illustrated in Proposition 6.1. In particular, the observer design problem amounts to solving the convex optimization problem (6.9), carried out with the CVX toolbox [146, 147], resulting in an observer with $\lambda_o = 0.93$.

The FHOCP (6.12) has been setup in accordance to Theorem 6.2. In particular, the state weight matrix has been selected as $Q = I_{n_x,n_x}$, the input weight matrix as $R = 1$, and the terminal weight as $S = 2Q$. Such selection satisfies (6.13a).

Considering a control horizon $N_c = 20$, in order to design the prediction horizon via (6.13b), the terms μ_δ and λ_δ such that (6.1) holds need to be computed. A conservative estimate of these quantities can be computed with (6.2) and (6.3), yielding $\lambda_\delta = 0.997$. This choice leads, however, to an excessively long prediction horizon ($N \geq 460$) due to the conservativeness of this λ_δ.

Therefore, we opted for a numerical estimation of an upper bound $\lambda_\delta \in (0, 1)$ for which, given $\mu_\delta = 1$ from (6.2), condition (6.1) is satisfied for a sufficiently large number of pairs of state trajectories. Such trajectories have been generated by pairs of initial states randomly extracted from $\mathcal{X}$ and pairs of input sequences randomly extracted from $\mathcal{U}_{0:T}$, with T sufficiently high. Note that this numerical approximation is made possible by the fact that the existence of such λ_δ is guaranteed by the model's exponential δISS, see Remark 6.1. Considering 10^5 trajectories of length $T = 300$, the bound $\lambda_\delta \approx 0.9$ has been computed. In view of (6.13b), this value implies $N - N_c \geq 15$. The prediction horizon $N = 50$ has been therefore selected.

Overall, the NMPC design parameters here adopted are summarized in Table 6.1.

6.3.2 Offset-free NMPC synthesis

We now describe the synthesis of the offset-free NMPC scheme proposed in Section 6.2, which consists in three steps, i.e. (*i*) augmenting the plant model with a suitably designed integral action, (*ii*) finding the gains Φ_L

$Q = \mathrm{diag}(I_{7,7}, 5)$	$R = 1$	$\mu_\xi = 0.061$	$N = 50$

Table 6.2: *Design parameters of the offset-free NMPC law*

of augmented system's weak detector (6.20), and (*iii*) choosing the design parameters of the FHOCP (6.29).

In order to tune the integral action gain μ_ξ, the model $\Sigma(\Phi^\star)$ has been linearized around the nominal operating condition, i.e. $\bar{y} = 7$. To this end, we used CasADi [148] to compute the corresponding equilibrium $\bar{\Sigma} = (\bar{x}, \bar{u}, \bar{y})$, such that $\bar{y} = 7$, $\bar{x} \in \mathcal{X}$, and $\bar{u} \in \tilde{\mathcal{U}}$. Using CasADi automatic differentiation engine, we computed the linearized system $\delta\Sigma$ around such equilibrium, leading to the matrices $(A_\delta(\bar{\Sigma}), B_\delta(\bar{\Sigma}), C_\delta(\bar{\Sigma}))$, see (2.15). These matrices have been verified to satisfy Assumption 5.2. The Schur stability of $A_\delta(\bar{\Sigma})$, ensured by Theorem 2.1, was also verified.

On these premises, Corollary 6.1 could be invoked to design μ_ξ, where $\breve{\kappa}_\mu = 0.86$ has been computed numerically. Selecting $\kappa_\mu = 0.043 \in (0, 0.86)$ one has $\mu_\xi = 0.061$. Note that, albeit larger values of κ_μ could have been chosen, the adoption of small values allows for greater "robustness" to variations of the target equilibrium $\bar{\Sigma}$. That is, if $\bar{y}$ (and, consequently, $\bar{\Sigma}$) changes, the designed μ_ξ still ensures the local asymptotic stability of the augmented system's new equilibrium. To this end, we have verified that such property is ensured by $\mu_\xi = 0.061$ for a broad range of output setpoints.

Then, the Luenberger-like state observer described in (6.22) has been tuned to ensure that it is a weak detector of the augmented system. In particular, the nonlinear non-convex optimization problem (6.28) has been solved numerically using MATLAB's *fmincon*, where the trivial solution (see Corollary 6.2) has been used to warm-start the solver, so as to ensure the convergence of the problem to a feasible solution.

Finally, the design parameters of the FHOCP (6.29) have been selected. For the sake of simplicity, as illustrated in Remark 6.5, a zero-terminal constraint has been adopted. Consistently with the NMPC synthesized in Section 6.3.1, a prediction horizon $N = 50$ has been chosen[3]. The state weight matrix has been chosen as $Q = \mathrm{diag}(I_{7,7}, 5)$ and the input weight matrix as $R = 1$. The design parameters of the synthesized offset-free NMPC scheme are summarized in Table 6.2.

[3] In general, when dealing with piecewise-constant references it is hard to guarantee the feasibility of FHOCPs with zero-terminal constraints. While for the considered reference signal $N = 50$ turned out to be sufficient to ensure the feasibility of the FHOCP, one may get infeasibilities for different references. In practice, such case can be addressed either increasing the prediction horizon or relaxing the zero-terminal constraint (6.29h) by means of an heavily-penalized slack variable.

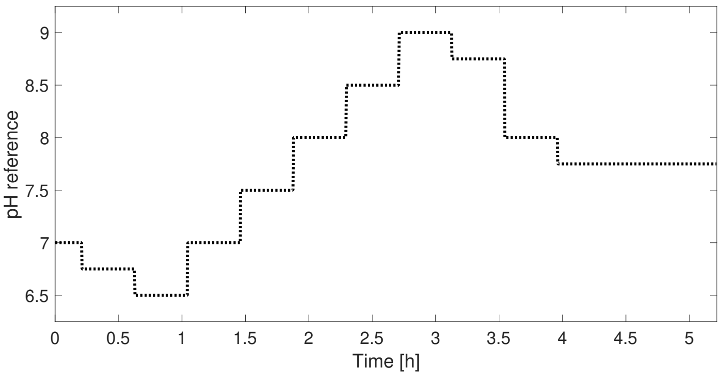

Figure 6.3: *Piecewise-constant output reference trajectory adopted to test the closed-loop performances.*

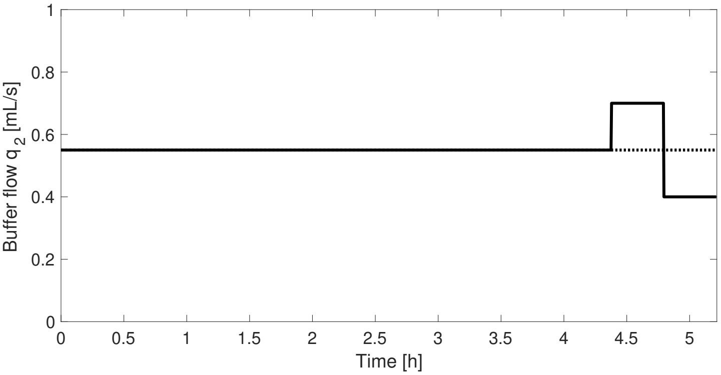

Figure 6.4: *Evolution of the disturbance q_2 (buffer flow rate, black solid line) compared to its nominal value (black dotted line).*

6.3.3 Closed-loop results

Both control architectures have been implemented and tested in MATLAB and Simulink, while the FHOCPs have been solved using CasADi with the IPOPT solver.

The performances of the control scheme have been tested on the piecewise-constant output reference signal depicted in Figure 6.3. To test the robustness of these schemes against disturbances acting on the plant, the buffer flow rate q_2 (see (4.13)) is changed as depicted in Figure 6.4. Specifically, at time $t = 4.38h$ the buffer flow rate is increased to 0.7 ($+27\%$ with respect to the nominal value), while at time $t = 4.8h$ the buffer flow rate is decreased to 0.4 (-27% with respect to the nominal value).

The closed-loop output trajectories achieved by both schemes are com-

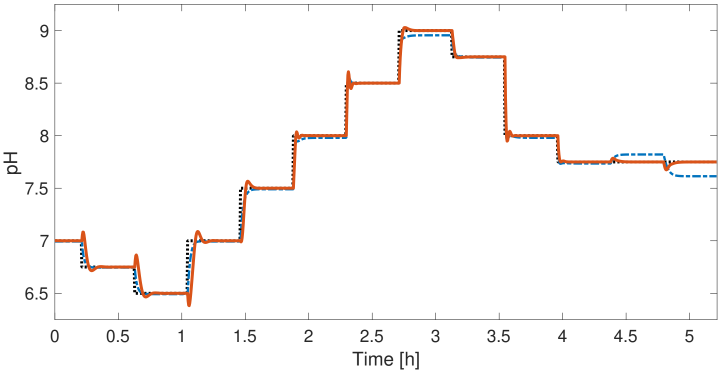

Figure 6.5: *Closed-loop output trajectory achieved by the NMPC scheme (blue dashed-dotted line) and by the offset-free NMPC scheme (red solid line) compared to the piecewise-constant output reference signal (black dotted line).*

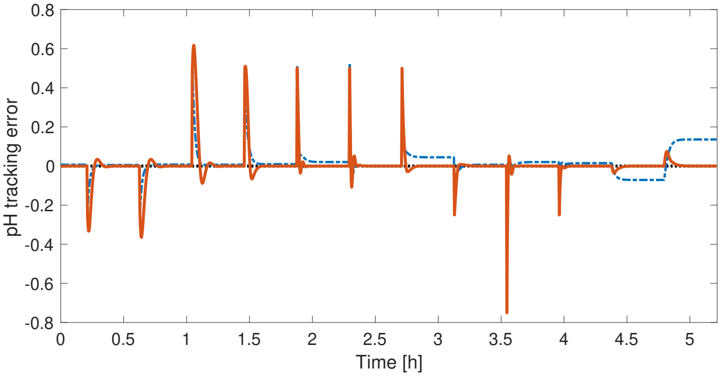

Figure 6.6: *Output tracking error achieved by the NMPC scheme (blue dashed-dotted line) and by the offset-free NMPC scheme (red solid line). The black dotted line indicates the zero error.*

pared to the output reference signal in Figure 6.5. It is easy to notice that, owing to the high accuracy of the GRU model of the plant, the NMPC scheme proposed in Section 6.1 allows to achieve a limited output tracking error. Albeit this error is generally lower than 0.02, for several output reference values, such as $\bar{y} = 8$ and $\bar{y} = 9$, the tracking error becomes not negligible due to the plant-model mismatch, see Figure 6.6. Changes in the buffer flow rate ($t > 4.38h$) erode, however, the output tracking performances of this first scheme.

On the other hand, the offset-free NMPC scheme allows to achieve asymptotic zero error output regulation for all the output reference values, even in presence of disturbances affecting the plant, see again Figure 6.6.

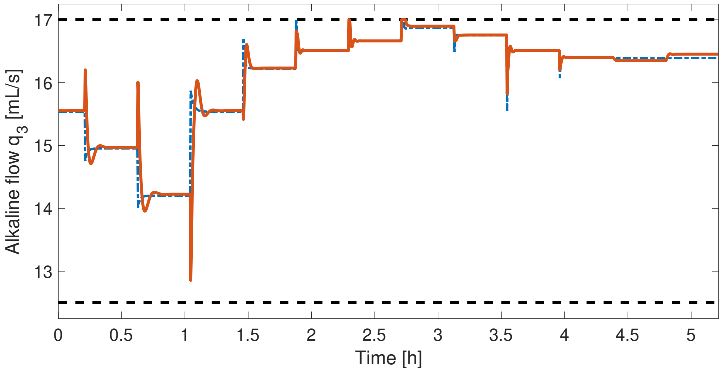

Figure 6.7: *Control variable requested by the NMPC scheme (blue dashed-dotted line) compared to that requested by the offset-free NMPC scheme (red solid line). The black dashed lines indicate the saturation constraint* (4.14).

This static performance comes, however, at the price of slightly worse dynamic performances. In particular, albeit limited, overshoots are present, although these could be mitigated, e.g., by increasing the prediction horizon or adopting a terminal set and terminal cost configuration in place of the simpler zero-terminal constraint. This latter, however, requires to recompute the terminal set and terminal cost when the output reference changes.

Lastly, the control action requested by the two schemes is reported in Figure 6.7. Note that both the control scheme satisfy the input saturation constraint (4.14). Among the two, the offset-free NMPC is less moderated, due to the terminal constraint on the state that needs to be fulfilled. Once more, increasing the prediction horizon or adopting a terminal set and terminal cost configuration may lead to a more moderate control action.

6.4 Summary

In this chapter, two control scheme based on Nonlinear Model Predictive Control (NMPC) have been proposed for systems identified via shallow GRU models.

The first control scheme relies upon a suitably designed state observer with nominal convergence guarantees, which can be easily tuned by solving a convex optimization problem. The underlying Finite Horizon Optimal Control Problem (FHOCP) adopted is reminiscent of quasi-infinite horizon NMPC approaches, and it consists of quadratic stage and terminal costs. Conditions on the weight matrices and on the prediction horizon that guarantee the nominal closed-loop stability of this scheme have been provided.

The second control scheme relies upon the augmentation of the system model with a properly tuned integral control action which, owing to the internal model principle, allows to guarantee offset-free tracking capabilities. After having provided guidelines for the design of the integral action, a weak detector, i.e. a nominally converging state observer, has been proposed for the augmented system. At last, a nominally stabilizing NMPC law have been formulated. This second scheme guarantees asymptotic zero-error output tracking and nominal closed-loop stability.

Finally, the two schemes have been tested on the pH neutralization process benchmark system introduced in the previous chapter, demonstrating remarkable closed-loop performances.

CHAPTER 7

Integral-Derivative MPC design for NNARX models

In the previous chapter, two model predictive control schemes have been proposed for GRU models. As discussed, being GRUs black-box models, state observer are required in order to employ these models for the synthesis of closed-loop predictive control laws.

Designing appropriate state observers, however, is not trivial, especially when theoretical convergence guarantees are sought. To this end, in Chapter 6, it has been shown that the exponential δISS of shallow GRU models allows to design nominally convergent state observers, and similar results have been obtained for shallow LSTM models in [22]. When the control architecture is deployed on the real system, the reliability of the state observer is critical to achieve satisfactory closed-loop performance. In fact, if the CEP is dropped, an inadequate state observer could, in principle, even destabilize the closed loop.

The aim of this chapter is to provide an alternative control scheme, tailored for NNARX models of the plant. As discussed in Chapter 3.1, the main advantage motivating the use of NNARX models is that their state vector is a collection of past input and output data: the state is therefore known at every time instant. This makes the state observer unnecessary, greatly facilitating the synthesis of the control architecture.

The control scheme here proposed, formulated in [61], relies upon the augmentation of the system model with two elements: (*i*) an integrator of the output tracking error, which allows to attain offset-free tracking capabilities, and (*ii*) a derivative action, which allows to ensure that – at steady state – the regulation of the system relies entirely upon the integral action, while the goal of such derivative action, generated by the NMPC, is to improve the closed-loop dynamic performances and to ensure the constraint satisfaction during transients. In this context, the δISS of the model allows, under mild assumptions, to tune the integral control action so as to preserve the local asymptotic stability of the enlarged system. The closed-loop stability can thus be guaranteed in nominal conditions.

In the following, the proposed control architecture is formulated to address Problem 5.1, and its performances are compared to a traditional disturbance estimation-based NMPC law [118] on a benchmark system. Results show that the proposed approach is able to attain offset-free control even in spite of severe disturbances affecting the plant.

7.1 Control architecture

The proposed control architecture assumes that the system model (5.1) is a square ($n_u = n_y$) NNARX model in the form

$$\Sigma(\Phi^\star): \begin{cases} x_{k+1} = f(x_k, u_k; \Phi^\star) \\ y_k = C x_k \end{cases}, \tag{7.1}$$

where the state update function f and the fixed matrix C have been described in Chapter 3.1, see (3.5). The state of the system, i.e. $x_k \in \mathbb{R}^{n_x}$, is known, since it is a collection of the input and output data of the past H time-steps. As discussed in Chapter 5, it is here assumed that the model (7.1) is exponentially δISS, e.g., by Theorem 3.2.

We recall that, given the output setpoint $\bar{y}$, according to Assumption 5.1, there exists a corresponding feasible equilibrium triplet $\bar{\Sigma} = (\bar{x}, \bar{u}, \bar{y})$. Moreover, owing to Assumption 5.2, the linearization of $\Sigma(\Phi^\star)$ around $\bar{\Sigma}$, which is fully described by the triplet $\delta\Sigma(\bar{\Sigma}) = (A_\delta(\bar{\Sigma}), B_\delta(\bar{\Sigma}), C)$, is reachable, observable, and has no invariant zero in $z = 1$. According to Remark 5.1, this guarantees the solvability of the output tracking problem, which puts us in a position to introduce the control architecture depicted in Figure 7.1.

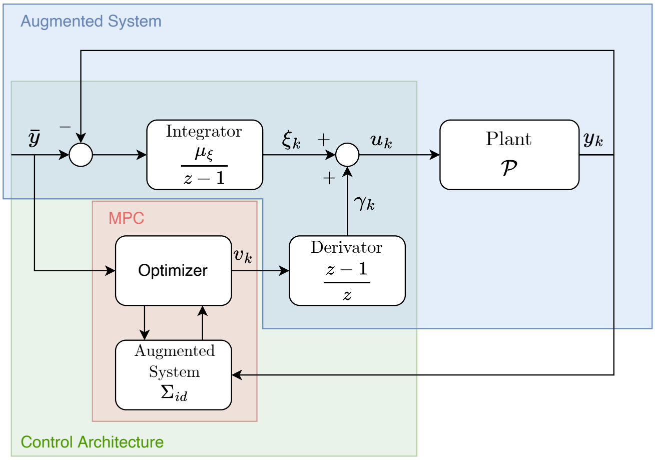

Figure 7.1: *Integral-Derivative NMPC architecture.*

As evident from the diagram, the control variable is given by the sum of two actions:

i. The integral action ξ_k, given by the integral of the output tracking error $e_k = \bar{y} - y_k$. The Internal Model Principle [141] guarantees that the presence of an integral action ensures, as long as the closed-loop stability is preserved, robust zero-error output regulation for constant reference signals and under parametric uncertainties in the system.

ii. The derivative action γ_k, given by the discrete-time derivative of the MPC output variable v_k. This allows to guarantee that, at steady state, the control variable associated to the MPC law is null, so that the system regulation relies entirely upon the integral action. This control action is therefore relegated to the improvement of closed-loop dynamic performance and, as detailed in the following, allows a simple definition of suitable terminal constraint of the FHOCP problem.

The control input u_k is therefore defined as

$$u_k = \xi_k + \gamma_k. \tag{7.2a}$$

The first term is the integral of the output tracking error, i.e.

$$\xi_{k+1} = \xi_k + \mu_\xi(\bar{y} - y_k), \tag{7.2b}$$

where μ_ξ is the integrator's gain, whose design is later discussed. The latter term, γ_k, is the discrete-time derivative of the variable v_k, i.e.

$$\begin{cases} \theta_{k+1} = v_k \\ \gamma_k = v_k - \theta_k \end{cases}. \tag{7.2c}$$

By combining (7.1) and (7.2), one obtains the augmented system $\Sigma_{id}(\Phi^\star)$, defined as

$$\Sigma_{id}(\Phi^\star):\begin{cases} x_{k+1}=f(x_k,u_k;\Phi^\star)\\ \xi_{k+1}=\xi_k+\mu_\xi(\bar{y}-Cx_k)\\ \theta_{k+1}=v_k\\ \gamma_k=v_k-\theta_k\\ u_k=\xi_k+\gamma_k\\ y_k=Cx_k\end{cases} \tag{7.3}$$

Defining the enlarged state χ_k as

$$\chi_k=[x_k',\xi_k',\theta_k']', \tag{7.4a}$$

the augmented system (7.3) can be compactly re-written as

$$\Sigma_{id}(\Phi^\star):\begin{cases}\chi_{k+1}=\varphi_{id}(\chi_k,v_k,\bar{y})\\ y_k=\psi_{id}(\chi_k)\end{cases}. \tag{7.4b}$$

The input definition (7.2a) is hereafter compactly denoted as

$$u_k=\pi_{id}(\chi_k,v_k). \tag{7.4c}$$

We henceforth denote by $\bar{\Sigma}_{id}=(\bar{\chi},\bar{v},\bar{y})$ the equilibrium of (7.4) corresponding to $\bar{\Sigma}$, which can be easily computed as

$$\bar{\chi}=[\bar{x}',\bar{u}',\bar{v}']', \tag{7.5}$$

where $\bar{v}$ is an arbitrary constant value.

Remark 7.1. *It can be easily shown that Corollary 6.1 still applies. That is, the model's exponential δISS entails that it is possible to design the integrator gain μ_ξ so as to preserve the local asymptotic stability of the augmented system (7.4) around the equilibrium $\bar{\Sigma}_{id}$. Therefore, it is assumed that μ_ξ is designed accordingly.*

At last, an MPC law is synthesized for the augmented system Σ_{id}. Letting N denote the prediction horizon and $\mathcal{N}=\{0,...,N-1\}$, we can formulate the predictive control law by adopting a standard FHOCP with zero-terminal constraint, stated as follows.

$$
\begin{aligned}
\min_{v_{k:k+N-1|k}} \quad & \sum_{\tau=0}^{N-1} \left(\left\| \chi_{k+\tau|k} - \bar{\chi} \right\|_Q^2 + \left\| u_{k+\tau|k} - \bar{u} \right\|_R^2 \right) && && \text{(7.6a)} \\
\text{s.t.} \quad & \chi_{k|k} = \chi_k && && \text{(7.6b)} \\
& \chi_{k+\tau+1|k} = \varphi_{id}(\chi_{k+\tau|k}, v_{k+\tau|k}, \bar{y}) && \forall \tau \in \mathcal{N} && \text{(7.6c)} \\
& u_{k+\tau|k} = \pi_{id}(\chi_{k+\tau|k}, v_k) && \forall \tau \in \mathcal{N} && \text{(7.6d)} \\
& u_{k+\tau|k} \in \tilde{\mathcal{U}} && \forall \tau \in \mathcal{N} && \text{(7.6e)} \\
& \chi_{k+N|k} = \bar{\chi} && && \text{(7.6f)}
\end{aligned}
$$

In the above-reported FHOCP, the state of the predictive model is initialized in the state χ_k via constraint (7.6b). It is worth noticing that χ_k is known, since its component are known, see (7.4a). The dynamics of the augmented model (7.4) are implemented by means of constraints (7.6c) and (7.6d), where this latter allows to explicitly compute the input applied to the plant. The input constraint satisfaction is enforced via constraint (7.6e), while in (7.6f) the terminal equality constraint is imposed.

The cost function adopted, reported in (7.6a), penalizes the deviation of the augmented state and of the overall control variable from their equilibrium. These terms are weighted by the positive definite matrices Q and R, respectively. More specifically, the state weight matrix is defined as $Q = \text{diag}(Q_x, Q_\xi, Q_\theta)$, where the blocks Q_x, Q_ξ, and Q_θ, are the positive definite weights associated to the respective components of χ. Note that the weight Q_θ can be chosen so that $Q_\theta \ll Q_x, Q_\xi$, since its sole purpose is to avoid unnecessarily large deviations of the derivator state θ_k from its equilibrium value $\bar{v}$. The input weight R, instead, penalizes the difference between the overall control action applied to the system and its equilibrium value $\bar{u}$.

The solution of the nonlinear FHOCP (7.6) yields the optimal augmented system's input sequence $v^\star_{k:k+N-1|k}$, and the corresponding optimal control sequence $u^\star_{k:k+N-1|k}$. In observance of the Receding Horizon principle, the first optimal control move is applied, i.e. $u_k = u^\star_{k|k}$, and at the successive time instant the entire procedure is repeated. This yields the implicit state-feedback MPC law $u_k = \kappa_{\text{MPC}}(\chi_k)$.

Remark 7.2. *The proposed MPC law, characterized by the FHOCP (7.6), is a standard MPC with zero-terminal constraint. Therefore, its recursive feasibility and closed-loop stability can be guaranteed in nominal conditions [116].*

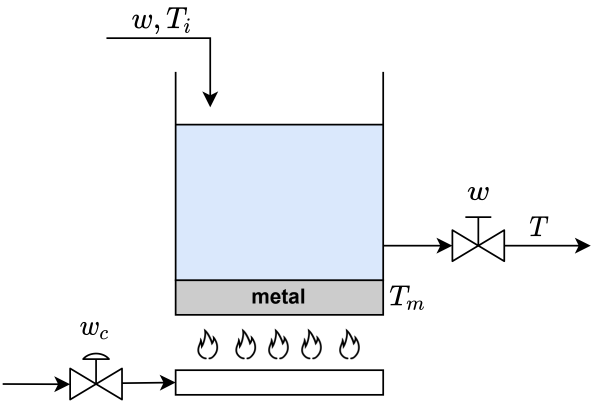

Figure 7.2: *Scheme of the water heating benchmark system.*

Remark 7.3. *As the proposed MPC law steers χ_k to $\bar{\chi}$, θ_k settles to its target equilibrium $\bar{v}$, which in turn implies that $\gamma_k \to 0$. Therefore, while γ_k allows to improve the dynamic closed-loop performances and to fulfill the input constraint (7.6e), at steady state the system is regulated by the integral action ξ_k only.*

7.2 Numerical example

7.2.1 Benchmark system description

The proposed control architecture has been tested on the water-heating benchmark system depicted in Figure 7.2. The objective of this system is to control the temperature of the water in a reservoir, so that users can be supplied with the desired water at the correct temperature. The water demand from the user is denoted by w, and represent an exogenous disturbance. For simplicity, it is assumed that the water flow rate at the inlet matches the demand, so that the level dynamics are neglected.

We indicate by T_i the temperature of the water at the inlet, and by T the temperature of the water served to the users. The water temperature is assumed to be uniform throughout the tank.

The water is heated through a metal plate placed under the tank, which is heated by means of a gas burner. More specifically, the metal plate is at a temperature T_m, and it heats the water inside the tank by conduction. The metal plate is, in turn, heated by the flames of a gas burner, whose gas flow rate is denoted by w_c.

Then, assuming the absence of heat losses, and that the flame heat is

Parameter	Description	Value	Units
A_t	Tank's cross-section	$\frac{\pi}{4}$	m^2
ρ_w	Water's density	997.8	$\frac{kg}{m^3}$
c_w	Water's specific heat	4180.0	$\frac{J}{kg \cdot K}$
M_m	Metal plate's mass	617.32	kg
c_m	Metal's specific heat	481.0	$\frac{J}{kg \cdot K}$
σ_r	Radiation coefficient	5.67×10^{-8}	$\frac{W}{m^2 \cdot K^4}$
k_{lm}	Heat exchange coefficient	3326.4	$\frac{kg}{s^3 \cdot K}$
T_f	Flame's temperature	1200	K
k_f	Heat exchange coefficient	8.0	$\frac{m^2 \cdot s}{kg}$
z_w	Water level	2.0	m
$\bar{w}$	Nominal water demand	1.0	$\frac{kg}{s}$
$\bar{T}_i$	Nominal inlet water temperature	298	K

Table 7.1: *Parameters of the water-heating benchmark system*

exchanged only via radiation, the following model of the system can be formulated:

$$\begin{cases} \dot{T} = \dfrac{1}{\rho_w A_t z_w}\Big[w(T_i - T) + \dfrac{k_{lm} A_t}{c_w}(T_m - T)\Big] \\ \dot{T}_m = \dfrac{1}{M_m c_m}\Big[-k_{lm} A_t (T_m - T) + \sigma_r k_f w_c (T_f^4 - T_m^4)\Big] \end{cases} \tag{7.7}$$

The parameters of the plant model are reported in Table 7.1. The model is characterized by one controllable input $u = [w_c]$, one output $y = [T]$, and two states $x_p = [T, T_m]'$. Two disturbances, whose nominal values are reported in Table 7.1, also affect the plant, i.e. $d_p = [w, T_i]'$. The gas flow rate w_c is subject to saturation,

$$w_c \in [0.05, 0.18]. \tag{7.8}$$

Finally, a simulator of the described benchmark system has been implemented in Simulink, so as to collect the training data and to test the proposed control architecture.

7.2.2 System identification

To identify the benchmark system described above by means of the training procedure described in Chapter 4, the required datasets have been collected from the simulator of the plant (7.7). Therefore, the plant simulator has been excited with a multilevel pseudo-random signal spanning the input space (7.8). The input-output sequences have been collected with the sampling time $\tau_s = 120\, s$.

Overall, the length of the collected sequences is $T_{\text{tr}} = 2500$ for the training data, $T_{\text{val}} = 1200$ for the validation data, and $T_{\text{te}} = 400$ for the independent test data. Then, according to the TBPTT approach, the training data has been randomly split in $N_{\text{tr}} = 120$ subsequences of length $T_s = 400$, while the validation data has been split in $N_{\text{val}} = 30$ subsequences having the same length T_s. We recall that the input-output data needs to be normalized, which has been done according to Remark 4.1. Nonetheless, in the figures to follow denormalized input and output trajectories are represented for the sake of interpretability.

The RNN architecture here considered is a NNARX, see Section 3.1, with a single-layer ($L = 1$) FFNN as nonlinear regression function, characterized by 30 neurons. The input-output regression horizon is $H = 5$. The training procedure has been carried out using Algorithm 1 implemented in PyTorch 1.10, adopting a batch size equal to 5 and the piecewise-linear regularization term (4.9) to enforce the model's δISS, via Theorem 3.2. Overall, the training procedure took 1288 epochs.

The trained NNARX model, which is exponentially δISS ($\nu(\Phi^\star) = -0.06 < 0$), has been tested on the independent test set, whose input sequence is depicted in Figure 7.3. In Figure 7.4, the corresponding NNARX's open-loop prediction is compared to the ground truth, witnessing fair modeling performances. The FIT index scored, computed as in (4.6), is 92.8%.

7.2.3 Control design

The proposed control architecture has been implemented, with the goal of tracking piecewise-constant water temperature references, and to asymptotically reject possible (unmeasured) disturbances associated to variations of the water demand w or of the water temperature at the inlet T_i. In particular, the reference signal for the water temperature is depicted in Figure 7.5, while the disturbances the realizations illustrated in Figure 7.6 have been considered. Notice that these trajectories allow to assess the nominal offset-free tracking capabilities, as in the first half of the closed-loop experiment the disturbances match their nominal values, and to test the robustness against the disturbances injected in the second half of the experiment.

We point out that, being the output reference piecewise constant, at every change of the setpoint $\bar{y}$ the nominal equilibrium triplet $\bar{\Sigma} = (\bar{x}, \bar{u}, \bar{y})$ needs to be computed via (5.2), and the corresponding nominal equilibrium triplet of the augmented system $\bar{\Sigma}_{id} = (\bar{\chi}, \bar{v}, \bar{y})$ needs to be defined

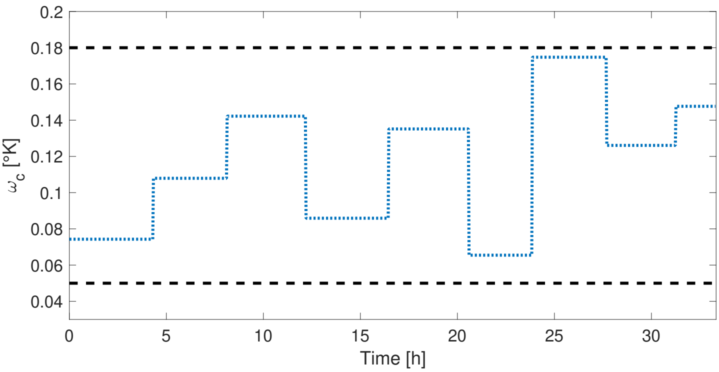

Figure 7.3: *Input sequence of the water heating system's test dataset.*

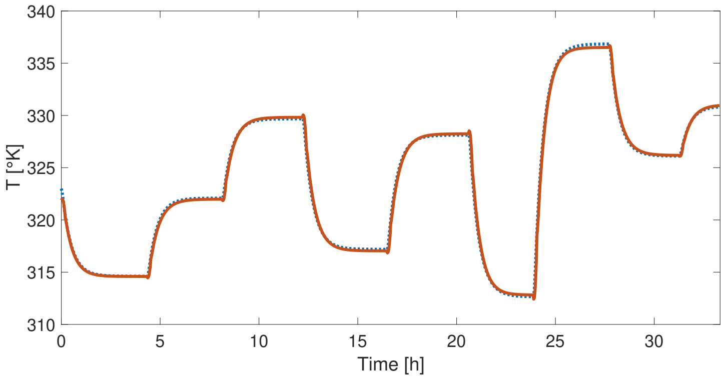

Figure 7.4: *Modeling performances of the trained NNARX model, tested on the independent test dataset. The NNARX open-loop simulation (red continuous line) is compared to the ground truth (blue line).*

via (7.5). Moreover, if the new setpoint is not in the neighborhood of the previous equilibrium, Assumption 5.2 should be verified, and the integrator gain μ_ξ might need to be recomputed if the local stability around the new equilibrium is not preserved, see Remark 7.1.

The prediction horizon of the FHOCP (7.6) has been chosen as $N = 50$, while the weights of the cost function has been defined as $R = 0.1$ and $Q = \text{diag}(Q_x, R, Q_\theta)$, with $Q_\theta = 10^{-5}$. The submatrix Q_x penalizes the displacement of the predicted model's state from the equilibrium. Owing to the particular structure of the NNARX state vector, and recalling that $H = 5$, we select $Q_x = \text{diag}(Q_z, Q_z, Q_z, Q_z, Q_z)$, where $Q_z = \text{diag}(10, 0.1)$.

The integrator gain μ_ξ has been selected in accordance to Remark 7.1. Specifically, the NNARX model has been linearized around the nominal operating condition, corresponding to $\bar{T} = 323°K$. For such equilibrium,

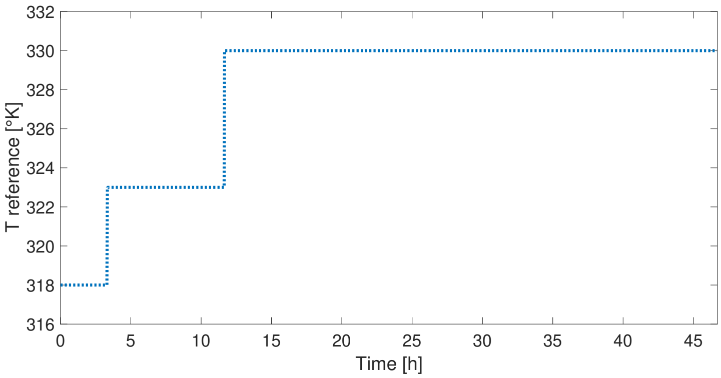

Figure 7.5: *Piecewise-constant output reference signal considered for assessment of the closed-loop performances.*

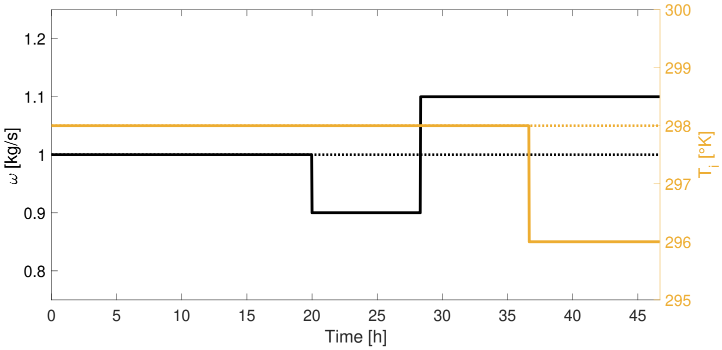

Figure 7.6: *Realizations of the disturbances affecting the system, i.e. w (black solid line) and T_i (yellow solid line), compared to their nominal values (dotted lines).*

the value $\check{\kappa}_\mu = 0.27$ has been numerically computed, which ensures the local asymptotic stability of augmented model's linearization. In accordance to Corollary 6.1, the choice of $\kappa_\mu = 0.145 \in (0, 0.27)$ has led to the integrator gain $\mu_\xi = 0.14$. Such integrator gain has been verified to preserve the local stability also around the other equilibria of interest, i.e., those corresponding to $\bar{T} = 318°K$ and to $\bar{T} = 330°K$.

Baseline control architecture

The popular offset-free MPC strategy proposed in [118] has also been implemented as a baseline against which to compare the performance achieved by the control architecture here proposed. This control strategy, henceforth named Disturbance Estimation Based NMPC (DEB-NMPC), requires to augment the NNARX model with a fictitious matched disturbance acting on

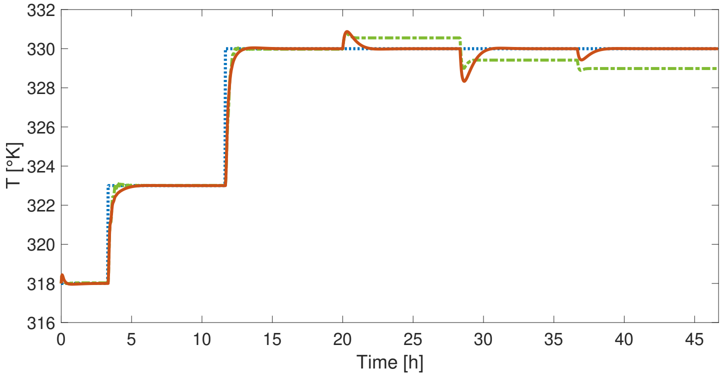

Figure 7.7: *Comparison between the closed-loop output trajectory achieved by the proposed control architecture (red solid line) and by the DEB-NMPC (green dashed-dotted line). The output reference has also been reported (blue dotted line).*

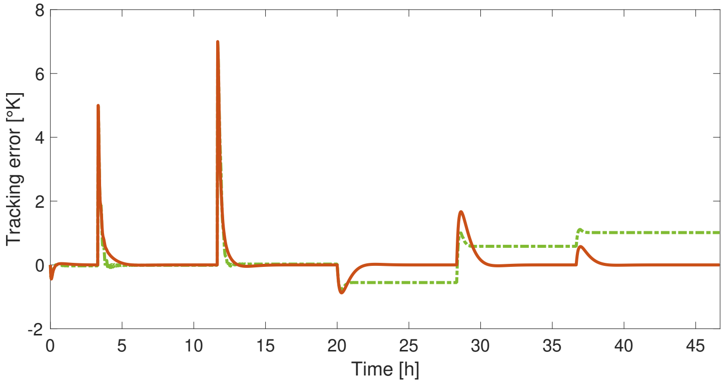

Figure 7.8: *Output tracking error achieved by the proposed control architecture (red solid line) versus that of the DEB-NMPC (green dashed-dotted line).*

the input variable. Such disturbance, customarily assumed to be constant, is then estimated by means of a moving horizon estimator [149]. Finally, a standard state-feedback NMPC law is designed to stabilize the enlarged system, consisting of the NNARX system model and the (constant) disturbance model, featuring a prediction horizon $N = 50$ and weights in line with those adopted for the proposed control architecture.

7.2.4 Closed-loop results

The closed-loop output tracking performances achieved by the proposed approach are compared to those of DEB-NMPC in Figure 7.7, while in Figure 7.8 the output tracking error is depicted. It is apparent that, while initially the DEB-NMPC scheme is able to compensate the plant-model

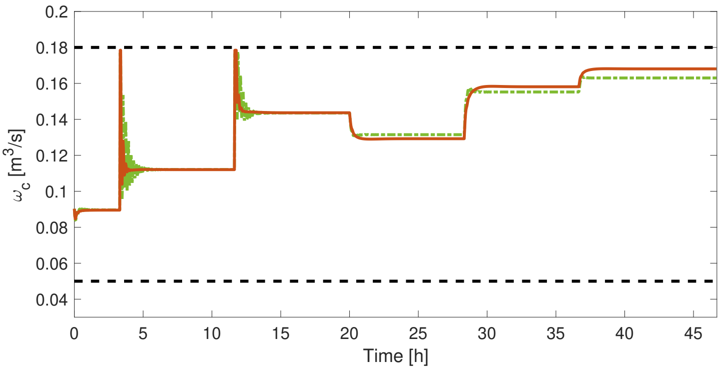

Figure 7.9: *Control action requested by the proposed control architecture (red solid line) compared to that of the DEB-NMPC (green dashed-dotted line). The black dashed lines correspond to the input saturation values.*

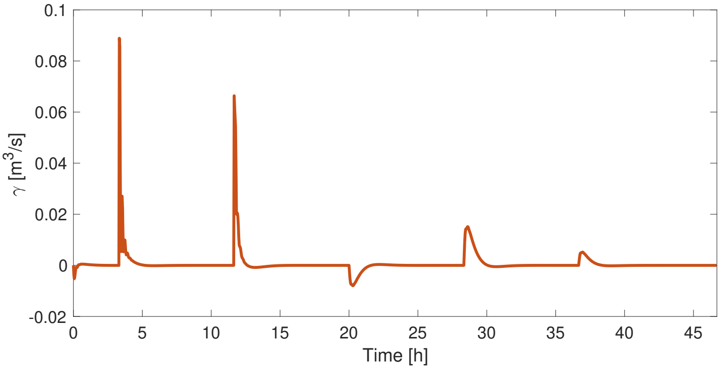

Figure 7.10: *Component γ_k of the closed-loop control variable.*

mismatch thanks to a reliable estimate of the fictitious matched disturbance, after the instant $t = 20$h – when disturbances occur, see again Figure 7.6 – the output tracking performances of such control scheme are lost. These unsatisfactory static performances are likely due to the inability of the moving horizon estimator to produce a suitable estimate of the disturbance. In contrast, the proposed control architecture is able to attain zero tracking error, even in the presence of the severe disturbances that affect the system.

In Figure 7.9, the control action requested by the two scheme is compared. In both schemes the input constraint (7.8) is fulfilled, although it can be observed that the control action issued by DEB-MPC is less moderate, mainly due to the transients of the disturbance estimator. It should be noted that in the DEB-NMPC approach, the choice of the fictitious disturbance model and its correct estimation are paramount to obtain satisfactory closed-loop performances. Recent works proposing alternative disturbance estimation-based strategies, see [130], will thus be object of future investigations.

Finally, in Figure 7.10 the evolution of the component γ_k of the control variable has been reported. From the figure it is apparent that, as expected, this component is used exclusively during transients to improve the closed-loop dynamic performance, while at steady-state the control variable relies entirely on the integral component ξ_k to fulfill the static performance requirement of asymptotic zero-error output regulation.

7.3 Summary

In this chapter, an NMPC architecture guaranteeing offset-free tracking of constant reference signals was formulated for NNARX models. This architecture relies on the idea of enlarging the system model with the integrator of the output tracking error, which entails asymptotic zero-error output regulation, and with a derivative action, which allows to improve the closed-loop dynamic performances while not affecting the static ones. The proposed control scheme attains nominal closed-loop stability and asymptotic offset-free tracking of constant references. The control law was tested on a water heating benchmark system, demonstrating satisfactory closed-loop performance and a good degree of robustness to the disturbances affecting the plant.

CHAPTER 8

Internal Model Control design

In this chapter, a control architecture based on the internal model control approach is described, with the main objective of providing a computationally less onerous alternative to the predictive control laws presented in previous chapters. As described in Chapter 5, at this stage it is assumed that an accurate RNN model of the system is available. The IMC approach boils down to constructing a controller ideally coincident with the model's inverse or, at least, a suitably accurate approximation of it. Fed with the output reference signal, the controller should thus generate a control sequence that drives the model's output as close as possible to the reference.

In view of their capabilities in approximating dynamical systems, we here propose to learn the controller by means of a gated RNN, such as LSTMs or GRUs, see Section 3. This idea is inspired by [15, 137] where, in the context of SISO systems, static FFNNs in autoregressive configurations have been used to learn the system model and an approximation of its inverse. With respect to the existing methods, the proposed approach yields the following advantages:

i. Gated RNNs represent better candidates for learning dynamical systems (such as the model and its inverse), owing to their long-term mem-

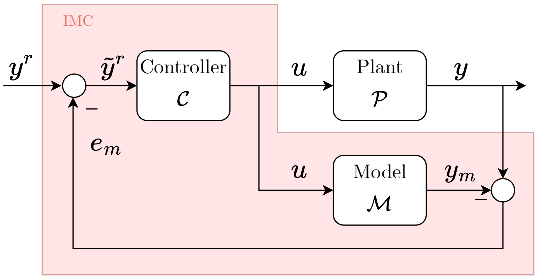

Figure 8.1: *General scheme of IMC.*

ory. In contrast, with feed-forward auto-regressive architectures such memory is enforced by supplying the past input-output data-points as inputs of the network [137], typically resulting in less accurate long-term learning compared to gated RNNs [57].

ii. The controller is learned by an inherently strictly-proper gated RNN, as opposed to feed-forward architectures, for which one needs to deal with the issue of the controller's improperness in the controller design phase [137], e.g. in presence of time delays.

iii. Owing to the model's and controller's proposed learning procedures, the proposed scheme can easily handle MIMO systems, and allows to account for input saturation.

The clear advantages of using gated RNNs come, however, at the cost of losing the guarantees of existence of the exact model's inverse. Nonetheless, we show that such inverse can be accurately approximated, and that the closed-loop stability and satisfactory performances can still be achieved.

The stability properties of the control scheme are also discussed. In particular, we show how the δISS of the model and the controller allows to provide stability and performance properties more solid than those available in the literature for generic nonlinear systems [15, 133].

8.1 Internal model control architecture

To best explain the IMC architecture, let us discuss the control scheme depicted in Figure 8.1, which represents the core of the IMC approach. Additional ingredients will be later introduced in the scheme.

At its core, the IMC scheme features three blocks: the unknown plant $\mathcal{P}$ that needs to be controlled, the model $\mathcal{M}$ of such plant, and the controller $\mathcal{C}$, which generates the control action to be applied to the two previous blocks. The model and the controller are referred to as IMC *ingredients*.

The plant's $\mathcal{M}$ can be, in general, derived from the first-principle equations or identified from the data. Consistently with the control problem discussed in Chapter 5, we here assume that it is described by an exponentially δISS RNN. With a slight abuse of notation, in order to improve the interpretability of the description to follow, let us recast model (5.1) as

$$\mathcal{M} = \Sigma_m(\Phi_m^\star) : \begin{cases} x_{m,k+1} = f_m(x_{m,k}, u_k; \Phi_m^\star) \\ y_{m,k} = g_m(x_{m,k}; \Phi_m^\star) \end{cases}, \tag{8.1}$$

where $x_{m,k} \in \mathbb{R}^{n_{m,x}}$ denotes the model's state, $y_{m,k}$ its output, and $\Phi_m^\star$ its trained weights.

The plant $\mathcal{P}$ is an unknown state-space dynamical system, characterized by the input vector u_k and the output vector y_k. Such plant is assumed to be approximated (ideally, very accurately) by the model from an input-output perspective. More formally, this means that given the input sequence $u_{0:k}$ applied to the plant and the corresponding measured output $y_{0:k}$, there exists an initial state of the model $x_{m,0}$ such that $\Sigma_m(\Phi_m^\star)$ replicates the measured output sequence exactly, i.e., $y_{m,0:k}(x_{m,0}, u_{0:k}; \Phi_m^\star) = y_{0:k}$. In such case, the plant-model mismatch feedback, defined as

$$e_{m,k} = y_k - y_{m,k}, \tag{8.2}$$

is constantly null, meaning that the feedback loop is cut and the controller operates in open-loop.

According to the IMC paradigm, the controller block $\mathcal{C}$ is a dynamical system approximating the inverse of the model (8.1). Here we propose to implement such controller via a gated RNN, i.e.

$$\mathcal{C} = \Sigma_c(\Phi_c) : \begin{cases} x_{c,k+1} = f_c(x_{c,k}, \tilde{y}_k^r; \Phi_c) \\ u_k = \phi(g_c(x_{c,k}; \Phi_c)) \end{cases}, \tag{8.3}$$

where $x_{c,k} \in \mathbb{R}^{n_{c,x}}$ denotes the controller state, the reference $\tilde{y}_k^r \in \mathbb{R}^{n_{c,u}}$ its input vector, and the control action u_k its output vector. The functions f_c and g_c of the controller depend on the chosen RNN architecture. For example, if $\Sigma_c(\Phi_c)$ is described by a deep GRU, f_c is described by (3.36a) and g_c by (3.36e). Note that the output transformation of (8.3) includes a $\tanh$ activation function, which allows to ensure that $u_k \in \mathcal{U}$, see [62][1].

[1] Note that the controller architecture (8.3) does not straightforwardly allow to consider tightened input constraints $u_k \in \tilde{\mathcal{U}} \subset \mathcal{U}$. By suitably designing the output transformation of (8.3) one may constrain the generated control action in an orthotopical tightened input set, but such set needs to be known at the training stage and time-invariant.

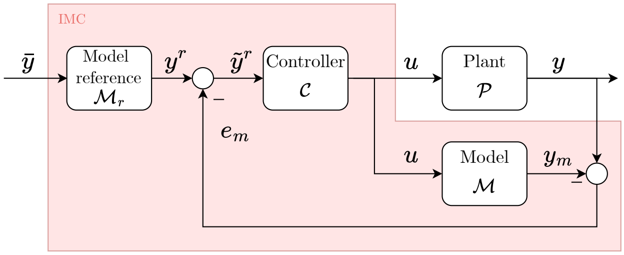

Figure 8.2: *General scheme of IMC with the model reference block.*

The training procedure to tune the controller's weights Φ_c is described later this chapter. From Figure 8.1, the term $\tilde{y}_k^r$ is defined as

$$\tilde{y}_k^r = y_k^r - e_{m,k}. \tag{8.4}$$

In the controller design phase, in view of the certainty equivalence principle, one assumes the modeling error feedback (8.2) null, i.e. $\tilde{y}_k^r = y_k^r$. Ideally, the controller is designed so that, for any output reference signal y_k^r, and for any initial state of the model, there exists a controller initialization $x_{c,0}$ such that the generated control sequence steers the model's output to the reference. Precisely, this condition implies that for any $y_{0:k}^r$ and $x_{m,0}$ there exists $x_{c,0}$ such that, letting $u_{0:k}(x_{c,0}, y_{0:k}^r; \Phi_c)$ be the control sequence generated by the controller (initialized in $x_{c,0}$ and fed with the output reference $\tilde{y}_k^r = y_{0:k}^r$), it holds that

$$y_{m,0:k}\big(x_{m,0}, u_{0:k}(x_{c,0}, y_{0:k}^r; \Phi_c); \Phi_m^\star\big) = y_{0:k}^r.$$

In practice, however, the controller is synthesized as a suitable approximation of the model's inverse, since the exact inverse may be not proper, not analytically defined, or even not stable [134].

In the remainder, we consider the modified control scheme shown in Figure 8.2, where a model reference block $\mathcal{M}_r$ has been added [62, 137]. The model reference encodes the desired closed-loop response to the actual reference signal[2] $\bar{y}_k$. Indeed, in the case of perfect control (i.e., $y_k = y_k^r$), the relationship between the reference signal $\bar{y}_k$ and the system output y_k is exactly $\mathcal{M}_r$.

Remark 8.1. *The above-described IMC synthesis procedure takes place offline. Albeit training a RNN is a computationally-intensive task, it can be performed on a sufficiently powerful computational platform and then deployed to an embedded controller with limited computational resources. Indeed, during online system operations the IMC control schemes boils down*

[2] The output reference is here denoted as $\bar{y}_k$ for consistency to the notation adopted in Chapter 5.

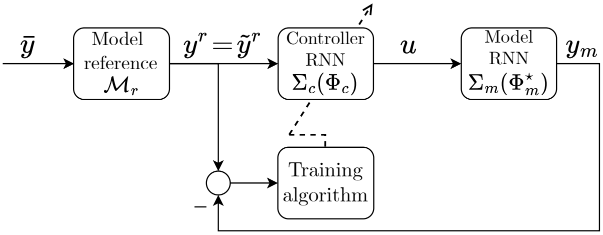
$\bar{y}$
Model reference $\mathcal{M}_r$
$y^r = \tilde{y}^r$
Controller RNN $\Sigma_c(\Phi_c)$
u
Model RNN $\Sigma_m(\Phi_m^\star)$
y_m
Training algorithm
$-$

Figure 8.3: *Scheme of the controller's learning procedure.*

to the propagation of the model $\mathcal{M}$ and of the controller $\mathcal{C}$ based on the measured output and the filtered reference signal which, for RNN architectures, corresponds to a sequence of tensor operations that can be efficiently carried out even in a real-time environment.

We are now in the position to describe the training procedure adopted to learn the controller (8.3).

8.1.1 Controller learning

Under the certainty equivalence principle, i.e., $\tilde{y}_k^r = y_k^r$, the controller is trained to match the right-inverse of the model, informally denoted by $\mathcal{M}^{-1}$. While ideally one would retrieve an analytical expression of the inverse of the gated RNN model, due to the complexity and nonlinearity of $\Sigma_m(\Phi_m^\star)$, finding such inverse is not possible in general, as its existence is not even guaranteed.

Instead, along the lines of [15, 137], the controller $\Sigma_c(\Phi_c)$ is trained to approximate the inverse of the model by means of the learning scheme depicted in Figure 8.3. The rationale of this scheme is to tune the controller weights Φ_c so that $\Sigma_c(\Phi_c)$ generates a control action that steers the output $y_{m,k}$ of the model $\Sigma_m(\Phi_m^\star)$ as close as possible to output reference generated by the model reference block, i.e. y_k^r. Note that, in this context, the model's weights $\Phi_m^\star$ are fixed, since it has been already identified.

To describe the controller learning procedure, we first discuss what data is necessary and how it should be generated, after which the training algorithm is described.

Controller training data

The dataset used for the controller's learning procedure consists of a set of reference signals that the controller learns to track, and it is generated synthetically based on the reference signals of interest. The closer these refer-

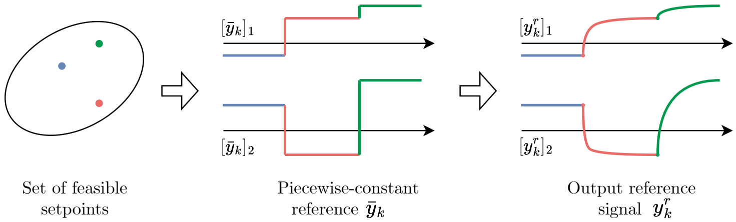
$[\bar{y}_k]_1$
$[\bar{y}_k]_2$
$[y^r_k]_1$
$[y^r_k]_2$
Set of feasible setpoints
Piecewise-constant reference $\bar{y}_k$
Output reference signal y^r_k

Figure 8.4: *Sketch of the reference signals' generation: first, a piecewise-constant output reference characterized by feasible setpoints is generated; then, such signal is filtered by the model reference $\mathcal{M}_r$.*

ences are to those imposed in closed-loop system operations, the more accurate the learned controller action will be. In this thesis, since the closed-loop is expected to be operated with constant reference signals, the dataset is generated as a collection of piece-wise constant references $\bar{y}_k$, which are then filtered with the selected model reference $\mathcal{M}_r$ to obtain the output reference y^r_k.

These setpoints, however, can not be generated in a completely random fashion. First, being the reference signal the input of $\Sigma_c(\Phi_c)$, it is assumed to fulfill Assumption 3.1[3]. Secondly, these references need to be *feasible* for the model. This means that, at least asymptotically, the difference between the filtered reference y^r_k and the model output $y_{m,k}$ can be made sufficiently small by means a suitable control sequence. Letting $\bar{\bar{y}}$ denote a a setpoint of the piecewise-constant reference $\bar{y}_k$, and by $\bar{y}^r$ the corresponding steady-state value[4] of y^r_k, the feasibility condition boils down to requiring that Assumption 5.1 is satisfied for $\bar{y}^r$. That is, the model $\Sigma_m(\Phi^\star_m)$ admits $(\bar{x}_m, \bar{u}, \bar{y}^r)$ as an equilibrium for some $\bar{u} \in \mathcal{U}$ and $\bar{x}_m \in \mathcal{X}_m$. This issue is further discussed in the numerical example given in Section 8.2. In Figure 8.4 the extraction of a random feasible output reference is sketched.

Therefore, the generation of the controller training set amounts to synthetically generating a sufficiently large number of reference signals. We denote each one of these references as

$$y^{r,\{i\}}_{0:T_{c,s}} \tag{8.5a}$$

where $i \in \mathcal{I}_{c,\text{tr}} = \{1, ..., N_{c,\text{tr}}\}$ represents the index of the sequence, $N_{c,\text{tr}}$ the number of training sequence, and $T_{c,s}$ the fixed length of each sequence.

[3] This assumption is not restrictive. Indeed, the model $\Sigma_m(\Phi^\star_m)$ itself is trained with output sequences whose subunitarity is guaranteed by normalization, see Remark 4.1. My means of the same normalization, the output reference can be guaranteed to have $\|y^r_k\|_\infty \leq 1$.

[4] In general, the model reference $\mathcal{M}_r$ is designed to have a unitary static gain [137], i.e., $\bar{\bar{y}} = \bar{y}^r$.

The validation sequences are defined as in (8.5a), but they are indexed by $i \in \mathcal{I}_{c,\text{val}} = \{N_{c,\text{tr}} + 1, ..., N_{c,\text{tr}} + N_{c,\text{val}}\}$. Similarly, the output reference trajectory used for testing is denoted as

$$y^r_{\text{te},0:T_{c,\text{te}}}, \tag{8.5b}$$

where $T_{c,\text{te}}$ denotes the length of the sequence.

Controller's training algorithm

The controller training procedure is performed along the lines of Algorithm 1. At each training epoch, the training set $\mathcal{I}_{c,\text{tr}}$ is partitioned in B batches, denoted as $\mathcal{I}^{\{1\}}_{c,\text{tr}}$, ..., $\mathcal{I}^{\{B\}}_{c,\text{tr}}$, sharing the same cardinality, which is the batch size. The following loss function is then minimized for each batch $b \in \{1, ..., B\}$

$$\mathcal{L}_c(\Phi_c) = \text{MSE}(\mathcal{I}^{\{b\}}_{c,\text{tr}}; \Phi_c) + \rho(\nu(\Phi_c)). \tag{8.6}$$

The first term of (8.6) measures the controller's performances on the batch $\mathcal{I}^{\{b\}}_{c,\text{tr}}$. Such performance index measures is defined as

$$\text{MSE}(\mathcal{I}_c; \Phi_c) = \frac{1}{|\mathcal{I}_c|} \sum_{i \in \mathcal{I}_c} \text{MSE}\left(y^{\{i\}}_{m,0:T_{c,s}}(\Phi_c), y^{r,\{i\}}_{0:T_{c,s}}\right), \tag{8.7a}$$

where, with a slight abuse of notation, we indicate by $y^{\{i\}}_{m,0:T_{c,s}}(\Phi_c)$ the output of the model $\Sigma_m(\Phi^\star_m)$ when it is controlled by $\Sigma_c(\Phi_c)$ to track the reference $y^{r,\{i\}}_{0:T_{c,s}}$, i.e,

$$y^{\{i\}}_{m,0:T_{c,s}}(\Phi_c) = y_{m,0:T_{c,s}}\left(x_{m,0}, u_{0:k}(x_{c,0}, y^{r,\{i\}}_{0:T_{c,s}}; \Phi_c); \Phi^\star_m\right). \tag{8.7b}$$

Note that, as discussed in Section 4, the initial states of the networks, i.e. $x_{m,0}$ and $x_{c,0}$, are randomly drawn from their invariant sets, denoted as $\mathcal{X}_m$ and $\mathcal{X}_c$, respectively. The MSE introduced in (8.7a) is hence defined analogously to (4.7), with a suitable washout period to accomodate for the initialization transient, and it measures the mean square output tracking error of the model under the control sequence generated by $\Sigma_c(\Phi_c)$.

The second term is instead a δISS regularization term, designed according to the prescriptions given in Section 4.2.2, which enforces the exponential δISS of the controller (8.3).

The gradient of $\mathcal{L}_c(\Phi_c)$ with respect to Φ_c is then computed and backpropagated via gradient descent methods (e.g., Adam or RMSProp). The training procedure yields the controller's weights $\Phi^\star_c$. In Algorithm 2 the resulting training algorithm is reported.

Algorithm 2 Controller training algorithm

```
Require: Training references $\mathcal{I}_{c,\text{tr}}$ and validation references $\mathcal{I}_{c,\text{val}}$
  Initialize the weights $\Phi_c$
  for epoch $e = 1, ..., E$ do
      Randomly partition $\mathcal{I}_{c,\text{tr}}$ into batches $\mathcal{I}_{c,\text{tr}}^{\{1\}}, ..., \mathcal{I}_{c,\text{tr}}^{\{B\}}$
      for batch $b = 1, ..., B$ do
          Compute the loss $\mathcal{L}_c(\mathcal{I}_{c,\text{tr}}^{\{b\}}, \Phi_c)$ using random initial states      ▷ Forward pass
          Compute the batch gradient $\nabla_{\Phi_c}\mathcal{L}_c(\mathcal{I}_{c,\text{tr}}^{\{b\}}, \Phi_c)$      ▷ Gradient computation
          Update $\Phi_c$ using gradient descent algorithms      ▷ Backward pass
      end for
      Compute the validation metrics $\text{MSE}(\mathcal{I}_{c,\text{val}}; \Phi_c)$
      if $\nu(\Phi_c) < 0$ and the validation metrics stops improving then
          Stop the training procedure
      end if
  end for
  Assess the performances on the test reference $y^r_{0:T_{c,\text{te}}}$
```

8.1.2 Stability properties

In order to describe the stability properties attained by the proposed control architecture, let us introduce some notion required to the purpose.

First, we point out that a δISS system with a Lipschitz-continuous output transformation is also Incremental Input-to-Output Stable (δIOS), i.e., it admits functions $\beta_y \in \mathcal{KL}$ and $\gamma_y \in \mathcal{K}_\infty$ such that

$$\|y_{a,k} - y_{b,k}\|_p \leq \beta_y(\|x_{a,0} - x_{b,0}\|_p, k) + \gamma_y(\|u_{a,0:k} - u_{b,0:k}\|_{p,\infty}). \tag{8.8}$$

In turn, under minor assumptions[5], this property implies the Input-Output Practical Stability (IOPS). This property, which is a generalization of the $\mathcal{L}_p$-gain stability [150], implies the existence of some finite $\varrho_y > 0$ and $\gamma_y \in \mathcal{K}_\infty$ such that

$$\|y_k\|_p \leq \gamma_y(\|u_{0:k}\|_{p,\infty}) + \varrho_y. \tag{8.9}$$

It thus entails that bounded inputs lead to bounded output trajectories.

Ideal closed-loop properties

We are now in the position to state the closed-loop properties commonly claimed in the IMC literature, see [15, 133].

Property 8.1 (Ideal stability). *If the model is exact and both the model and the controller are IOPS, the closed-loop is IOPS.*

[5] Specifically, it is necessary to assume that the equilibrium manifold of the system is not empty.

According to this property, in absence of plant-model mismatch and output noise, the modeling error feedback is bounded, and the controller operates in open-loop. Therefore, the IOPS of the both the controller and of the plant implies the IOPS of their cascade.

Property 8.2 (Ideal perfect control)**.** *Assume that the model is exact and IOPS, and that the controller matches the model's inverse, i.e.* $\mathcal{C} = \mathcal{M}^{-1}$*. Then, if the controller is IOPS, the closed-loop matches the model reference* $\mathcal{M}_r$*.*

Under the assumption of exactness of the model, and upon its suitable initialization, the control system operates in open-loop with a constantly null modeling error feedback. Since $\mathcal{C} = \mathcal{M}^{-1} = \mathcal{P}^{-1}$, there exists an initial state of the controller such that, at any time instant, $y_k = y_k^r$. The relationship between $\bar{y}_k$ and y_k is therefore the model reference $\mathcal{M}_r$ itself.

Property 8.3 (Ideal zero offset)**.** *Assume that the model is exact and IOPS, and that the steady state control action generated by the controller matches the steady-state value of the model's inverse. Then, if the controller is IOPS, offset-free tracking is asymptotically attained.*

The exactness of the model implies that, upon a suitable model initialization, the control system operates in open-loop with null feedback $e_{m,k}$. If, at least at steady state, the IOPS controller $\mathcal{C}$ matches the model's inverse, there exists a suitable initialization of the controller such that

$$y_k(x_{p,0}, u_{0:k}(x_{c,0}, y_{0:k}^r; \Phi_c^\star)) \xrightarrow[k\to\infty]{} y_k^r.$$

That is, asymptotic offset-free control is achieved.

These properties are called "ideal" because they require not only exactness of the model and the controller, but also correct initialization of these IMC ingredients. In practice, however, these assumptions are difficult to guarantee. Therefore, in the following we show how the model's and controller's δISS helps alleviating the restrictiveness of such premises.

Practical closed-loop properties

Along the lines the properties reported above, let us assume that the model is exact, i.e., that the plant can be described by the equations of model (8.1)

$$\mathcal{P}: \begin{cases} x_{p,k+1} = f_m(x_{p,k}, u_k; \Phi_m^\star) \\ y_k = g_m(x_{p,k}; \Phi_m^\star) \end{cases}. \tag{8.10}$$

We point out that, since $\Sigma_m(\Phi_m^\star)$ is exponentially δISS and δIOS, the plant (8.10) enjoys the same properties. Therefore, since

$$e_{m,k} = y_k(x_{p,0}, u_{0:k}; \Phi_m^\star) - y_{m,k}(x_{m,0}, u_{0:k}; \Phi_m^\star), \tag{8.11}$$

this allows to state that, for any $u_{0:k} \in \mathcal{U}_{0:k}$, any $x_{p,0} \in \mathcal{X}_m$, and any $x_{m,0} \in \mathcal{X}_m$,

$$\|e_{m,k}\|_p \leq \beta_y(\|x_{p,0} - x_{m,0}\|_p, k). \tag{8.12}$$

Under the model's exactness assumption, the modeling error feedback exponentially thus converges to zero even when the initial conditions of the models are wrongly set.

Property 8.4 (Practical closed-loop stability)**.** *If the model is initialized in a sufficiently small neighborhood of the plant's state, the closed-loop IMC scheme (Figure 8.2) is IOPS with respect to the reference trajectory y^r.*

This property is motivated by the fact that, in view of (8.12), if $x_{m,0}$ is sufficiently close to $x_{p,0}$, the modeling error preserves the unity-boundedness of $\tilde{y}_k^r$, i.e., the controller is operated in the input set with respect to which it is guaranteed to be δISS and IOPS. Since $e_{m,k}$ exponentially converges to zero, the control scheme is asymptotically open-loop. In such configuration, the cascade between two IOPS systems, i.e., the controller and the plant, ensures the IOPS of the scheme with respect to the reference signal[6].

Property 8.5 (Practical perfect control)**.** *Assume that*

i. the model is initialized in a sufficiently small neighborhood of the plant's state;

ii. there exists a possibly unknown initial state of the controller, denoted by $x_{c,0}^*$, such that the generated control action matches the model's inverse, i.e., for any $k \in \mathbb{Z}_{\geq 0}$ it holds

$$y_k(x_{0,p}, u_{0:k}(x_{c,0}^*, y^r; \Phi_c^\star); \Phi_m^\star) = y_k^r. \tag{8.13}$$

Then closed-loop asymptotically matches the model reference $\mathcal{M}_r$.

As discussed above, regardless of the applied control action, if the model is exact and δISS, the modeling error feedback asymptotically converges to zero, and the control scheme is asymptotically open-loop, and $\tilde{y}_k^r \to y_k^r$. Moreover, if the model is initialized sufficiently close to the plant's

[6] In addition, as discussed in Chapter 3, for the proposed RNN architectures the devised δISS sufficient conditions also imply their ISPS. Therefore, not only the closed-loop is IOPS, but also ISPS.

state, the unity boundedness of $\tilde{y}_k^r$ is preserved, which guarantees that the controller is δISS and δIOS.

Therefore, if there exists such a $x_{c,0}^*$, the generated control sequence is asymptotically independent of $x_{c,0}$ and converges to $u_k(x_{c,0}^*, y_{0:k}^r; \Phi_c^\star)$, i.e.

$$u_k(x_{c,0}, \tilde{y}^r; \Phi_c^\star); \to u_k(x_{c,0}^*, y^r; \Phi_c^\star).$$

Then, in view of the plant's δIOS,

$$y_k(x_{0,p}, u_{0:k}(x_{c,0}, \tilde{y}^r; \Phi_c^\star); \Phi_m^\star) \to y_k(x_{0,p}, u_{0:k}(x_{c,0}^*, y^r; \Phi_c^\star); \Phi_m^\star).$$

That is, the control scheme asymptotically achieves perfect control.

Property 8.6 (Practical zero offset)**.** *Assume that*

i. the model is initialized in a sufficiently small neighborhood of the plant's state;

ii. for any output setpoint $\bar{y}^r$, the unique steady-steady state control action $\bar{u}(\bar{y}^r)$ is such that the model admits $(\bar{x}_m, \bar{u}(\bar{y}^r), \bar{y}^r)$ as equilibrium.

Then, asymptotic zero-error output tracking is achieved.

Under the exact model assumption, as discussed above, the control scheme is asymptotically open-loop, and $\tilde{y}_k^r \to y_k^r$. Moreover, for model initial states sufficiently close to the plant's states, the unity-boundedness of $\tilde{y}_k^r$ is preserved. Let us note that, owing to the controller δISS, for any constant output setpoint $\bar{y}^r$ the controller admits a unique equilibrium $(\bar{x}_c, \bar{y}^r, \bar{u})$ with $\bar{x}_c \in \mathcal{X}_c$. Then, since both the model and the plant are δISS, for any constant input $\bar{u} = \bar{u}(\bar{y}^r)$, they admits a unique equilibrium $(\bar{x}, \bar{u}, \bar{y}(\bar{u}))$ with $\bar{x} \in \mathcal{X}_m$. Then, if this equilibrium is such that $\bar{y}\big(\bar{u}(\bar{y}^r)\big) = \bar{y}^r$, asymptotic zero-offset control is achieved.

8.2 Numerical example

8.2.1 Benchmark system description

The performances of the proposed control architecture have been tested on the Quadruple Tank benchmark system described in [58,151]. This system, depicted in Figure 8.5, consists of four tanks containing water. The levels of these tanks are denoted by h_1, h_2, h_3, and h_4. The tanks are fed by two pumps, which deliver the water flow rates q_a and q_b flows. Specifically, a triple valve splits the water flow rate q_a in $q_1 = \gamma_a q_a$ and $q_4 = (1-\gamma_a)q_a$, supplied to Tank 1 and Tank 4, respectively, and another triple valve splits

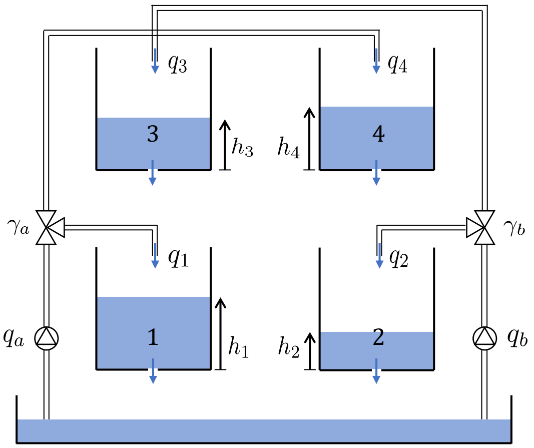

Figure 8.5: *Quadruple tank benchmark system [58].*

the water flow rate q_b in $q_2 = \gamma_b q_b$ and $q_3 = (1-\gamma_b)q_b$, supplied to Tank 2 and Tank 3, respectively.

The system is therefore described by the following equations

$$\begin{aligned}
\dot{h}_1 &= -\frac{a_1}{S}\sqrt{2gh_1} + \frac{a_3}{S}\sqrt{2gh_3} + \frac{\gamma_a}{S}q_a, \\
\dot{h}_2 &= -\frac{a_2}{S}\sqrt{2gh_2} + \frac{a_4}{S}\sqrt{2gh_4} + \frac{\gamma_b}{S}q_b, \\
\dot{h}_3 &= -\frac{a_3}{S}\sqrt{2gh_3} + \frac{1-\gamma_b}{S}q_b, \\
\dot{h}_4 &= -\frac{a_4}{S}\sqrt{2gh_4} + \frac{1-\gamma_a}{S}q_a,
\end{aligned} \tag{8.14a}$$

where the parameters of the system have been reported in Table 8.1. The water levels, as well as the control variables, are also subject to saturation limits

$$\begin{aligned}
h_i &\in [\underset{\wedge}{h}_i, \check{h}_i] \quad \forall i \in \{1, ..., 4\}, \\
q_a &\in [\underset{\wedge}{q}_a, \check{q}_a], \\
q_b &\in [\underset{\wedge}{q}_b, \check{q}_b].
\end{aligned} \tag{8.14b}$$

In the following it is assumed that only h_1 and h_2 are measurable. That is, the output vector of the system is $y = [h_1, h_2]'$, while the control variables are the pumps flow rates, i.e. $u = [q_a, q_b]'$. The control goal is to steer the system's output y_k to the reference $\bar{y}_k$ mimicking the response of the reference model $\mathcal{M}_r$, later specified.

The synthesis of the IMC scheme is articulated in three steps: (*i*) learning an RNN model of the system, $\Sigma_m(\Phi_m)$; (*ii*) generating a dataset of

Parameter	Value	Units	Parameter	Value	Units
a_1	$1.31 \cdot 10^{-4}$	m^2	$[\underset{\wedge}{h}_1, \check{h}_1]$	$[0, 1.36]$	m
a_2	$1.51 \cdot 10^{-4}$	m^2	$[\underset{\wedge}{h}_2, \check{h}_2]$	$[0, 1.36]$	m
a_3	$9.27 \cdot 10^{-5}$	m^2	$[\underset{\wedge}{h}_3, \check{h}_3]$	$[0, 1.3]$	m
a_4	$8.82 \cdot 10^{-5}$	m^2	$[\underset{\wedge}{h}_4, \check{h}_4]$	$[0, 1.3]$	m
S	0.06	m^2	$[\underset{\wedge}{q}_a, \check{q}_a]$	$[0,\ 9 \cdot 10^{-4}]$	$\frac{\mathrm{m}^3}{s}$
γ_a	0.3		$[\underset{\wedge}{q}_b, \check{q}_a]$	$[0,\ 1.3 \cdot 10^{-3}]$	$\frac{\mathrm{m}^3}{s}$
γ_b	0.4				

Table 8.1: *Parameters of the quadruple tank benchmark system*

feasible reference trajectories; (*iii*) learning a controller RNN $\Sigma_c(\Phi_c)$ that approximates the model's inverse for the given output reference trajectories. In the following subsections, these three tasks are tackled.

8.2.2 Model identification

The Quadruple Tank system described by (8.14) has been implemented in MATLAB and, in order to collect the data required for the training of the model, it has been fed with MPRS so as to properly excite the system and to collect data in a broad operating region. Considering a sampling time of $\tau_s = 25$s, a pair of input-output sequences of length $T_{\text{tr}} = 15000$ has been collected for training, while pairs sequences of length $T_{\text{val}} = 5000$ and $T_{\text{te}} = 700$ have been collected for validation and testing, respectively. According to the TBPTT paradigm, $N_{\text{tr}} = 200$ partially-overlapping subsequences of length $T_s = 700$ have been randomly extracted from the training sequences, while $N_{\text{val}} = 25$ subsequences have been extracted from the validation data. As highlighted in Remark 4.1, the data needs to be normalized in such a way that the saturation constraints (8.14b) translate into the unity-boundedness of inputs and outputs.

The RNN architecture here considered to identify the plant is a deep GRU model with $L = 2$ layers, with $n_c^{(1)} = n_c^{(2)} = 10$ units. The training has been carried out by means of Algorithm 1, implemented in TensorFlow 1.15 [152] using RMSProp as optimizer. In particular, the washout period and the batch size have been set to $\tau_w = 25$ and $|\mathcal{I}_{\text{tr}}^{\{b\}}| = 15$, respectively. The δISS of the model has been also enforced including the piecewise-linear δISS regularization constraint (4.9) in the loss function, where $\bar{\pi} = 5 \cdot 10^{-4}$, $\underline{\pi} = 10^{-5}$, and $\varepsilon_\nu = 0.03$.

In Figure 8.6 the evolution of the loss function and of the validation MSE performance index throughout the training procedure is reported. Overall,

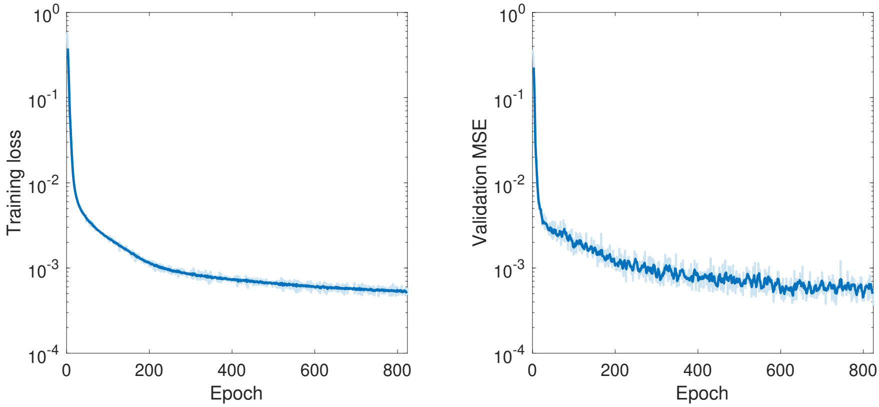

Figure 8.6: *Model training procedure: on the left, the evolution of the training loss function is depicted in logarithmic scale; on the right, the evolution of the validation MSE metrics is depicted. In both plots, the actual curves (light-blue lines) and their moving average (bold blue lines) are illustrated.*

the training procedure took 823 epochs, and it led to a δISS model, since the residuals of conditions (3.46) achieved, $\nu(\Phi_m^\star) = [-0.065, -0.088]$, are strictly negative. Of course, in view of Remark 3.6, the model is also ISPS.

The performances of the trained RNN model have been tested on the independent test dataset. In Figure 8.7 the two inputs' sequences are reported, while in Figure 8.8 the two outputs of the free-run simulation of the models are compared to the measured ones. The corresponding FIT index scored by the RNN model $\Sigma_m(\Phi_m^\star)$ is 96.5%, which indicates remarkable modeling performances.

8.2.3 Generation of feasible reference trajectories

Having learned an accurate model of the system, as discussed in Section 8.1.1, it is now necessary to generate a dataset of suitable reference trajectories for the training procedure of the controller RNN $\Sigma_c(\Phi_c)$. This dataset consists of a collection of reference signals that the controller should learn to track – no data needs to be collected from the real system at this stage. Since the control system will be employed to track piecewise-constant references signals, such dataset is generated by filtering MPRS signals, denoted as $\bar{y}^{\{i\}}_{0:T_{c,s}}$, with the model reference $\mathcal{M}_r$, yielding the filtered reference signals (8.5). The model reference here considered is the discrete-time equivalent of two decoupled first-order systems with unitary static gain and time constant $\tau_r = 2000$s.

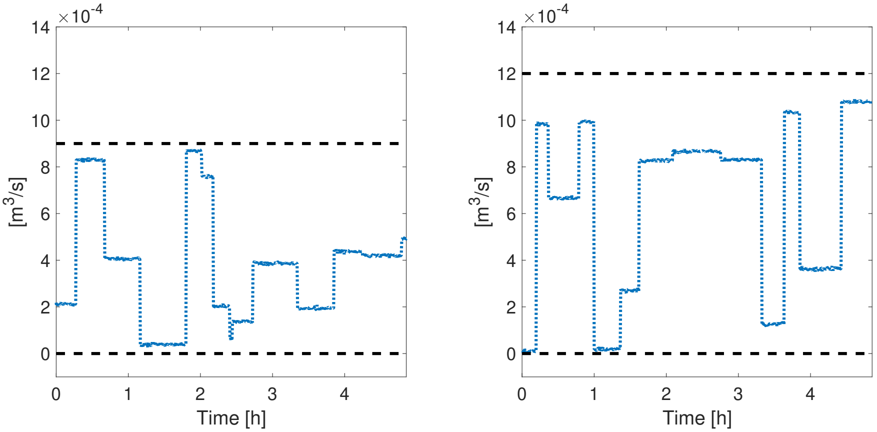

Figure 8.7: *Test dataset's input sequences used to assess the performances of the trained RNN model: the pump flow rate q_a is shown on the left, the pump flow rate q_b is reported on the right.*

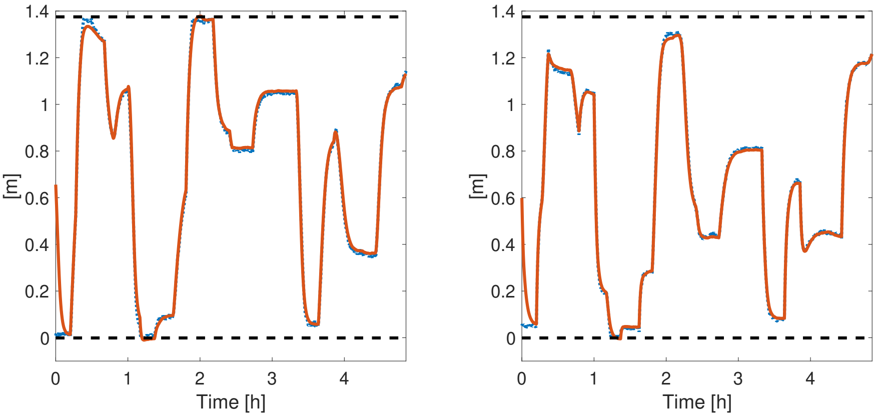

Figure 8.8: *Output of the RNN model's free-run simulation corresponding to the test dataset's input sequences (red solid line) compared to the actual measured outputs (blue dotted line). The level h_1 is depicted, while on the right the level h_2 is shown.*

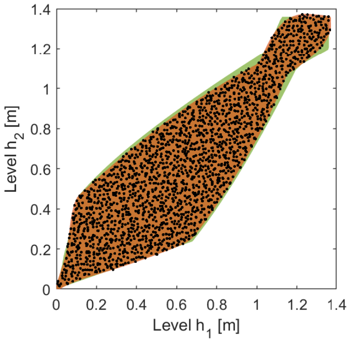

Figure 8.9: *Generation of the dataset for the training procedure of the controller RNN. The setpoints (black dots) are extracted from the set of model's feasible output setpoints (orange area); the green area corresponds to the set of plant's feasible outputs, generally unknown.*

We recall that, for a successful controller training procedure, the reference signals' dataset must be generated appropriately. Indeed, the inclusion of unfeasible setpoints $\bar{\bar{y}}$, i.e. setpoints which (given the input and output constraints (8.14b)) do not correspond to any feasible equilibrium of the model, would inevitably bias the loss function's gradient. This would lead, in turn, to poor performances of the trained control scheme [62].

Notably, owing to the δISS of the model (8.1), to check the feasibility of some setpoint $\bar{y}^r$ boils down to assess the existence of a corresponding feasible equilibrium, i.e., a tuple $(\bar{x}_m, \bar{u}, \bar{y}^r)$ such that

$$\begin{cases} \bar{x}_m = f_m(\bar{x}_m, \bar{u}; \Phi_m^\star) \\ \bar{y}^r = g_m(\bar{x}_m; \Phi_m^\star) \end{cases} \tag{8.15}$$

where $\bar{x}_m \in \mathcal{X}_m$, see (3.39), and $\bar{u} \in \mathcal{U}$. Moreover, the model's δISS guarantees that, if such equilibrium exists, it can be reached starting from any initial $x_{m,0} \in \mathcal{X}_m$. This consideration allows to further relieve the computational complexity of solving (8.15), since initial guesses of $\bar{u}$ and $\bar{x}_m$ can be retrieved with open-loop simulations of (8.1) and then used to warm-start the nonlinear problem underlying the equilibrium computation.

Overall, the generated controller's training set consists of $N_{c,\text{tr}} = 380$ reference trajectories for training and $N_{c,\text{val}} = 40$ reference trajectories for validation. All these trajectories, which have a length $T_{c,s} = 600$ time-steps, are obtained by filtering the feasible piecewise-constants setpoints $\bar{y}^{\{i\}}_{0:T_{c,s}}$, where $i \in \{1, ..., N_{c,\text{tr}} + N_{c,\text{val}}\}$, with the aforementioned model reference. In Figure 8.9 such randomly-generated training setpoints are depicted and compared to the set of feasible model outputs, as well as to the set of feasible outputs of the real plant[7]. A longer reference trajectory is also generated to be used as an independent test set for assessing the controller's nominal performances. Such a sequence, denoted by $y^r_{\text{te},0:T_{\text{te}}}$, satisfies the same conditions as the training and validation reference trajectories, and has a length equal to $T_{\text{te}} = 1800$ time-steps.

8.2.4 Controller training

The controller (8.3) is designed as deep GRU with $L = 3$ layers with 5 units each. The training procedure of $\Sigma_c(\Phi_c)$ follows Algorithm 2, and has been carried out with TensorFlow 1.15, using RMSProp as optimizer and a batch size $|\mathcal{I}^{\{b\}}_{c,\text{tr}}|= 20$. Moreover, the δISS of the controller has been enforced by means of the piecewise-linear δISS regularization term (4.9), see the loss function (8.6), penalizing the violation of condition (3.46). The chosen regularizer parameters are $\overline{\pi} = 3 \cdot 10^{-4}$, $\underline{\pi} = 5 \cdot 10^{-5}$, and $\varepsilon_\nu = 0.03$.

In Figure 8.10 the evolution of the training loss throughout the controller's training procedure, as well as the validation MSE performance index, are depicted. After 1543 epochs, the training was halted when the performances stopped to improve, yielding the controller weights $\Phi^\star_c$.

Eventually, the nominal performances of the trained controller $\Sigma_c(\Phi^\star_c)$ have been tested on the independent test reference trajectory $y^r_{\text{te},0:T_{\text{te}}}$. To this end, the output trajectory of the model $\Sigma_m(\Phi^\star_m)$ under the open-loop control law generated by $\Sigma_c(\Phi^\star_c)$ has been computed as

$$y_{m,0:T_{\text{te}}}\big(x_{m,0}, u_{0:T_{\text{te}}}(x_{c,0}, y^r_{\text{te},0:T_{\text{te}}}; \Phi^\star_c); \Phi^\star_m\big). \tag{8.16}$$

In Figure 8.11 the controlled model's outputs are compared to the reference signals. Note that, in addition to limited steady-state tracking errors, cross-coupling effects between the two outputs are present. Despite being quickly compensated, these effects may deteriorate the FIT index[8]. Nonetheless, $\text{FIT} = 90.4\%$ index scored, which indicates fair control performances.

8.2.5 Closed-loop results

The closed-loop performances of the proposed control architecture have been tested on the simulated Quadruple Tank benchmark system. For the sake of interpretability, no output noise has been considered. Analogous results featuring white Gaussian noise affecting the plant's outputs are however reported in [62].

[7] This latter set is generally unknown, and is here reported just for comparison purposes.

[8] The controller's FIT index is computed between (8.16) and its reference $y^r_{\text{te},0:T_{\text{te}}}$ according to (4.6).

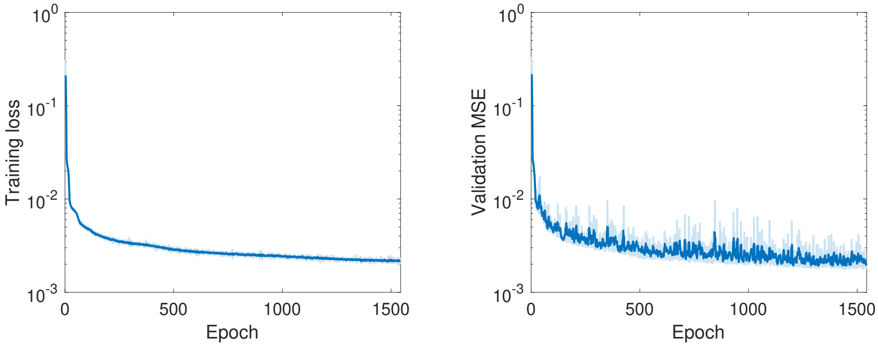

Figure 8.10: *Controller training procedure: on the left, the evolution of the loss function is depicted in logarithmic scale; on the right, the evolution of the validation MSE metrics is depicted. In both plots, the actual curves (light-blue lines) and their moving average (bold blue lines) are illustrated.*

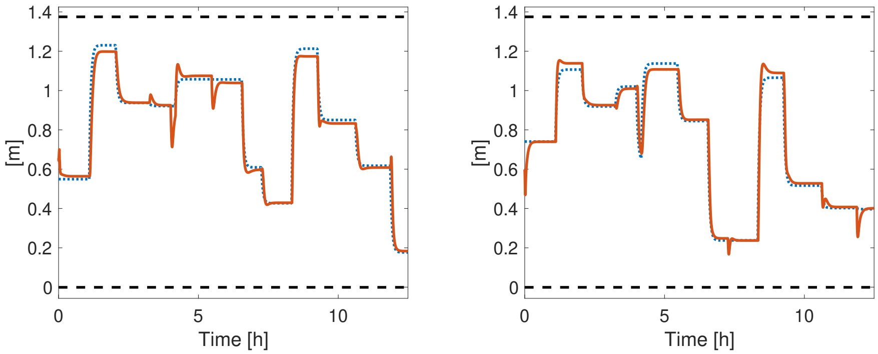

Figure 8.11: *Performances of the trained controller on the independent test set reference trajectory: IMC-controlled nominal output trajectory (red solid line) compared to the filtered reference (blue dotted line). The level h_1 is displayed on the left, the level h_2 on the right.*

Consistently with the internal model control literature [62, 133], the modified architecture shown in Figure 8.12 has been considered, where the modeling error feedback is filtered by a suitably designed low-pass filter. As customarily done, we here adopt a low-pass filter $\mathcal{F}$ with the same time constant as the model reference $\mathcal{M}_r$, i.e. $\tau_r = 2000\text{s}$. It is worth noticing that the low-pass filter does not affect the closed-loop stability [133], since the modeling error feedback $e_{m,k}$ acts as an additive disturbance on the reference y^r_k, see (8.4), and it converges to the bounded plant-model mismatch (or to zero, in the case of perfect model), as illustrated in Section 8.1.2.

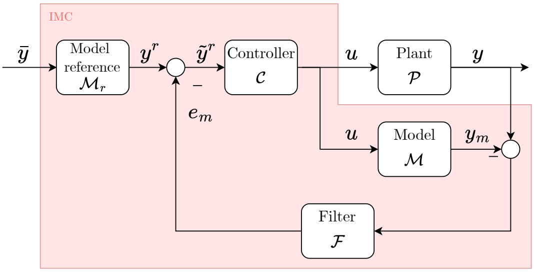

Figure 8.12: *IMC architecture with filtered modeling error feedback.*

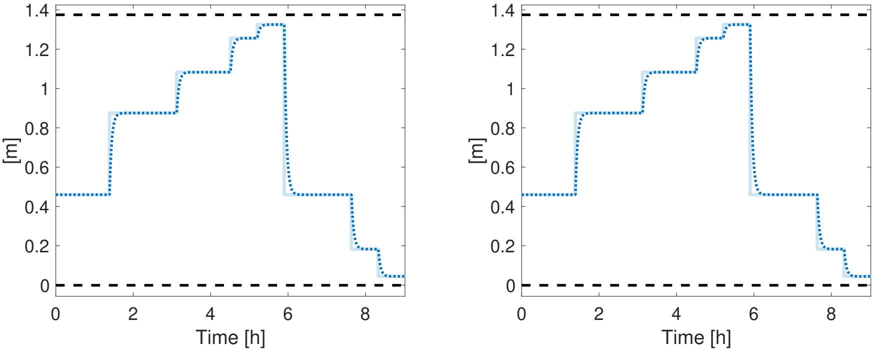

Figure 8.13: *Reference trajectories used for assessing the closed-loop performances: piecewise-constant reference $\bar{y}_k$ (light blue solid line) and filtered reference y_k^r (blue dotted line). The reference for level h_1 is reported on the left, that for level h_2 on the right.*

The closed-loop performances of the proposed approach have been tested and compared to those of two other alternative control architectures:

a. An IMC control architecture realized using FFNNs as system model and controller, along the lines of [137], adapted to work with MIMO systems; zero control delay has been assumed and the adopted FFNNs have been designed to embed the previous $H = 6$ input-output data points.

b. A standard nonlinear MPC architecture synthesized using the GRU model $\Sigma_m(\Phi_m^\star)$ as predictive model, along the lines of the strategy discussed in Section 6.1. The adopted prediction horizon is $N = 50$, whereas the weight matrices are $Q = I_{n_x,n_x}$, $R = I_{2,2}$, $P = 25 \cdot I_{n_x,n_x}$. A control horizon of $N_c = 15$ has also been adopted to relieve the computational complexity.

For this comparison, the reference trajectories depicted in Figure 8.13 have been adopted. Notice that the reference $\bar{y}_k$ is a piecewise-constant signal spanning the set of model's feasible steady states depicted in Figure 8.9.

In Figure 8.14 the closed-loop output tracking performances of the three implemented control architectures (i.e. the proposed IMC approach, the NMPC, and the FFNN-based IMC) are depicted. The corresponding output tracking error, $y_k^r - y_k$, is shown in Figure 8.15. As expected, cross-couplings are promptly rejected and the controlled outputs are kept close to their reference values. Moreover, the controller's architecture allows to satisfy the input saturation constraint[9], as illustrated in Figure 8.16.

Let us now introduce some performance indices to evaluate and compare the closed-loop performances of the control algorithms here considered.

The first performance index considered is the tracking Root-Means-Square Error (RMSE), i.e. the RMSE between the output reference and the controlled output trajectory. Letting T_{cl} indicate the duration of the closed-loop experiment, the RMSE is defined as

$$\epsilon_e = \frac{\|y^r_{0:T_{cl}} - y_{0:T_{cl}}\|_{2,2}}{\sqrt{T_{cl}}}. \tag{8.17a}$$

Of course, the smaller ϵ_e, the better the reference tracking capabilities of the the control scheme.

Moreover, in order to evaluate the static performances of the three control architectures, the steady-state output tracking error has been computed. To this end, each time the output setpoint is changed, it is qualitatively verified if, after a sufficiently long period of time, the closed-loop output is settled. In such case, the steady-state ouput tracking error is computed as

$$\epsilon_{ss} = \|\bar{y}_{k_{ss}} - y_{k_{ss}}\|_2, \tag{8.17b}$$

where k_{ss} denotes a time index at which the closed-loop is at steady state. This index, however, has not been computed for the FFNN-IMC since, as apparent from Figure 8.14, it is generally unable to reach a steady-state condition. Based on ϵ_{ss}, two indexes of the closed-loop static performance have been defined. Namely, the maximum steady-state output tracking error, i.e. $\check{\epsilon}_{ss} = \max(\epsilon_{ss})$, and its mean value, i.e. $\bar{\epsilon}_{ss} = \mathbb{E}[\epsilon_{ss}]$. These quantities are, of course, empirically computed over all the setpoints issued in closed-loop operations.

The performance indexes achieved by the three control schemes, as well as the computational cost, are reported in Table 8.2. Based on the results obtained, the following conclusions can be drawn concerning the strengths of the proposed control strategy.

[9] In the implemented NMPC law, input constraints have been explicitly stated in the optimization problem, while in the FFNN-based IMC a $\tanh$ activation function has been used to constrain the generated control variable to be unity-bounded.

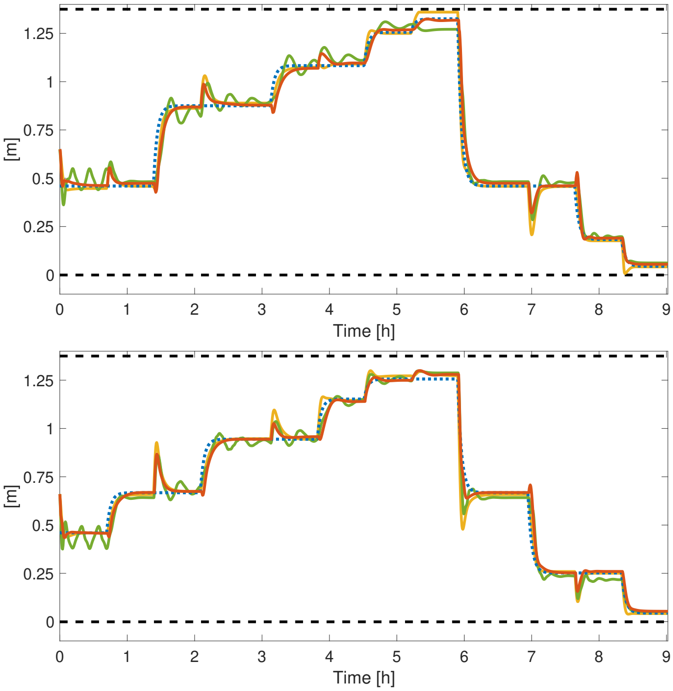

Figure 8.14: *Closed-loop performances of the three control architectures: plant's output under NMPC (yellow solid line), FFNN-IMC (green solid line), and the proposed IMC approach (red solid line). The controlled outputs are compared to their reference values (blue dotted line) and constraints (black dashed lines). Level h_1 is displayed on top, level h_2 on the bottom.*

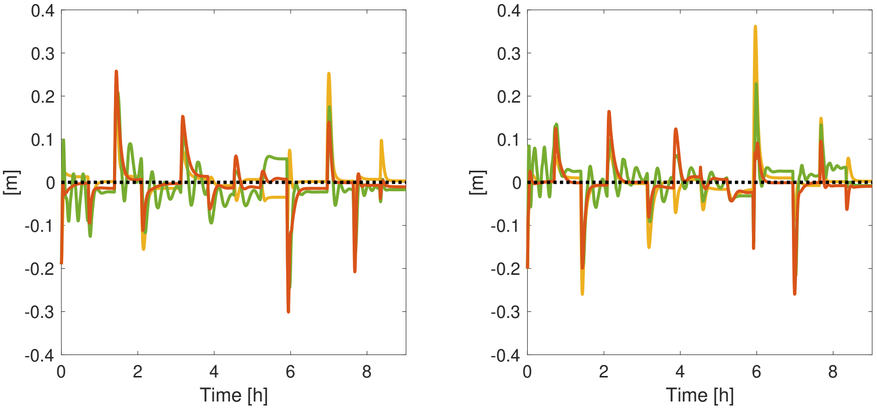

Figure 8.15: *Closed-loop output tracking error achieved by the NMPC (yellow solid line), FFNN-IMC (green solid line), and by the proposed IMC approach (red solid line). The tracking error associated to output h_1 is displayed on the left, that associated to output h_2 on the right.*

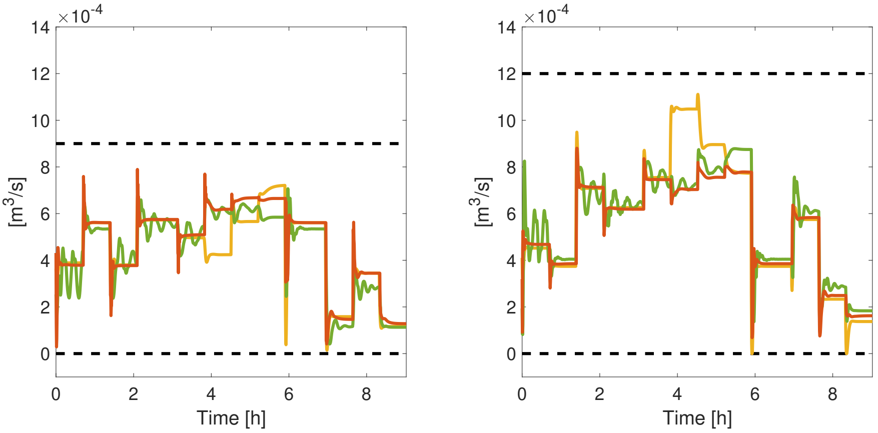

Figure 8.16: *Comparison of the control action requested by NMPC (yellow solid line), FFNN-IMC (green solid line), and by the proposed IMC approach (red solid line). Input q_a is displayed on the left, q_b on the right.*

	NMPC	FFNN-IMC	Proposed IMC
Average computational time [s][†]	3.82	$8.4 \cdot 10^{-3}$	$1.7 \cdot 10^{-3}$
Tracking RMSE ϵ_e [m]	0.133	0.131	0.128
Average steady-state error $\bar{\epsilon}_{ss}$ [m]	$0.82 \cdot 10^{-2}$	-	$0.79 \cdot 10^{-2}$
Maximum steady-state error $\check{\epsilon}_{ss}$ [m]	$3.46 \cdot 10^{-2}$	-	$2.35 \cdot 10^{-2}$
Closed-loop input-output stability	Nominal	No	Yes

Table 8.2: *Performances of the proposed IMC approach*

i. **Performances** – Owing to the superior modeling capabilities of gated RNNs, the proposed IMC approach outperforms the FFNN-based IMC, especially from the steady-state tracking error perspective. Surprisingly, it also slightly outperforms NMPC. This is probably due to the non-ideal performance of the state observer, which is required to synthesize a state-feedback NMPC law with a black-box model such as the considered GRU network.

ii. **Computational time** – The computational load of IMC lies entirely upon the (offline) synthesis stage. Therefore, the proposed IMC approach requires limited online computational costs, which consist of the propagation of the model $\Sigma_m(\Phi_m^\star)$ and of the controller $\Sigma_c(\Phi_c^\star)$. The proposed approach beats the slightly higher cost of the FFNN-based IMC, as this latter requires the storage of a sufficient amount of

[†] Average computational time at each control step. The control architectures have been implemented on a desktop with a 4x4GHz processor and 16Gb of RAM.

past data points. As expected, the computational burden of NMPC is significantly higher, since it requires to solve an online nonlinear optimization problem at each step, that is likely unbearable for low-power embedded control systems.

iii. **Closed-loop stability** – In light of Property 8.4, since the model and controller RNNs are trained with δISS guarantees, the input-output stability of the closed-loop is guaranteed. The adopted NMPC law allows instead to guarantee the closed-loop stability in the nominal case by relying on the arguments discussed in Section 6.1. On the contrary, no criterion for synthesizing closed-loop stable FFNN-based IMC control strategies has been provided in [137], especially for MIMO systems.

The advantages of the proposed approach, however, come at the cost of a more involved training procedure. Indeed, as discussed in Section 4, training provenly δISS RNNs generally entails a fairly involved and time-consuming training procedure.

8.3 Summary

In this section, an internal model control strategy based on gated RNNs has been proposed. The synthesis of such control architecture has been recast to a rather standard RNN's training procedure, in which a controller network is trained to approximate the inverse of the plant's RNN model for a given class of output reference trajectories. The δISS stability conditions discussed in previous chapters have also been exploited to ensure some degree of closed-loop input-output stability of the scheme.

The proposed approach has been tested on the Quadruple Tank nonlinear benchmark system, and the achieved closed-loop performances have been compared to alternative control architectures. The results witness remarkable static and dynamic performances, and a reduced computational burden, which make the proposed control strategy suitable to contexts where limited online computational resources are available.

Part III

Towards practical applications of deep learning for control

CHAPTER 9

Further work and hints for future research objectives

As discussed in the previous chapters, owing to their modeling capabilities, RNNs have the potential to play an increasingly important role in the design and control of dynamical systems. To this end, there is a number of open issues that, albeit preliminarily addressed, need further research efforts [20].

In the following, these issues are briefly summarized, and hints to the proposed solutions, as well as to their current limitations, are provided.

9.1 Safety verification

In the context of neural network-based regression, one of the most well-known problems is that of *output range analysis* or *safety verification* [153, 154]. In brief, this problem involves determining a set that encloses all the possible outputs that correspond to a given set of inputs, and then assessing whether such output set lies within some known safe set.

In the realm of (recurrent) neural networks approximating dynamical systems, this problem corresponds to the computation of the *output reachable set*, and is known to be considerably involved, since one would need to consider the infinite-horizon evolution of the system's output for any possible initial state and input signal from prescribed sets. Once such output reachable set is available, one certifies the safety/reliability of the model by checking whether the latter set lies in some space of interest, e.g., that space of physically meaningful outputs or the space where training, validation, and testing data were collected.

Unfortunately, due to its infinite dimensionality, the estimation of the output reachable set is difficult to perform for generic nonlinear dynamical systems. In the context of NNs, most results nowadays available have been developed for FFNNs, see [155–158]. While in principle these approaches can be also applied to RNNs by "unrolling" them and then applying FFNNs' verification algorithms [159], how to choose the timespan throughout which the RNN is simulated and unrolled is still unclear.

In the following, two possible solutions to the problem of calculating the output reachable set of RNNs are outlined. For more details, the interested reader is addressed to [20, 65].

9.1.1 Analytical bound on the output reachable set

As discussed in Chapter 2.3, the ISPS, ISS, and δISS properties allow to establish analytical bounds of the output reachable set. For the sake of generality, let us consider an ISPS[1] RNN model in the state space form (3.1). Under the assumption of Lipschitz-continuity of its output transformation, the system is also IOPS, see Section 8.1.2. Hence, there exist functions $\beta_y \in \mathcal{KL}$ and $\gamma_y \in \mathcal{K}_\infty$, and a scalar $\varrho_y > 0$ such that

$$\|y_k(x_0, u_{0:k})\|_p \leq \beta_y(\|x_0\|_p, k) + \gamma_y(\|u_{0:k}\|_{p,\infty}) + \varrho_y \tag{9.1}$$

for any $x_0 \in \mathcal{X}$ and any $u_{0:k} \in \mathcal{U}$.

In the context of the safety verification, given a set of candidate input variables $\tilde{\mathcal{U}} \subseteq \mathcal{U}$ and a set of candidate initial states $\tilde{\mathcal{X}}_0$, the aim is to find a set $\tilde{\mathcal{Y}}(\tilde{\mathcal{X}}_0, \tilde{\mathcal{U}})$ within which any possible output trajectory must lie, see Definition 2.12. Such a set can be analytically computed by means of (9.1). Indeed, in light of the definitions of β_y and γ_y, it hold that

$$\tilde{\mathcal{Y}}(\tilde{\mathcal{X}}_0, \tilde{\mathcal{U}}) = \left\{ y \in \mathbb{R}^{n_y} : \|y\|_p \leq \beta_y\Big(\sup_{x_0 \in \tilde{\mathcal{X}}_0} \|x_0\|_p, 0\Big) + \gamma_y\Big(\sup_{u \in \tilde{\mathcal{U}}} \|u\|_p\Big) + \varrho_y \right\} \tag{9.2}$$

[1] We recall that, as discussed in Section 2, under minor assumptions both ISS and δISS imply the ISPS property.

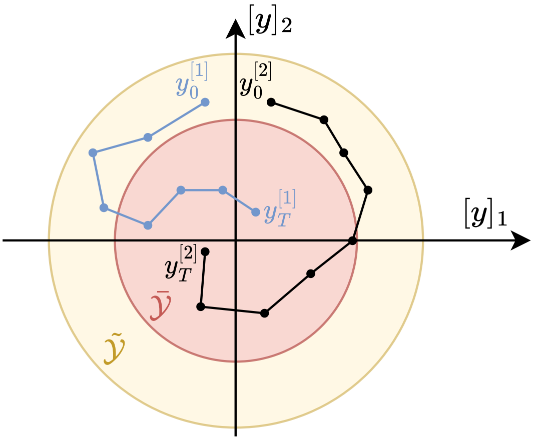

Figure 9.1: *Examples of analytical output reachable set (yellow area) and asymptotic output reachable set (red area) computed using the ℓ_2-IOPS functions, for a system with $n_y = 2$ output components.*

surely encloses any output trajectory generated by the system. Noticing that $\beta_y(\cdot, k) \to 0$ for $k \to \infty$, one can also neglect the first term of (9.2) and compute an asymptotic output reachable set $\bar{\mathcal{Y}}(\tilde{\mathcal{U}})$ as

$$\bar{\mathcal{Y}}(\tilde{\mathcal{U}}) = \left\{ y \in \mathbb{R}^{n_y} : \|y\|_p \leq \gamma_y \left(\sup_{u \in \tilde{\mathcal{U}}} \|u\|_p \right) + \varrho_y \right\}. \tag{9.3}$$

A graphical interpretation of these sets is provided in Figure 9.1.

Albeit (9.2) and (9.3) represent theoretically-sound bounds on the output reachable set, these analytical expressions are often overly conservative, to the point of not being usable for safety verification purposes. Less conservative numerical methods, such as the one discussed below, would therefore be recommended.

9.1.2 Numerical bound on the output reachable set via randomized procedures

While the analytical bounds implied by the ISPS, ISS, and δISS may be too conservative to be employed for safety verification, as pointed out in [20], these properties represent solid theoretical foundations for the numerical estimation of the output reachable set, since they imply the continuity at the equilibrium point, the existence of a uniform asymptotic gain, and ultimately the boundedness itself of the reachability set [160].

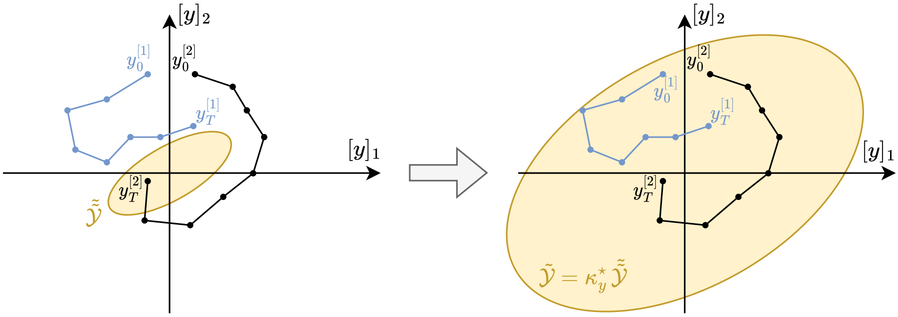

Figure 9.2: *Application of the numeric output reachable set computation for $S = 2$ trajectories and an ellipsoidal convex set $\tilde{\tilde{Y}}$.*

Under these premises, tighter bounds can be numerically computed via randomized algorithms (such as the Scenario Approach, see [161]). These approaches, which have been illustrated in [20,65,162] consist of two steps.

Given the sets $\tilde{\mathcal{X}}_0$ and $\tilde{\mathcal{U}}$ of initial states and inputs candidates, respectively, a suitably large number of output trajectories is first generated. To this end, letting $s \in \{1, ..., S\}$, S pairs $(x_0^{[s]}, u_{0:T}^{[s]})$ of initial states and input sequences are randomly extracted from $\tilde{\mathcal{X}}_0 \times \tilde{\mathcal{U}}_{0:T}$ according to some possibly unknown probability distribution, with the simulation horizon T being sufficiently long.

Then, a suitably designed [162] convex set $\tilde{\tilde{\mathcal{Y}}} \subset \mathbb{R}^{n_y}$ (containing the origin) is scaled to ensure that it encloses all the generated output trajectories $y_{0:T}(x_0^{[s]}, u_{0:T}^{[s]})$. This is done by solving the following optimization problem

$$
\begin{aligned}
\kappa_y^\star(\tilde{\mathcal{X}}_0, \tilde{\mathcal{U}}) = \arg\min_{\kappa} \quad & \kappa \\
s.t. \quad & y_k(x_0^{[s]}, u_{0:k}^{[s]}) \in \kappa \tilde{\tilde{\mathcal{Y}}} \quad \forall k \in \{0, ..., T\}, \\
& \qquad\qquad\qquad\qquad\quad \forall s \in \{1, ..., S\}.
\end{aligned} \tag{9.4a}
$$

The numerically estimated output reachable set then reads as

$$
\tilde{\mathcal{Y}}(\tilde{\mathcal{X}}_0, \tilde{\mathcal{U}}) = \kappa_y^\star(\tilde{\mathcal{X}}_0, \tilde{\mathcal{U}})\, \tilde{\tilde{\mathcal{Y}}}, \tag{9.4b}
$$

see Figure 9.2 for an illustrative graphical representation.

It is worth noting that, owing to the ISPS of the system, it is known that the output reachable set is asymptotically independent from the initial state. Therefore, it is possible to obtain an even tighter estimate by enforcing the scaled convex set to enclose the output trajectories after a sufficient number of steps (washout period τ_w), after which the effects of initial condition are vanished. This yields the asymptotic output reachable set estimate $\bar{\mathcal{Y}}(\tilde{\mathcal{U}})$.

An advantage of this approach is that, by considering the pairs $(x_0^{[s]}, u_{0:T}^{[s]})$ as independent identically distributed samples drawn from an unknown probability distribution, it is also possible to associate, with some confidence, a violation probability to the output reachable set (9.4), see [20,162]. The approaches proposed in [41, 161, 163] indeed provide conditions on the number of samples S needed to achieve, under the unknown distribution generating the samples, the desired violation probability with the prescribed confidence level.

9.1.3 Open challenges

Although in [65] we verified that the numerically computed output reachable set yields results that are not overly conservative and, hence, are definitely adequate for the RNN model's safety verification, the main challenge of this approach lies in the definition of an appropriate convex set $\tilde{\tilde{\mathcal{Y}}}$ to be scaled. An inadequate definition of such set may indeed lead to potentially conservative results [162].

9.2 Lifelong learning

A common problem in the context of system identification is to ensure that the identified model remains representative of the learned physical system throghout its lifespan. In fact, for a multitude of reasons, a model that is initially extremely accurate may become inaccurate, e.g., in the event that the plant experiences faults or severe disturbances. Indeed, albeit the control strategies synthesized in Part II guarantee closed-loop stability and robustness properties in the nominal case, in case the plant undergoes changes, model adaptation is typically required to preserve the attained closed-loop stability guarantees and control performances.

This problem has been known for decades. For example, for linear models such as ARXs, the popular self-tuning approach has been proposed [164], which consists of the online tuning of the model by recursively solving a least square problem based on the measured output data.

In the machine learning realm, the problem of when and how to update a NN based on the most recently-collected data, known as the *lifelong learning problem*, has become increasingly important in recent years, see the recent reviews [165–168] and references theirein. In the specific case of RNNs employed for nonlinear system identification, two main events may call for such an adaptation framework. Namely, (*i*) the gradual loss of modeling performances, that could be due to plant's time-varying parameters (e.g., due to wear) or to slight variations of the plant's operating conditions; (*ii*) the dramatic loss of performances, caused by the fact that the plant moves to new and unexplored operating conditions (e.g., due to a fault) that the available model is unable to represent.

In the latter case, generally speaking, it may be necessary to adapt the structure of the model (e.g., adding neurons if necessary), collect data related to the new operating conditions, and repeat the training procedure from scratch [168].

On the other hand, in the former case, a reasonable solution is to use the data collected online to adapt the RNN model over time. How to conduct this online tuning procedure without storing an increasing amount of data, and avoiding the problem of catastrophic forgetting (which occurs when, while learning new data, previous knowledge stored in the model is forgotten), is still widely debated and represents a challenging and relevant research topic.

9.2.1 Moving Horizon Estimation approach

As a preliminary attempt to address case (*i*), i.e., the adaptation of an existing RNN model to plant variations associated to slowly time-varying parameters, in [66] we proposed an approach based on the Moving Horizon Estimation (MHE) strategy [116, 149]. In brief, this approach involves collecting measurements over a time window N after which, in case of an abnormal reduction of the model's performances, an optimization problem is solved to adjust the weights of the network. With a slight abuse of notation, such optimization problem may be stated as

$$\begin{aligned} \Phi_k^\star = \arg\min_{\Phi} \;\; & \mathrm{MSE}(y_{k-N:k|k}(x_{k-N}, u_{k-N:k}; \Phi), y_{k-N:k}) + \mu \|\Phi - \Phi_{k-N}^\star\|_2, \\ s.t. \;\; & \nu(\Phi) < 0, \end{aligned} \tag{9.5}$$

where the loss function seeks a trade off between the minimization of the RNN model's free-run simulation error over the freshly collected data and the minimization of the regularization term, which penalizes the displacement from the previously tuned weights $\Phi_{k-N}^\star$. This regularization term can be interpreted as a prior information that allows to limit the catastrophic forgetting problem. The constraint allows to preserve the satisfaction of the sufficient stability conditions, if any.

By relying on the results reported in [149], in [66] we have shown that under ideal conditions (i.e., existence of a unique exact parametrization Φ^o, no measurement noise, and known state x_{k-N}), there exists a condition on the regularization weight μ which guarantees the convergence to the exact solution Φ^o.

The approach has been tested on a chemical benchmark system undergoing a parameter drift. In this scenario, we have shown that the modeling performances of the initially trained RNN model would have significantly deteriorated in time, whereas the weights' fine-tuning via the proposed MHE strategy allowed to preserve the network's modeling performances.

9.2.2 Open challenges

Albeit the proposed strategy led to good results on the considered benchmark system, further research efforts are needed to address the limitations of this algorithm. First and foremost, as discussed in Chapter 6, the RNNs' states are often unmeasurable and must be estimated by means of properly designed state observers. Note that they could alternatively be introduced as optimization variables in (9.5), but the impact of such strategy needs to be assessed. Secondly, the proposed approach does not scale well with respect to network size. In this regard, one might adopt gradient-based strategies, as proposed in [169]. Finally, one might consider designing an adaptation strategy in conjunction with a control law that encourages exploration, according to the dual control paradigm, so as to improve the informativeness of the data collected online.

9.3 Physics-based modeling

In recent years, the scientific community has devoted considerable research interest to the possibility of exploiting the available knowledge of the physical system to be identified to overcome the limitations of purely black-box RNN-based identification procedures, such as those discussed in this thesis.

The new branch of machine learning that merges physical knowledge into RNN modeling takes various names, such as Theory-Guided Data Science [26] and Physics-Based Modeling [170, 171]. The idea underlying these approaches is that by leveraging quantitative or qualitative knowledge of the system under consideration, highly desirable results can be obtained, such as improved model interpretability (e.g., by associating physical meaning to the states of the identified RNN model), improved generalizability to new operating conditions for which data are not available in the training set, improved modeling performances, and finally a faster training procedure, see [19, 172].

In [20] an attempt has been made to sketch a classification of the main strategies for nonlinear system identification via physics-based RNNs, as outlined in the following. For more details, the interested reader is addressed to [20].

9.3.1 Overview of the main approaches for physics-based RNNs

Physical knowledge of a system can be exploited in two ways, namely, during the design of the RNN architecture or during the training of the network itself.

Physics-based structure design

The first strategy is to encode the physical knowledge in the RNN model via a suitable design of the network architecture [26, 170]. Examples of this paradigm are reported below.

i. Models with known and measurable states

When the states of the system to be identified are known and measurable, a simple – yet effective – strategy may rely on FFNN to learn the increments of the discretized state variables. That is, assume the plant to be described by the unknown continuous-time system

$$\dot{x}_c(t) = f_c(x_c(t), u(t)),$$

where the state variables are fully measured, and t is the continuous time index. By discretizing such system with the sampling time τ via a generic explicit method, such as Explicit Runge-Kutta, one obtains the discrete-time approximation

$$x_{k+1} = x_k + \tau f(x_k, u_k).$$

Based on this idea, a FFNN can be used to learn the function $f(x_k, u_k)$ that describes the discrete-time state increment. This paradigm is that adopted by the popular ResNet [173] and ODE-NN [174] architectures.

ii. Models with known states and structure parametrized by NN

Another case is that of physical systems that enjoy a model with measurable states and known structure, which however depends on parameters that are unknown or varying based on the plant's operating conditions, i.e.

$$x_{k+1} = f(x_k, u_k; \Theta(x_k, u_k)).$$

In such case, a common and effective choice is to train a neural network, generally a FFNN one, that approximates the map $\Theta(x_k, u_k)$, see e.g. [29].

iii. Models with known relationships among variables

When used to learn physical systems, the outputs of a model should generally fulfill physical conditions. Indeed, constraints may affect outputs individually, such as positivity, monotonicity, and range bounds, or may even involve relationships between them, such as a zero-sum constraints coming from mass or energy conservation. An idea for exploiting this rather qualitative physical knowledge is to design the RNN model architecture so as to ensure that these constraints are met. This can ensure the consistency of the model to the physical system and, by limiting its unnecessary representational power, often results in training procedures that converge faster and more robustly with respect to the weights' initialization. Some examples of this paradigm, which we deem very relevant and didactic, are given in [170, 175, 176].

iv. Models reflecting the plant's block structure

Many complex processes can be decomposed in sparsely-interconnected subsystems, that is, subsystems whose dynamics are directly influenced only by few neighboring ones. If such subsystems can be easily recognized, and the coupling variables that describe their mutual influences are measured, one can design the architecture of the RNN so as to mimic the plant's sparsely-interconnected structure. In this way, non-physical mutual interactions between subsystems are avoided, which allows to obtain a more reliable model, less subject to overfitting, as well as significantly faster training procedures, see [30, 177].

Physics-guided loss function formulation

Instead of enforcing the consistency to physical relationships via structure selection, one may also embed the physical knowledge of the plant by suitably designing the training loss function, with the aim of encouraging the attainment of the desired behavior by the trained RNN model. Such approach is particularly useful when the physical relationships to be enforced can not be easily encoded via architecture design, and is, in a sense, reminiscent of constraint relaxation.

From this point of view, the idea of enforcing RNNs' ISPS, ISS, and δISS properties by penalizing the violation of their sufficient condition in the loss function, discussed in Chapter 4, can be regarded as an application of physics-guided cost function design, which aims to ensure consistency of the model to the known plant's stability-like properties [58].

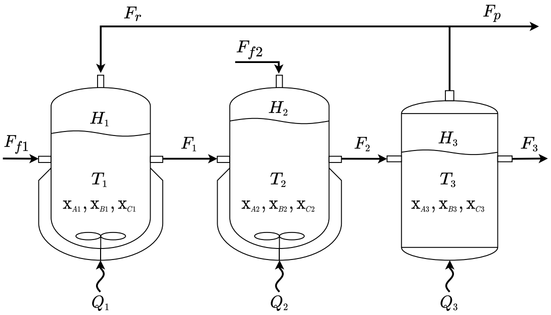

Figure 9.3: *Chemical process used as benchmark for the physics-based RNN design.*

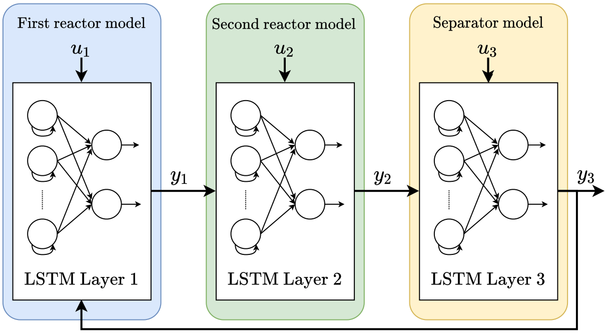

Figure 9.4: *Architecture of the adopted physics-based RNN model.*

9.3.2 Application to a chemical process

To exemplify the application of physics-based RNN principles and assess its enhanced performance over purely black-box models, in [20] we considered the chemical system depicted in Figure 9.3. This process is characterized by two continuously stirred reactors, cascaded with a separator. A fraction of the solution is also fed back to the first reactor.

Even without entering into the details of the process' model, it can be easily noticed that the system is composed by three subsystems, each characterized by its own states and dynamics, which are sparsely interconnected. For example, the second reactor receives a solution from the first one and feeds, in turn, the separator. Such a case falls naturally into what has been called "plant's block structure". In light of this rather qualitative insight, one can think to design a RNN model that enjoys the very same block structure. In particular, we considered the model depicted in Figure 9.4, which clearly mimics the process structure. This design allows to rule out unphysical connection between subsystem, meaning that, for example, the output of the separator can not directly affect the states of the preceding reactor.

A further physical argument that has been exploited for the RNN architecture design is that the products' concentrations (denoted as x_{Ai} and x_{Bi} in Figure 9.3) lie in the range $(0, 1)$. A sigmoidal function has been therefore applied to the corresponding model's outputs, to guarantee that they lie in such range.

Finally, the concentration of product C, i.e. x_{Ci}, is generally included neither in the data, nor in the model, since it can be derived from the relationship $\mathrm{x}_{Ai} + \mathrm{x}_{Bi} + \mathrm{x}_{Ci} = 1$. In order to ensure the physical consistency of the model, one needs to ensure that $\mathrm{x}_{Ai} + \mathrm{x}_{Bi} \leq 1$ for each vessel $i \in \{1, 2, 3\}$. This has been obtained resorting to the physics-guided loss function design, by including a suitable regularization term in the loss function.

Overall, this simple, qualitative physical information about the plant to be identified has led not only to greater physical consistency of the model, but also to a significant improvement of its accuracy (in particular, the average inaccuracy is more than halved) and to a remarkable reduction of the required training epochs [20].

9.3.3 Open challenges

That of physics-based NN is a very relevant research topic since, as discussed, it allows for safer, more physically consistent, and more accurate models. On the other hand, however, the design of this model is extremely application-dependent. A more systematic approach for design and training would be desirable. Furthermore, it is currently unclear how physics-based RNN models enjoying stability properties, such as those obtained in this thesis for black-box RNN models, can be trained.

9.4 Robust control

Albeit the RNN architectures proposed in this thesis can score a remarkable accuracy, the assumption of no plant-model mismatch, on which the CEP relies, is often very strong and not acceptable. This holds, in general, for any black-box model, e.g., because of the lack of direct correspondence between the states of the models and those of the plant. Although, among model-based control strategies, MPC is often able to compensate for this mismatch and to preserve closed-loop stability, nominal guarantees are inevitably lost.

To address this crucial problem, several robust MPC control strategies have been proposed in the literature for linear systems to ensure closed-loop stability even in presence of bounded disturbances on the state [178, 179]. In this context, owing to its bearable computational burden, tube-based MPC [120] is one of the most popular approaches. In the following, a preliminary application of this control strategy to NNARX models, detailed in [64], is reported.

9.4.1 Tube-based NMPC for NNARX models

In view of their peculiar structure, a robust tube-based NMPC law can be easily designed for NNARX models to cope with the plant-model mismatch in the control synthesis. Indeed, consider the NNARX model (3.5), and assume that the plant, with state $x_{p,k}$, can be described by the system

$$\begin{cases} x_{p,k+1} = Ax_{p,k} + B_u u_k + B_\eta \eta(x_{p,k}, u_k) + B_\eta \delta(x_{p,k}, u_k) \\ y_{p,k} = Cx_{p,k} \end{cases} \tag{9.6}$$

where the bounded uncertainty $\delta(x_{p,k}, u_k)$ affects the components of $x_{p,k+1}$ associated to $y_{p,k+1}$, see (3.5d). Under this assumption, the state error, $e_k = x_{p,k} - x_k$, evolves according to

$$e_{k+1} = Ae_k + B_\eta \Big(\underbrace{\eta(x_{p,k}, u_k) - \eta(x_k, u_k) + \delta(x_{p,k}, u_k)}_{w_k} \Big). \tag{9.7}$$

Under the definition of w_k reported above, the error displays linear dynamics, forced by the bounded term $w_k \in \mathcal{W}$. In the spirit of data-driven control, in [64] it is proposed to estimate the bound $\mathcal{W}$ from the data, i.e., from the free-run simulation error. It is then shown that, by means of this bound, a tube-based NMPC law, similar to the one widely adopted in the linear case [116], that ensures robust asymptotic zero-error output regulation and closed-loop stability can be designed.

9.4.2 Open challenges

The application of tube-based NMPC strategies to other RNN architectures, such as LSTMs and GRUs, remains an open problem and should be investigated. In this regard, as discussed in [20, 22], the main obstacle lies in the design of state observers that guarantee the boundedness of the state estimation error in presence of plant-model mismatch.

9.5 Summary

In this chapter, the main open challenges in the use of RNNs for model-based control design are discussed, and the preliminary research efforts we devoted to these topics have been outlined. Specifically, the issues discussed are the safety verification of the RNN model and its fine-tuning during the plant's lifespan (i.e., the so-called lifelong learning); the exploitation of physics-based knowledge for the design of physics-based RNN models, which allow further performance improvement and better consistency of such model to the underlying physical system; the implementation of robust control laws that are able to guarantee closed-loop stability and performances even in the presence of plant-model mismatch.

CHAPTER 10

Conclusions

In this doctoral thesis, an attempt was made to reconcile deep learning with control theory by building a framework to employ Recurrent Neural Networks (RNNs) for synthesizing model-based control laws that attain closed-loop stability guarantees.

The underlying idea of this thesis has been that, in light of their well-known universal approximation capabilities, RNNs represent good candidates for identifying unknown dynamical systems that display stability-like properties. Therefore, the first part of the thesis was devoted to the formulation of the main RNN architectures as dynamical system in state-space form, and to the analysis of their stability properties. In particular, popular strong stability notions used for nonlinear systems, such as Input-to-State Stability (ISS), Input-to-State Practical Stability (ISPS), and Incremental Input-to-State Stability (δISS), were considered. The stability analysis of the proposed RNN architectures led to sufficient conditions under which the ISS, ISPS, and δISS of these models can be guaranteed. Such conditions come in the form of nonlinear inequalities on the network's weights.

A suitable algorithm, based on the popular truncated back-propagation through time was proposed to train these RNN models, i.e., to learn the weights by which the RNN model best identifies the unknown system.

This training procedure was designed to contextually ensure the ISS, ISPS, or δISS of the trained RNN. Numerical results confirmed the remarkable modeling performances of RNNs for nonlinear system identification, also showing that the enforced stability conditions, despite being only sufficient, do not harm the approximation capabilities of these networks.

The second part of the thesis was devoted to the synthesis of theoretically-sound model-based control strategies designed, under the certainty equivalence principle, based on the trained RNN models. To this end, two approaches were considered, i.e., Nonlinear Model Predictive Control (NMPC) and Internal Model Control (IMC).

In particular, three different NMPC-based schemes were proposed. A first NMPC scheme was proposed for single-layer Gated Recurrent Units (GRU) models, in which the model's δISS was shown to enable the design of a provenly convergent state observer and of a nominally closed-loop stable control law. Notably, such stability guarantee relies on a condition on the NMPC's prediction horizon, rather than on the inclusion of terminal sets and terminal costs as generally done in the context of NMPC.

For the same single-layer GRU model, another NMPC scheme was also devised, which relies on the inclusion of a suitably-tuned integrator on the output tracking error to guarantee asymptotic zero-error output regulation to constant references, as well as nominal closed-loop stability, in the nominal case. In such context, we showed that the model's δISS plays a fundamental role in the design of the components of the control scheme.

A third predictive-control scheme was then proposed for Neural NARX (NNARX) models, for which an NMPC law based on the combination of an integral and a derivative control action was proposed to achieve asymptotic zero-error regulation and nominal closed-loop stability guarantees.

Finally, an alternative control strategy based on IMC was proposed. The main advantage of such control scheme is that, unlike NMPC-based schemes, it involves virtually no online computational cost, since the entire computational burden occurs during the controller synthesis phase. In this context, we have shown that the approximation capabilities of RNNs can be exploited to learn the components of the IMC scheme, namely the system model and a stable inverse of such model. Also for this scheme, the δISS property of said RNNs allowed to attain nominal closed-loop stability guarantees.

The described RNN architectures and the proposed control schemes were tested on several nonlinear benchmark systems. In the considered numerical examples, RNNs confirmed their appeal for the black-box identification of nonlinear systems characterized by stability-like properties. The proposed control laws demonstrated remarkable closed-loop performances, and they proved to attain the stability – and, if the case, also the asymptotic zero-error output regulation – guarantees described in the thesis, even in the non-ideal case of disturbances affecting the plant or moderate inaccuracies of the model.

As discussed in Chapter 9, the use of neural networks for the synthesis of model-based control laws still presents many open challenges. In particular, it is first of all necessary to develop approaches for computing the output reachable sets for RNN models, for their consequent use for safety verification. These models must also be updated and refined over time within the framework of lifelong learning procedures, so that their modeling performances are preserved even in the presence of expected variations in operating conditions. A further major contribution to the interpretability and physical consistency of the model with respect to the modeled plant can be provided by so-called physics-based neural networks. Such approach is deemed increasingly promising and relevant, not only by the control community, but also by the machine learning community. Yet, it needs a systematic approach and theoretical foundations similar to those we here tried to devise for black-box RNN models. Finally, it is important to investigate the application of robust control laws, which can ensure closed-loop stability and performances, not only in the nominal case, but also in presence of model-plant mismatch or disturbances, that should in turn be estimated in a data-based fashion.

Overall, we believe that these challenges can and should be addressed, in order to foster the application of deep learning-based techniques and tools to model-based control design problems. Indeed, as great as these challenges may seem, so great are the possible benefits of their solution to currently unexplored application domains.

Appendices

APPENDIX $\mathcal{A}$

Proofs

A.1 Proof of Chapter 2

A.1.1 Proof of Lemma 2.1

First, let us notice that (2.4), (2.5), and (2.6) share the same structure, i.e.

$$\|\chi_k\|_p \leq \beta(\|\chi_0\|_p, k) + \gamma(\|\nu_{0:k}\|_{p,\infty}) + c, \tag{A.1}$$

where $\chi \in \mathbb{R}^{n_x}$ and $\nu \in \mathbb{R}^{n_u}$. Notice that $c = \varrho$ for ISPS and $c = 0$ for ISS and δISS. The proof consists of two distinct cases, i.e., $q > p$ and $q < p$.

Case $q > p$
Note that, by standard norm arguments, for any vector $v \in \mathbb{R}^n$ and $q > p$, the following chain of inequality holds

$$\|v\|_q \leq \|v\|_p \leq n\|v\|_q \tag{A.2}$$

Hence, (A.1) implies that

$$\|\chi_k\|_q \leq \|\chi_k\|_p \leq \beta(\|\chi_0\|_p, k) + \gamma(\|\nu_{0:k}\|_{p,\infty}) + c. \tag{A.3}$$

Recalling that β and γ is strictly increasing with their (first) argument, they can be upper-bounded as

$$\begin{aligned} \beta(\|\chi_0\|_p, k) &\leq \beta(n_x\|\chi_0\|_q, k) := \tilde{\beta}(\|\chi_0\|_q, k), \\ \gamma(\|\nu_{0:k}\|_{p,\infty}) &\leq \gamma(n_u\|\nu_{0:k}\|_{q,\infty}) := \tilde{\gamma}(\|\nu_{0:k}\|_{q,\infty}). \end{aligned} \tag{A.4}$$

Thus, upper-bounding (A.3) with (A.4) one gets

$$\|\chi_k\|_q \leq \tilde{\beta}(\|\chi_0\|_q, k) + \tilde{\gamma}(\|\nu_{0:k}\|_{q,\infty}) + c, \tag{A.5}$$

which corresponds to the ℓ_q formulation of ISS, ISPS, and δISS.

Case $q < p$

By standard norm arguments, for any vector $v \in \mathbb{R}^n$ and $q < p$, the following chain of inequalities holds

$$\frac{1}{n}\|v\|_q \leq \|v\|_p \leq \|v\|_q \leq n\|v\|_p. \tag{A.6}$$

Therefore, (A.1) implies that

$$\|\chi_k\|_q \leq n_x\|\chi_k\|_p \leq n_x\beta(\|\chi_0\|_p, k) + n_x\gamma(\|\nu_{0:k}\|_{p,\infty}) + n_x\, c. \tag{A.7}$$

In light of the monotonicity of β and γ, these terms can be upper-bounded as

$$\begin{aligned} n_x\beta(\|\chi_0\|_p, k) &\leq n_x\beta(\|\chi_0\|_q, k) := \tilde{\beta}(\|\chi_0\|_q, k), \\ n_x\gamma(\|\nu_{0:k}\|_{p,\infty}) &\leq n_x\gamma(\|\nu_{0:k}\|_{q,\infty}) := \tilde{\gamma}(\|\nu_{0:k}\|_{q,\infty}). \end{aligned} \tag{A.8}$$

Hence, defining $\tilde{c} = n_x\, c$, by upper-bounding (A.1) with (A.8), one gets

$$\|\chi_k\|_q \leq \tilde{\beta}(\|\chi_0\|_q, k) + \tilde{\gamma}(\|\nu_{0:k}\|_{q,\infty}) + \tilde{c}, \tag{A.9}$$

which corresponds to the ℓ_q formulation of ISS, ISPS, and δISS. □

A.1.2 Proof of Lemma 2.2

As done in the proof of Lemma 2.1, we first notice that the conditions (2.7), (2.8), and (2.9) share the following common structure

$$\begin{aligned} \alpha_1(\|\chi_k\|_p) \leq V(\cdot) &\leq \alpha_2(\|\chi_k\|_p) + c_1, \\ \Delta V(\cdot) &\leq -\alpha_3(\|\chi_k\|_p) + \alpha_4(\|\nu_k\|_p) + c_2, \end{aligned} \tag{A.10}$$

where $c_1 = \varrho_1$ and $c_2 = \varrho_2$ in the case of ISPS Lyapunov function and $c_1 = c_2 = 0$ in the case of ISS and δISS Lyapunov functions. Moreover, $\chi_k \in \mathbb{R}^{n_x}$ and $\nu_k \in \mathbb{R}^{n_u}$. The two cases $q > p$ and $q < p$ are now discussed.

Case $q > p$
Recalling the norms' chain of inequality (A.2), since α_1, α_2, α_3, and α_4 are of class $\mathcal{K}_\infty$, and hence strictly increasing with their arguments, it holds that

$$\begin{aligned} \alpha_1(\|\chi_k\|_p) &\geq \alpha_1(\|\chi_k\|_q) := \tilde{\alpha}_1(\|\chi_k\|_q), \\ \alpha_2(\|\chi_k\|_p) &\leq \alpha_2(n_x\|\chi_k\|_q) := \tilde{\alpha}_2(\|\chi_k\|_q), \\ -\alpha_3(\|\chi_k\|_p) &\leq -\alpha_3(\|\chi_k\|_q) := -\tilde{\alpha}_3(\|\chi_k\|_q), \\ \alpha_4(\|\nu_k\|_p) &\leq \alpha_4(n_u\|\nu_k\|_q) := \tilde{\alpha}_4(\|\nu_k\|_q). \end{aligned} \tag{A.11}$$

By bounding (A.10) with (A.11), one gets

$$\begin{aligned} \tilde{\alpha}_1(\|\chi_k\|_q) \leq V(\cdot) &\leq \tilde{\alpha}_2(\|\chi_k\|_q) + c_1, \\ \Delta V(\cdot) &\leq -\tilde{\alpha}_3(\|\chi_k\|_q) + \tilde{\alpha}_4(\|\nu_k\|_q) + c_2, \end{aligned} \tag{A.12}$$

which means that $V(\cdot)$ is also an ℓ_q Lyapunov function.

Case $q < p$
In light of (A.6), and once more exploiting the strict monotonicity of α_1, α_2, α_3, and α_4, it holds that

$$\begin{aligned} \alpha_1(\|x_k\|_p) &\geq \alpha_1\Big(\frac{1}{n_x}\|x_k\|_q\Big) := \tilde{\alpha}_1(\|x_k\|_q) \\ \alpha_2(\|x_k\|_p) &\leq \alpha_2(\|x_k\|_q) := \tilde{\alpha}_2(\|x_k\|_q) \\ -\alpha_3(\|x_k\|_p) &\leq -\alpha_3\Big(\frac{1}{n_x}\|x_k\|_q\Big) := -\tilde{\alpha}_3(\|x_k\|_q) \\ \alpha_4(\|u_k\|_p) &\leq \alpha_4(\|u_k\|_q) := \tilde{\alpha}_4(\|u_k\|_q) \end{aligned} \tag{A.13}$$

Applying the bounds reported in (A.13) to (A.10), we get

$$\begin{aligned} \tilde{\alpha}_1(\|\chi_k\|_q) \leq V(\cdot) &\leq \tilde{\alpha}_2(\|\chi_k\|_q) + c_1, \\ \Delta V(\cdot) &\leq -\tilde{\alpha}_3(\|\chi_k\|_q) + \tilde{\alpha}_4(\|\nu_k\|_q) + c_2, \end{aligned} \tag{A.14}$$

which implies that $V(\cdot)$ is an ℓ_q Lyapunov function. □

A.1.3 Proof of Proposition 2.4

In view of the Lipschitz continuity of the output transformation $g(x_k)$, there exists $L_g > 0$ and $\varrho_g \geq 0$ such that, for any $x \in \mathcal{X}$, $\|y\|_p \leq L_g\|x\|_p + \varrho_g$. Therefore, in light of the ISPS condition (2.4), it asymptotically holds that

$$\|y_k\|_p \leq \gamma_y(\|u_{0:k}\|_{p,\infty}) + \varrho_y,$$

where $\gamma_y(\cdot) = L_g\gamma_x(\cdot)$ and $\varrho_y = \varrho_g + \varrho$. □

A.1.4 Proof of Theorem 2.1

The proof is structured as follows: first, the bound of the linearization error is characterized; then, the exponential δISS property is shown to entail the existence of a quadratic Lyapunov function for the nonlinear system (2.1); finally, such Lyapunov function is shown to be a local ISS Lyapunov function for the linearized system. For the sake of simplicity, without loss of generality, the ℓ_2 norm is herein considered (see Lemma 2.1).

Exponential δISS property
The exponential δISS property implies the existence of $\mu > 0$ and $\lambda \in (0, 1)$ such that

$$\|\delta x_k\|_2 \leq \mu \|\delta x_0\|_2 \lambda^k + \gamma(\|\delta u_{0:k}\|_{2,\infty}).$$

Owing to the boundedness of $\delta u_{0:k}$, which is entailed by $u_{0:k} \in \mathcal{U}_{0:k}$, there exists $\bar{\gamma} > 0$ such that the exponential δISS property reads as

$$\|\delta x_k\|_2 \leq \mu \|\delta x_0\|_2 \lambda^k + \bar{\gamma} \|\delta u_{0:k}\|_{2,\infty}. \tag{A.15}$$

Consider now initial states $\delta x_0 \in \mathcal{D}_{x_0}$ and $\delta u_\tau \in \mathcal{D}_u$, for all $\tau \in \mathbb{Z}_{\geq 0}$, where

$$\begin{aligned} \mathcal{D}_{x_0}(r_{x_0}) &= \{\delta x_0 \in \mathbb{R}^{n_x} : \|\delta x_0\|_2 \leq r_{x_0} \wedge \bar{x} + \delta x_0 \in \mathcal{X}\}, \\ \mathcal{D}_u(r_u) &= \{\delta u \in \mathbb{R}^{n_u} : \|\delta u\|_2 \leq r_u \wedge \bar{u} + \delta u \in \mathcal{U}\}. \end{aligned} \tag{A.16}$$

Condition (A.15) then implies that, for any $\tau \in \mathbb{Z}_{\geq 0}$, $\delta x_\tau \in \mathcal{D}_x$, where

$$\mathcal{D}_x(r_{x_0}, r_u) = \{\delta x \in \mathbb{R}^{n_x} : \|\delta x_\tau\|_2 \leq \mu r_{x_0} + \bar{\gamma} r_u\}. \tag{A.17}$$

Characterization of the linearization error
Let $i \in \{1, ..., n_x\}$ denote the i-th state vector component. In light of the Mean Value Theorem, there exist

$$\begin{aligned} \hat{x} &\in \left\{v \in \mathbb{R}^n : \min([\bar{x}]_i, [\bar{x}]_i + [\delta x_k]_i) \leq [v]_i \leq \max([\bar{x}]_i, [\bar{x}]_i + [\delta x_k]_i)\right\} \\ \hat{u} &\in \left\{v \in \mathbb{R}^m : \min([\bar{u}]_i, [\bar{u}]_i + [\delta u_k]_i) \leq [v]_i \leq \max([\bar{u}]_i, [\bar{u}]_i + [\delta u_k]_i)\right\} \end{aligned}$$

such that

$$\begin{aligned} f_i(\bar{x}+\delta x_k, \bar{u}+\delta u_k) - f_i(\bar{x},\bar{u}) &= \left.\frac{\partial f_i}{\partial x}\right|_{\hat{x},\hat{u}} \delta x_k + \left.\frac{\partial f_i}{\partial u}\right|_{\hat{x},\hat{u}} \delta u_k \\ &= \left.\frac{\partial f_i}{\partial x}\right|_{\bar{x},\bar{u}} \delta x_k + \left.\frac{\partial f_i}{\partial u}\right|_{\bar{x},\bar{u}} \delta u_k \\ &\quad + \left[\left.\frac{\partial f_i}{\partial x}\right|_{\hat{x},\hat{u}} - \left.\frac{\partial f_i}{\partial x}\right|_{\bar{x},\bar{u}}\right] \delta x_k \\ &\quad + \left[\left.\frac{\partial f_i}{\partial u}\right|_{\hat{x},\hat{u}} - \left.\frac{\partial f_i}{\partial u}\right|_{\bar{x},\bar{u}}\right] \delta u_k \end{aligned} \tag{A.18}$$

where f_i indicates the i-th component of $f(x,u)$. Equivalently, letting $A_{\delta i}$, $B_{\delta i}$, and $\varepsilon_i(\delta x_k, \delta u_k)$ denote the i-th row of A_δ, B_δ, and of the linearization error, respectively, (A.18) can be rewritten as

$$\begin{aligned} \delta x_{k+1} &= f_i(\bar{x}+\delta x_k, \bar{u}+\delta u_k) - f_i(\bar{x},\bar{u}) \\ &= A_{\delta i}\, \delta x_k + B_{\delta i}\, \delta u_k + \varepsilon_i(\delta x_k, \delta u_k), \end{aligned} \tag{A.19a}$$

where

$$\varepsilon_i(\delta x_k, \delta u_k) = \underbrace{\left[\left.\frac{\partial f_i}{\partial x}\right|_{\hat{x},\hat{u}} - \left.\frac{\partial f_i}{\partial x}\right|_{\bar{x},\bar{u}}\right] \delta x_k}_{\varepsilon_{x,i}(\delta x_k, \delta u_k)} + \underbrace{\left[\left.\frac{\partial f_i}{\partial u}\right|_{\hat{x},\hat{u}} - \left.\frac{\partial f_i}{\partial u}\right|_{\bar{x},\bar{u}}\right] \delta u_k}_{\varepsilon_{u,i}(\delta x_k, \delta u_k)} . \tag{A.19b}$$

Owing to the smoothness of f, over $\mathcal{D}_x$ and $\mathcal{D}_u$ there exist constants $L_{1,i} \geq 0$ and $L_{2,i} \geq 0$ such that $\varepsilon_{x,i}$ can be bounded as

$$\begin{aligned} \|\varepsilon_{x,i}(\delta x_k, \delta u_k)\|_2^2 &\leq \left\| \left.\frac{\partial f_i}{\partial x}\right|_{\hat{x},\hat{u}} - \left.\frac{\partial f_i}{\partial x}\right|_{\bar{x},\bar{u}} \right\|_2^2 \|\delta x_k\|_2^2 \\ &\leq \left[\left\| \left.\frac{\partial f_i}{\partial x}\right|_{\hat{x},\hat{u}} - \left.\frac{\partial f_i}{\partial x}\right|_{\bar{x},\hat{u}} \right\|_2^2 + \left\| \left.\frac{\partial f_i}{\partial x}\right|_{\bar{x},\hat{u}} - \left.\frac{\partial f_i}{\partial x}\right|_{\bar{x},\bar{u}} \right\|_2^2 \right] \|\delta x_k\|_2^2 \\ &\leq \left[L_{1,i}^2 \|\delta x_k\|_2^2 + L_{2,i}^2 \|\delta u_k\|_2^2 \right] \|\delta x_k\|_2^2. \end{aligned} \tag{A.20a}$$

Similarly, there exist constants $L_{3,i} \geq 0$ and $L_{4,i} \geq 0$ such that

$$\begin{aligned}
\|\varepsilon_{u,i}(\delta x_k, \delta u_k)\|_2^2 &\leq \left\| \frac{\partial f_i}{\partial u}\bigg|_{\hat{x},\hat{u}} - \frac{\partial f_i}{\partial u}\bigg|_{\bar{x},\bar{u}} \right\|_2^2 \|\delta u_k\|_2^2 \\
&\leq \left[\left\| \frac{\partial f_i}{\partial u}\bigg|_{\hat{x},\hat{u}} - \frac{\partial f_i}{\partial u}\bigg|_{\bar{x},\hat{u}} \right\|_2^2 + \left\| \frac{\partial f_i}{\partial u}\bigg|_{\bar{x},\hat{u}} - \frac{\partial f_i}{\partial u}\bigg|_{\bar{x},\bar{u}} \right\|_2^2 \right] \|\delta u_k\|_2^2 \\
&\leq \left[L_{3,i}^2 \|\delta x_k\|_2^2 + L_{4,i}^2 \|\delta u_k\|_2^2 \right] \|\delta u_k\|_2^2.
\end{aligned} \tag{A.20b}$$

Recalling that $\varepsilon_x(\delta x_k, \delta u_k) = \left[\varepsilon_{x,1}(\delta x_k, \delta u_k), ..., \varepsilon_{x,n_x}(\delta x_k, \delta u_k)\right]'$, and defining $L_1^2 = \max_i L_{1,i}^2$ and $L_2^2 = \max_i L_{2,i}^2$, the following chain of inequalities holds

$$\begin{aligned}
\|\varepsilon_x(\delta x_k, \delta u_k)\|_2 &\leq \sqrt{\sum_{i=1}^{n_x} \left\|\varepsilon_{x,i}(\delta x_k, \delta u_k)\right\|^2} \\
&\leq \sqrt{n_x}\, \|\delta x_k\|_2 \sqrt{L_1^2 \|\delta x_k\|_2^2 + L_2^2 \|\delta u_k\|_2^2} \\
&\leq \sqrt{n_x} L_1 \|\delta x_k\|_2^2 + \sqrt{n_x}\, L_2 \|\delta x_k\|_2 \|\delta u_k\|_2.
\end{aligned} \tag{A.21a}$$

Analogously, letting $\varepsilon_u(\delta x_k, \delta u_k) = \left[\varepsilon_{u,1}(\delta x_k, \delta u_k), ..., \varepsilon_{u,n_x}(\delta x_k, \delta u_k)\right]'$, the following chain of inequality holds

$$\begin{aligned}
\|\varepsilon_u(\delta x_k, \delta u_k)\|_2 &\leq \sqrt{\sum_{i=1}^{n_x} \left\|\varepsilon_{u,i}(\delta x_k, \delta u_k)\right\|_2^2} \\
&\leq \sqrt{n_x}\, \|\delta u_k\|_2 \sqrt{L_3^2 \|\delta x_k\|_2^2 + L_4^2 \|\delta u_k\|_2^2} \\
&\leq \sqrt{n_x}\, L_3 \|\delta x_k\|_2 \|\delta u_k\|_2 + \sqrt{n_x}\, L_4 \|\delta u_k\|_2^2.
\end{aligned} \tag{A.21b}$$

Therefore, there exist $L_{\varepsilon x} \geq 0$ and $L_{\varepsilon u} \geq 0$ such that the linearization error can be bounded as

$$\begin{aligned}
\|\varepsilon(\delta x_k, \delta u_k)\|_2 &\leq \|\varepsilon_x(\delta x_k, \delta u_k)\|_2 + \|\varepsilon_u(\delta x_k, \delta u_k)\|_2 \\
&\leq \sqrt{n_x} L_1 \|\delta x_k\|_2^2 + \sqrt{n_x}(L_2 + L_3) \|\delta x_k\|_2 \|\delta u_k\|_2 \\
&\qquad + \sqrt{n_x} L_4 \|\delta u_k\|_2^2 \\
&\leq L_{\varepsilon x} \|\delta x_k\|_2^2 + L_{\varepsilon u} \|\delta u_k\|_2^2.
\end{aligned} \tag{A.22}$$

Lyapunov function definition

Since the system is exponentially δISS, Theorem 5.8 of [180] can be invoked to guarantee the existence of a (δISS) Lyapunov function that satisfies the following conditions

$$c_1\|\delta x_k\|_2^2 \leq V(\delta x_k) \leq c_2\|\delta x_k\|_2^2, \tag{A.23a}$$

$$V(A_\delta \delta x_k + B_\delta \delta u_k + \varepsilon(\delta x_k, \delta u_k)) - V(\delta x_k) \leq -c_3\|\delta x_k\|_2^2 + \sigma_3(\|\delta u_k\|_2), \tag{A.23b}$$

$$|V(x) - V(y)| \leq c_4(\|x\|_2 + \|y\|_2)\,\|x-y\|_2 \tag{A.23c}$$

where the scalars $c_1, c_2, c_3, c_4 > 0$ and $\sigma_3 \in \mathcal{K}_\infty$. We now show that, in a sufficiently small neighborhood of the equilibrium (i.e. for sufficiently small values of r_{x0} and r_u), (A.23) constitutes a suitable ISS Lyapunov function for the linearized system (2.15). By adding and subtracting the term $V(A_\delta \delta x_k + B_\delta \delta u_k)$ to the left-hand side of (A.23b) we get

$$\begin{aligned} &V(A_\delta \delta x_k + B_\delta \delta u_k) - V(\delta x_k) + \\ &\quad + \big[V(A_\delta \delta x_k + B_\delta \delta u_k + \varepsilon(\delta x_k, \delta u_k)) - V(A_\delta \delta x_k + B_\delta \delta u_k)\big] \\ &\qquad \leq -c_3\|\delta x_k\|_2^2 + \sigma_3(\|\delta u_k\|_2). \end{aligned} \tag{A.24}$$

Let us notice that, in light of (A.17), (A.22), and (A.23c), it holds that

$$\begin{aligned} &\big\|V(A_\delta \delta x_k + B_\delta \delta u_k + \varepsilon(\delta x_k, \delta u_k)) - V(A_\delta \delta x_k + B_\delta \delta u_k)\big\|_2 \\ &\quad \leq c_4\Big(\big\|A_\delta \delta x_k + B_\delta \delta u_k + \varepsilon(\delta x_k, \delta u_k)\big\|_2 + \big\|A_\delta \delta x_k + B_\delta \delta u_k\big\|_2\Big)\|\varepsilon(\delta x_k, \delta u_k)\|_2 \\ &\quad \leq c_4\Big(2\big\|A_\delta \delta x_k + B_\delta \delta u_k + \varepsilon(\delta x_k, \delta u_k)\big\|_2 + \|\varepsilon(\delta x_k, \delta u_k)\|_2\Big)\|\varepsilon(\delta x_k, \delta u_k)\|_2 \\ &\quad \leq c_4\Big(2\mu r_{x0} + 2\bar{\gamma} r_u + L_{\varepsilon x}\|\delta x_k\|_2^2 + L_{\varepsilon u}\|\delta u_k\|_2^2\Big)\big(L_{\varepsilon x}\|\delta x_k\|_2^2 + L_{\varepsilon u}\|\delta u_k\|_2^2\big) \end{aligned} \tag{A.25}$$

Since in a neighborhood of the equilibrium $\|\delta x_k\|_2^2 \ll \|\delta x_k\|_2 \leq r_x$ and $\|\delta u_k\|_2^2 \ll \|\delta u_k\|_2 \leq r_u$, there exist $\mu_{x_0} > 0$ and $\mu_u > 0$ such that

$$\begin{aligned} &\big\|V(A_\delta \delta x_k + B_\delta \delta u_k + \varepsilon(\delta x_k, \delta u_k)) - V(A_\delta \delta x_k + B_\delta \delta u_k)\big\|_2 \\ &\qquad \leq c_4\big(\mu_{x_0} r_{x_0} + \mu_u r_u\big)\big(L_{\varepsilon x}\|\delta x_k\|_2^2 + L_{\varepsilon u}\|\delta u_k\|_2^2\big). \end{aligned} \tag{A.26}$$

In light (A.26), from (A.24) one can obtain

$$\begin{aligned}
&V(A_\delta \delta x_k + B_\delta \delta u_k) - V(\delta x_k)\\
&\qquad \le -c_3\|\delta x_k\|_2^2 + \sigma_3(\|\delta u_k\|_2)\\
&\qquad\qquad - \left[V(A_\delta \delta x_k + B_\delta \delta u_k + \varepsilon(\delta x_k, \delta u_k)) - V(A_\delta \delta x_k + B_\delta \delta u_k)\right]\\
&\qquad \le -c_3\|\delta x_k\|_2^2 + \sigma_3(\|\delta u_k\|_2)\\
&\qquad\qquad + c_4\big(\mu_{x0} r_{x0} + \mu_u r_u\big)\big(L_{\varepsilon x}\|\delta x_k\|_2^2 + L_{\varepsilon u}\|\delta u_k\|_2^2\big)\\
&\qquad \le -\big[c_3 - c_4(\mu_{x0} r_{x0} + \mu_u r_u) L_{\varepsilon x}\big]\|\delta x_k\|_2^2\\
&\qquad\qquad + \sigma_3(\|\delta u_k\|_2) + c_4(\mu_{x0} r_{x0} + \mu_u r_u) L_{\varepsilon u}\|\delta u_k\|_2^2\\
&\qquad \le -\tilde{c}_3\|\delta x_k\|_2^2 + \tilde{\sigma}_3(\|\delta u_k\|_2),
\end{aligned} \tag{A.27}$$

where $\tilde{c}_3 = c_3 - c_4(\mu_{x0} r_{x0} + \mu_u r_u) L_{\varepsilon x}$ and $\tilde{\sigma}_3(\|\delta u_k\|_2) = \sigma_3(\|\delta u_k\|_2) + c_4\big(\mu_{x0} r_{x0} + \mu_u r_u\big)\big(L_{\varepsilon x}\|\delta x_k\|^2 + L_{\varepsilon u}\|\delta u_k\|_2^2\big)$. Notice that $\tilde{\sigma}_3 \in \mathcal{K}_\infty$.

Then, there exist sufficiently small r_u and r_{x0} such that $\tilde{c}_3 > 0$, which in turn imply that $V(\delta x_k)$ is a local ISS Lyapunov function for the the linearized system in $\mathcal{D}_x$ and $\mathcal{D}_u$. Hence, the linearized system is locally ISS. Since for linear system the ISS property implies the asymptotic stability of the origin, A_δ is Schur stable. □

A.2 Proofs of Chapter 3

A.2.1 Proof of Theorem 3.1

Define $P = \text{diag}(I_{n_z,n_z}, 2 \cdot I_{n_z,n_z}, ..., H \cdot I_{n_z,n_z})$. Recalling (3.5b), it is easy to see that P is the solution to the Lyapunov equation $A'PA - P = -Q$, where $Q = I$. Let therefore $V(x) = x'Px = \|x\|_P^2$ be a candidate ℓ_2-ISPS Lyapunov function. Since the minimum and maximum singular values of P are $\underline{\varsigma}_P = 1$ and $\bar{\varsigma}_P = H$, respectively, it holds that

$$\|x_k\|_2^2 \le V(x_k) \le H\|x_k\|_2^2. \tag{A.28}$$

In light of (3.5a) it holds that

$$\begin{aligned}
V(x_{k+1}) - V(x_k) &= x_{k+1}' P x_{k+1} - x_k' P x_k\\
&= x_k'(A'PA - P)x_k + u_k' B_u' P B_u u_k + 2x_k' A' P B_u u_k\\
&\quad + \eta(x_k, u_k)' B_\eta' P B_\eta \eta(x_k, u_k) + 2x_k' A' P B_\eta \eta(x_k, u_k)\\
&\quad + 2u_k' B_u' P B_\eta \eta(x_k, u_k).
\end{aligned} \tag{A.29}$$

Owing to the structure of A, B_u and B_η, see (3.5b), and being P block-diagonal, it follows that

$$A'PB_u = 0_{n_x,n_u}, \tag{A.30a}$$
$$A'PB_\eta = 0_{n_x,n_y}, \tag{A.30b}$$
$$B_u'PB_u = HI_{n_u,n_u}, \tag{A.30c}$$
$$B_\eta'PB_\eta = HI_{n_y,n_y}, \tag{A.30d}$$
$$B_u'PB_\eta = H\tilde{B}_u'\tilde{B}_\eta = 0_{n_u,n_y}. \tag{A.30e}$$

Combining (A.29) and (A.30) one can hence obtain

$$V(x_{k+1}) - V(x_k) = -x_k'x_k + Hu_k'u_k + H\eta(x_k,u_k)'\eta(x_k,u_k) \tag{A.31}$$

Let us now point out that, since the activation functions ψ_l are Lipschitz-continuous, η is Lipschitz-continuous as well. Hence, letting

$$b = [b^{(0)\prime}, ..., b^{(L)\prime}]'$$

be the concatenation of bias vectors, by standard norms arguments, for any scalar $q \neq 0$ it holds that

$$\begin{aligned} \|\eta(x_k,u_k)\|_2^2 \leq & \Big(1+\frac{1}{q^2}\Big)K_x^2\|x_k\|_2^2 + 2(1+q^2)K_u^2\|u_k\|_2^2 \\ & + 2(1+q^2)K_b^2\|b\|_2^2 \end{aligned} \tag{A.32a}$$

where the coefficients K_x, K_u, and K_b are defined as

$$\begin{aligned} K_x =& \|U^{(0)}\|_2 \prod_{l=1}^{L} L_\psi^{(l)} \|U^{(l)}\|_2, \\ K_u =& \|U^{(0)}\|_2 \sum_{l=1}^{L} \bigg(\prod_{j=l+1}^{L} L_\psi^{(j)} \|U^{(j)}\|_2 \bigg) L_\psi^{(l)} \|W^{(l)}\|_2, \\ K_b =& \|U^{(0)}\|_2 \sum_{l=1}^{L} \bigg(\prod_{j=l+1}^{L} L_\psi^{(j)} \|U^{(j)}\|_2 \bigg) L_\psi^{(l)}. \end{aligned} \tag{A.32b}$$

From (A.32) and (A.31) it follows that

$$\begin{aligned} V(x_{k+1}) - V(x_k) \leq & - \Big[1 - \Big(1+\frac{1}{q^2}\Big)HK_x^2\Big]\|x_k\|_2^2 \\ & + H\Big[1 + 2(1+q^2)K_u^2\Big]\|u_k\|_2^2 \\ & + 2(1+q^2)HK_b^2\|b\|_2^2. \end{aligned} \tag{A.33}$$

We point out that V is an ISPS Lyapunov function if the coefficient multiplying $\|x_k\|_2^2$ is strictly negative, i.e.

$$1 - \frac{q^2+1}{q^2} H \|U^{(0)}\|_2^2 \prod_{l=1}^{L} \left(L_\psi^{(l)}\right)^2 \|U^{(l)}\|_2^2 > 0$$

or, equivalently,

$$\prod_{l=0}^{L} \left\|U^{(l)}\right\|_2 - \sqrt{\frac{q^2}{q^2+1}} \frac{1}{\left(\prod_{l=1}^{L} L_\psi^{(l)}\right)\sqrt{H}} < 0. \tag{A.34}$$

To show that (A.34) holds, let us point out that, in light of (3.6), by continuity argument there exists a sufficiently small $\varepsilon > 0$ such that

$$\prod_{l=0}^{L} \left\|U^{(l)}\right\|_2 - (1-\varepsilon) \frac{1}{\left(\prod_{l=1}^{L} L_\psi^{(l)}\right)\sqrt{H}} < 0. \tag{A.35}$$

Thus, there exists $\underline{q}$ large enough such that, $1-\varepsilon \leq \sqrt{\frac{q^2}{q^2+1}}$ for any $q \geq \underline{q}$, so that

$$\begin{aligned} \prod_{l=0}^{L} \left\|U^{(l)}\right\|_2 &- \sqrt{\frac{q^2}{q^2+1}} \frac{1}{\left(\prod_{l=1}^{L} L_\psi^{(l)}\right)\sqrt{H}} \\ &\leq \prod_{l=0}^{L} \left\|U^{(l)}\right\|_2 - (1-\varepsilon) \frac{1}{\left(\prod_{l=1}^{L} L_\psi^{(l)}\right)\sqrt{H}} \\ &< 0. \end{aligned} \tag{A.36}$$

Owing to (A.36), one can hence guarantee the existence of some $\delta > 0$ such that

$$\begin{aligned} V(x_{k+1}) - V(x_k) \leq &- \delta \|x_k\|_2^2 + H\left[1 + 2(1+q^2)K_u^2\right] \|u_k\|_2^2 \\ &+ 2(1+q^2) H K_b^2 \|b\|_2^2. \end{aligned} \tag{A.37}$$

Given the inequalities (A.28) and (A.37), according to Definition 2.7 V is an ISPS Lyapunov function, with

$$\alpha_1(\|x_k\|_2) = \|x_k\|_2, \tag{A.38a}$$

$$\alpha_2(\|x_k\|_2) = H\|x_k\|_2, \tag{A.38b}$$

$$\alpha_3(\|x_k\|_2) = \delta\|x_k\|_2^2, \tag{A.38c}$$

$$\alpha_4(\|u_k\|_2) = H\big[1 + 2(1+q^2)K_u^2\big]\|u_k\|_2^2, \tag{A.38d}$$

$$\varrho_1 = 0, \tag{A.38e}$$

$$\varrho_2 = 2(1+q^2)HK_b^2\|b\|_2^2. \tag{A.38f}$$

By Proposition 2.2, the existence of an ISPS Lyapunov function implies that the system is ISPS, which concludes the proof. □

A.2.2 Proof of Theorem 3.2

To prove the Theorem, we show the existence of an ℓ_2-δISS Lyapunov function consistent with Definition 2.9. To this end, let $x_{a,k}$ and $x_{b,k}$ be two generic states, and consider the two generic inputs $u_{a,k}$ and $u_{b,k}$. We denote $x_{a,k+1} = f(x_{a,k}, u_{a,k})$ and $x_{b,k+1} = f(x_{b,k}, u_{b,k})$, where $f(x,u)$ is the state function of the NNARX model (3.5).

Consider $V(x_a, x_b) = (x_a - x_b)'P(x_a - x_b) = \|x_a - x_b\|_P^2$ as a candidate δISS Lyapunov function, where P is the solution to the Lyapunov equation $A'PA - P = -Q$, with $Q = I$. Then, it is easy to verify that P is a symmetric block-diagonal matrix, $P = \text{diag}(I_{n_z,n_z}, 2\cdot I_{n_z,n_z}, ..., H \cdot I_{n_z,n_z})$. Since the minimum and maximum singular value of P are $\underline{\varsigma}_P = 1$ and $\bar{\varsigma}_P = H$, $V(x_a, x_b)$ can be bounded as follows

$$\|x_{a,k} - x_{b,k}\|_2^2 \le V_\delta(x_{a,k}, x_{b,k}) \le H\|x_{a,k} - x_{b,k}\|_2^2. \tag{A.39}$$

Moreover, it holds that

$$\begin{aligned} &V_\delta(x_{a,k+1}, x_{b,k+1}) - V_\delta(x_{a,k}, x_{b,k}) \\ &\quad = \big[Ax_{a,k} + B_u u_{a,k} + B_\eta \eta(x_{a,k}, u_{a,k}) - Ax_{b,k} - B_u u_{b,k} - B_\eta \eta(x_{b,k}, u_{b,k})\big]' \cdot \\ &\qquad \cdot P \cdot \big[Ax_{a,k} + B_u u_{a,k} + B_\eta \eta(x_{a,k}, u_{a,k}) - Ax_{b,k} - B_u u_{b,k} - B_\eta \eta(x_{b,k}, u_{b,k})\big] \\ &\qquad - (x_{a,k} - x_{b,k})'P(x_{a,k} - x_{b,k}) \end{aligned} \tag{A.40}$$

Owing to (A.30), the previous equality can be re-written as

$$\begin{aligned} &V_\delta(x_{a,k+1}, x_{b,k+1}) - V_\delta(x_{a,k}, x_{b,k}) \\ &\quad = (x_{a,k} - x_{b,k})'(A'PA - P)(x_{a,k} - x_{b,k}) \\ &\qquad + (u_{a,k} - u_{b,k})'B_u'PB_u(u_{a,k} - u_{b,k}) \\ &\qquad + \big[\eta(x_{a,k}, u_{a,k}) - \eta(x_{b,k}, u_{b,k})\big]'B_\eta'PB_\eta\big[\eta(x_{a,k}, u_{a,k}) - \eta(x_{b,k}, u_{b,k})\big] \end{aligned} \tag{A.41}$$

By summing and subtracting $\eta(x_{b,k}, u_{a,k})$ to both the square brackets of the last term of (A.41), and applying standard norm arguments, it holds that

$$\begin{aligned}
&\big[\eta(x_{a,k}, u_{a,k}) - \eta(x_{b,k}, u_{b,k}) \pm \eta(x_{b,k}, u_{a,k})\big]' B_\eta' P \cdot \\
&\quad \cdot B_\eta \big[\eta(x_{a,k}, u_{a,k}) - \eta(x_{b,k}, u_{b,k}) \pm \eta(x_{b,k}, u_{a,k})\big] \\
&\qquad \leq H \Big\| \big(\eta(x_{a,k}, u_{a,k}) - \eta(x_{b,k}, u_{a,k})\big) + \big(\eta(x_{b,k}, u_{a,k}) - \eta(x_{b,k}, u_{b,k})\big) \Big\|_2^2 \\
&\qquad \leq H \Big(1 + \frac{1}{q^2}\Big) \|\eta(x_{a,k}, u_{a,k}) - \eta(x_{b,k}, u_{a,k})\|_2^2 \\
&\qquad\qquad + H\big(1 + q^2\big) \|\eta(x_{b,k}, u_{a,k}) - \eta(x_{b,k}, u_{b,k})\|_2^2
\end{aligned} \tag{A.42}$$

for any scalar $q \neq 0$. Then, in light of the Lipschitzianity of the activation functions, $\eta(x, u)$ is also Lipschitz, and specifically

$$\begin{aligned}
\|\eta(x_{a,k}, u_{a,k}) - \eta(x_{b,k}, u_{a,k})\|_2^2 &\leq K_x^2 \|x_{a,k} - x_{b,k}\|_2^2, \\
\|\eta(x_{b,k}, u_{a,k}) - \eta(x_{b,k}, u_{b,k})\|_2^2 &\leq K_u^2 \|u_{a,k} - u_{b,k}\|_2^2,
\end{aligned} \tag{A.43}$$

where K_x and K_u are defined as in (A.32b).

In light of (A.42) and (A.43), recalling that $A'PA - P = -Q = -I$, it holds that

$$\begin{aligned}
V_\delta(x_{a,k+1}, x_{b,k+1}) - V_\delta(x_{a,k}, x_{b,k}) \leq &- \Big[1 - \Big(1 + \frac{1}{q^2}\Big) H K_x^2\Big] \|x_{a,k} - x_{b,k}\|_2^2 \\
&+ H\Big[1 + (1 + q^2) K_u^2\Big] \|u_{a,k} - u_{b,k}\|_2^2.
\end{aligned} \tag{A.44}$$

Therefore, if the coefficient multiplying $\|x_{a,k} - x_{b,k}\|_2^2$ is negative, i.e.

$$1 - \Big(1 + \frac{1}{q^2}\Big) H K_x^2 > 0, \tag{A.45}$$

for some value of q, then V_δ is a δISS Lyapunov function. Recalling (A.32b), by minor manipulations, (A.45) is equivalent to

$$\prod_{l=0}^{L} \big\|U^{(l)}\big\|_2 - \sqrt{\frac{q^2}{q^2 + 1}} \frac{1}{\big(\prod_{l=1}^{L} L_\psi^{(l)}\big)\sqrt{H}} < 0. \tag{A.46}$$

As discussed in the proof of Theorem 3.1 (see Appendix A.2.1), since by assumption condition (3.6) is satisfied, it can be shown that (A.46) holds for sufficiently large values of q. Thus, there exists a scalar $\delta > 0$ such that

(A.44) can be bounded as

$$\begin{aligned} V_\delta(x_{a,k+1}, x_{b,k+1}) - V_\delta(x_{a,k}, x_{b,k}) \leq & - \delta \|x_{a,k} - x_{b,k}\|_2^2 \\ & + H\Big[1 + (1+q^2)K_u^2\Big] \|u_{a,k} - u_{b,k}\|_2^2 \end{aligned} \tag{A.47}$$

In light of (A.39) and (A.47), and according to Definition 2.9, V_δ is a δISS Lyapunov function, with functions

$$\alpha_1(\|x_{a,k} - x_{b,k}\|_2) = \|x_{a,k} - x_{b,k}\|_2^2, \tag{A.48a}$$

$$\alpha_2(\|x_{a,k} - x_{b,k}\|_2) = H\|x_{a,k} - x_{b,k}\|_2^2, \tag{A.48b}$$

$$\alpha_3(\|x_{a,k} - x_{b,k}\|_2) = \delta\|x_{a,k} - x_{b,k}\|_2^2, \tag{A.48c}$$

$$\alpha_4(\|u_{a,k} - u_{b,k}\|_2) = H\Big[1 + (1+q^2)K_u^2\Big] \|u_{a,k} - u_{b,k}\|_2^2. \tag{A.48d}$$

Invoking Proposition 2.3, the δISS of the NNARX model is thus proven. □

A.2.3 Proof of Lemma 3.1

First and foremost, let us notice that h_k is surely unity-bounded. Indeed, from (3.8a) it follows that the j-th component is $[h_k]_j = [z_k]_j \big[\phi(c_{k+1})\big]_j$, where, in light of the boundedness of σ and ϕ, it holds $|[z_k]_j| < 1$ and $\Big|\big[\phi(c_{k+1})\big]_j\Big| < 1$. Hence,

$$\|h_k\|_\infty \leq 1. \tag{A.49}$$

Now we show that (3.9a) holds. Recalling Assumption 3.1, noticing that σ is strictly increasing with its argument and that, by definition of infinity norm,

$$|[v]_j| \leq \|v\|_\infty,$$

the following chain of inequality holds

$$\begin{aligned} \Big|\big[\sigma(W_f u_k + U_f c_k + b_f)\big]_j\Big| & \leq \|\sigma(W_f u_k + U_f c_k + b_f)\|_\infty \\ & \leq \max_{u \in \mathcal{U}, h: \|h\|_\infty \leq 1} \|\sigma(W_f u + U_f h + b_f)\|_\infty \\ & \leq \Big\| \max_{u \in \mathcal{U}, h: \|h\|_\infty \leq 1} \sigma(W_f u + U_f h + b_f) \Big\|_\infty \\ & \leq \Big\| \sigma\Big(\max_{u \in \mathcal{U}, h: \|h\|_\infty \leq 1} (W_f u + U_f h + b_f) \Big) \Big\|_\infty \\ & \leq \sigma\Big(\max_{u \in \mathcal{U}, h: \|h\|_\infty \leq 1} \|W_f u + U_f h + b_f\|_\infty \Big) \\ & \leq \sigma\left(\|W_f \quad U_f \quad b_f\|_\infty\right) = \check{\sigma}_f. \end{aligned} \tag{A.50}$$

Notice that, owing to the boundedness of the sigmoidal function, $\check{\sigma}_f < 1$. Thus, $|[f_k]_j| \leq \check{\sigma}_f < 1$. By similar arguments, in light of the symmetry of σ with respect to the point $(0, \frac{1}{2})$, it can be shown that

$$\left| \left[\sigma(W_f u_k + U_f c_k + b_f)\right]_j \right| \geq \sigma\left(-\|W_f \quad U_f \quad b_f\|_\infty\right) = 1 - \check{\sigma}_f. \tag{A.51}$$

Since this implies that $0 < 1 - \check{\sigma}_f \leq |[f_k]_j|$, (3.9a) is proven. The same chain of inequalities can be used to prove (3.9b) and (3.9c).

As far as (3.9d) is concerned, we point out that the $\tanh$ activation function enjoys the same strict monotonicity property than σ, but it is symmetric with respect to the origin instead of the point $(0, \frac{1}{2})$. This entails that

$$\left| \left[\phi(W_r u_k + U_r c_k + b_r)\right]_j \right| \leq \phi\left(\|W_r \quad U_r \quad b_r\|_\infty\right) = \check{\phi}_r \tag{A.52}$$

and

$$\left| \left[\phi(W_r u_k + U_r c_k + b_r)\right]_j \right| \geq \phi\left(-\|W_r \quad U_r \quad b_r\|_\infty\right) = -\check{\phi}_r, \tag{A.53}$$

which proves the bound (3.9d). □

A.2.4 Proof of Proposition 3.1

First, we show that $\mathcal{C}$ is an invariant set for the cell state c. To this end, one must show that $c_k \in \mathcal{C} \implies c_{k+1} \in \mathcal{C}$ for any $u_k \in \mathcal{U}$.

Consider the j-th component of the c_{k+1}, with $j \in \{1, ..., n_c\}$. In light of (3.8a), by taking the absolute value we get

$$|[c_{k+1}]_j| \leq |[f_k]_j| \, |[c_k]_j| + |[i_k]_j| \, |[r_k]_j|. \tag{A.54}$$

Recalling (3.9), and owing to (3.11b), it follows that

$$\begin{aligned} |[c_{k+1}]_j| &\leq \check{\sigma}_f \, |[c_k]_j| + \check{\sigma}_i \, \check{\phi}_r \\ &\leq \check{\sigma}_f \, \|c_k\|_\infty + \check{\sigma}_i \, \check{\phi}_r \\ &\leq \check{\sigma}_f \, \check{c} + \check{\sigma}_i \, \check{\phi}_r = \check{\sigma}_f \, \frac{\check{\sigma}_i \, \check{\phi}_r}{1 - \check{\sigma}_f} + \check{\sigma}_i \, \check{\phi}_r = \check{c}, \end{aligned} \tag{A.55}$$

i.e. $\mathcal{C}$ is an invariant set for the state c_k.

We now show that $h_k \in \mathcal{H} \implies h_{k+1} \in \mathcal{H}$ for any $u_k \in \mathcal{U}$. In light of (3.8a), for any component $j \in \{1, ..., n_c\}$,

$$|[h_{k+1}]_j| \leq |[z_k]_j| \, |[\phi(c_{k+1})]_j|. \tag{A.56}$$

Leveraging the bound (3.9c) and the known bound of c_{k+1} (3.11b), thanks to the monotonicity of ϕ we get

$$\begin{aligned} |[h_{k+1}]_j| &\leq \|z_k\|_\infty \|\phi(c_{k+1})\|_\infty \\ &\leq \check{\sigma}_z \phi(\check{c}) \\ &\leq \phi(\check{c}) = \check{h}, \end{aligned} \tag{A.57}$$

i.e. $\mathcal{H}$ is an invariant set for the hidden state h_k. The fact that $\mathcal{X}$, defined in (3.11a), is an invariant set for x_k follows intuitively. □

A.2.5 Proof of Theorem 3.3

Consider the shallow LSTM equations (3.8). Noticing that, given two vectors v and w their Hadamard product $v \circ w$ is equivalent to $\text{diag}(v)w$, the first state equation can be rewritten as

$$c_{k+1} = \text{diag}(f_k)c_k + \text{diag}(i_k)\phi(W_r u_k + U_r h_k + b_r). \tag{A.58}$$

We point out that, for any diagonal matrix $A = \text{diag}(a_1, ..., a_n)$, it holds that $\|A\|_2 \leq \max_{i=1,...,n}|a_i|$. Moreover, being ϕ a 1-Lipschitz function, $\|\phi(v)\|_2 \leq 1\|v\|_2$. Thus, applying the 2-norm to both sides of (A.58) we get

$$\begin{aligned} \|c_{k+1}\|_2 &\leq \|\text{diag}(f_k)\|_2 \|c_k\|_2 + \|\text{diag}(i_k)\|_2 \|\phi(W_r u_k + U_r h_k + b_r)\|_2 \\ &\leq \check{\sigma}_f \|c_k\|_2 + \check{\sigma}_i \big(\|W_r\|_2 \|u_k\|_2 + \|U_r\|_2 \|h_k\|_2 + \|b_r\|_2\big), \end{aligned} \tag{A.59}$$

where the bounds (3.9) have been used. Analogously, by taking the 2-norm of the second state equation we get

$$\begin{aligned} \|h_{k+1}\|_2 &\leq \|\text{diag}(z_k)\|_2 \|\phi(c_{k+1})\|_2 \\ &\leq \check{\sigma}_z \|c_{k+1}\|_2 \\ &\leq \check{\sigma}_z \check{\sigma}_f \|c_k\|_2 + \check{\sigma}_z \check{\sigma}_i \big(\|W_r\|_2 \|u_k\|_2 + \|U_r\|_2 \|h_k\|_2 + \|b_r\|_2\big) \end{aligned} \tag{A.60}$$

In light of (A.59) and (A.60), it holds that

$$\begin{bmatrix} \|c_{k+1}\|_2 \\ \|h_{k+1}\|_2 \end{bmatrix} \leq \mathfrak{A} \begin{bmatrix} \|c_k\|_2 \\ \|h_k\|_2 \end{bmatrix} + \mathfrak{B}_u \|u_k\|_2 + \mathfrak{B}_b \|b_r\|_2, \tag{A.61a}$$

where matrix $\mathfrak{A}$ is that defined in (3.13), and matrices $\mathfrak{B}_u$ and $\mathfrak{B}_b$ are

$$\mathfrak{B}_u = \begin{bmatrix} \check{\sigma}_i \|W_r\|_2 \\ \check{\sigma}_z \check{\sigma}_i \|W_r\|_2 \end{bmatrix}, \quad \mathfrak{B}_b = \begin{bmatrix} \check{\sigma}_i \\ \check{\sigma}_z \check{\sigma}_i \end{bmatrix}. \tag{A.61b}$$

Iterating (A.61a) we get

$$\begin{bmatrix} \|c_k\|_2 \\ \|h_k\|_2 \end{bmatrix} \leq \mathfrak{A}^k \begin{bmatrix} \|c_0\|_2 \\ \|h_0\|_2 \end{bmatrix} + \sum_{\tau=0}^{k-1} \mathfrak{A}^{k-\tau-1} \left(\mathfrak{B}_u \|u_\tau\|_2 + \mathfrak{B}_b \|b_r\|_2 \right). \tag{A.62}$$

Noticing that

$$\left\| \begin{bmatrix} \|c_\tau\|_2 \\ \|h_\tau\|_2 \end{bmatrix} \right\|_2 = \left\| \begin{bmatrix} c_\tau \\ h_\tau \end{bmatrix} \right\|_2 = \|x_\tau\|_2,$$

the Schur stability of $\mathfrak{A}$ implies the existence of $\mu_{\mathfrak{A}} > 0$ and $\lambda_{\mathfrak{A}} \in (0,1)$ such that

$$\begin{aligned} \|x_k\|_2 &\leq \mu_{\mathfrak{A}} \, \|x_0\|_2 \, \lambda_{\mathfrak{A}}^k + \left\| \sum_{\tau=0}^{k-1} \mathfrak{A}^{k-\tau-1} \left(\mathfrak{B}_u \|u_\tau\|_2 + \mathfrak{B}_b \|b_r\|_2 \right) \right\|_2 \\ &\leq \mu_{\mathfrak{A}} \, \|x_0\|_2 \, \lambda_{\mathfrak{A}}^k + \left\| (I_{2,2} - \mathfrak{A})^{-1} \mathfrak{B}_u \right\|_2 \|u_{0:k}\|_{2,\infty} \\ &\qquad + \left\| (I_{2,2} - \mathfrak{A})^{-1} \mathfrak{B}_b \right\|_2 \|b_r\|_2. \end{aligned} \tag{A.63}$$

Therefore, according to Definition 2.4, the system is ℓ_2-ISPS with functions

$$\beta(\|x_0\|_2, k) = \mu_{\mathfrak{A}} \, \|x_0\|_2 \, \lambda_{\mathfrak{A}}^k, \tag{A.64a}$$

$$\gamma(\|u_{0:k}\|_{2,\infty}) = \left\| (I_{2,2} - \mathfrak{A})^{-1} \mathfrak{B}_u \right\|_2 \|u_{0:k}\|_{2,\infty}, \tag{A.64b}$$

$$\varrho = \left\| (I_{2,2} - \mathfrak{A})^{-1} \mathfrak{B}_b \right\|_2 \|b_r\|_2. \tag{A.64c}$$

□

A.2.6 Proof of Proposition 3.2

In order to prove the claims of Proposition 3.2, the following auxiliary lemma is required.

Lemma A.1. *Consider a* 2*-by-*2 *positive-valued matrix* A*, with* $\operatorname{trace}(A) > 0$*. Then* A *is Schur stable if and only if*

$$-1 + \operatorname{trace}(A) < \det(A) < 1. \tag{A.65}$$

Proof of Lemma A.1. To characterize the stability of A, let us compute its characteristic equation

$$p(s) = \det(sI_{2,2} - A) = s^2 + vs + w = 0, \tag{A.66}$$

where $v = -\operatorname{trace}(A) = -a_{11} - a_{22}$ and $w = \det(A) = a_{11}a_{22} - a_{12}a_{21}$, a_{ij} denoting the element of A in position (i,j). Jury's criterion [181] can be used to provide necessary and sufficient conditions for the Schur stability of A. In particular, the Jury table of $p(s)$ reads as

$$\begin{array}{ccc} 1 & v & w \\ 1-w^2 & v(1-w) & \\ \frac{1-w}{1+w}\left((1+w)^2-v^2\right) & & \end{array} \tag{A.67}$$

According to Jury's criterion, the entries on the first column must be positive, i.e.

$$\begin{cases} 1-w^2>0 \\ \frac{1-w}{1+w}\left((1+w)^2-v^2\right)>0 \end{cases}.$$

Since $\mathrm{trace}(A)\geq 0$, it holds that $v\leq 0$, so that the previous conditions boil down to

$$\begin{cases} w<1 \\ w>-v-1 \end{cases}, \tag{A.68}$$

or, equivalently,

$$-1+\mathrm{trace}(A)<\det(A)<1. \tag{A.69}$$

□

At this stage, let us notice that $\mathfrak{A}$ is a 2-by-2 matrix having $\mathrm{trace}(\mathfrak{A})=\breve{\sigma}_f+\breve{\sigma}_z\breve{\sigma}_i\|U_r\|_2>0$. Invoking Lemma A.1, we can state that $\mathfrak{A}$ is Schur stable if and only if (A.65) is fulfilled, i.e.

$$-1+\breve{\sigma}_f+\breve{\sigma}_z\breve{\sigma}_i\|U_r\|_2<0<1, \tag{A.70}$$

which is the same as (3.14). □

A.2.7 Proof of Theorem 3.4

Consider the two states $x_{a,k}=[c'_{a,k},h'_{a,k}]'\in\mathcal{X}$ and $x_{b,k}=[c'_{b,k},h'_{b,k}]'\in\mathcal{X}$, and two inputs $u_{a,k}\in\mathcal{U}$ and $u_{b,k}\in\mathcal{U}$. First, we compute a bound for $\|c_{a,k+1}-c_{b,k+1}\|_2$ and $\|h_{a,k+1}-h_{b,k+1}\|_2$, which will be then used to prove the δISS of the system.

For the sake of compactness, in the following we denote

$$f_{a,k} = \sigma(W_f u_{a,k} + U_f h_{a,k} + b_f) \qquad f_{b,k} = \sigma(W_f u_{b,k} + U_f h_{b,k} + b_f) \tag{A.71a}$$

$$i_{a,k} = \sigma(W_i u_{a,k} + U_i h_{a,k} + b_i) \qquad i_{b,k} = \sigma(W_i u_{b,k} + U_i h_{b,k} + b_i) \tag{A.71b}$$

$$z_{a,k} = \sigma(W_z u_{a,k} + U_z h_{a,k} + b_z) \qquad z_{b,k} = \sigma(W_z u_{b,k} + U_z h_{b,k} + b_z) \tag{A.71c}$$

$$r_{a,k} = \phi(W_r u_{a,k} + U_r h_{a,k} + b_r) \qquad r_{b,k} = \phi(W_r u_{b,k} + U_r h_{b,k} + b_r) \tag{A.71d}$$

In light of the first state equation (3.8a), $c_{a,k+1} - c_{b,k+1}$ reads as

$$c_{a,k+1} - c_{b,k+1} = f_{a,k} \circ c_{a,k} + i_{a,k} \circ r_{a,k} - f_{b,k} \circ c_{b,k} - i_{b,k} \circ r_{b,k}. \tag{A.72}$$

Let us sum and subtract the terms $f_{a,k} \circ c_{b,k}$ and $i_{a,k} \circ r_{b,k}$ to the right-hand side of (A.72). We thus obtain

$$\begin{aligned} c_{a,k+1} - c_{b,k+1} = f_{a,k} \circ (c_{a,k} - c_{b,k}) + i_{a,k} \circ (r_{a,k} - r_{b,k}) \\ + (f_{a,k} - f_{b,k}) \circ c_{b,k} - (i_{a,k} - i_{b,k}) \circ r_{b,k}. \end{aligned} \tag{A.73}$$

Recalling (A.71), owing to the 1-Lipschitzianity of ϕ and to the $\frac{1}{4}$-Lipschitzianity of σ, it holds that

$$\begin{aligned} \|f_{a,k} - f_{b,k}\|_2 &\leq \frac{1}{4} \left\| W_f u_{a,k} + U_f h_{a,k} - W_f u_{b,k} - U_f h_{b,k} \right\|_2 \\ &\leq \frac{1}{4} \big(\|W_f\|_2 \|u_{a,k} - u_{b,k}\|_2 + \|U_f\|_2 \|h_{a,k} - h_{b,k}\|_2 \big) \end{aligned} \tag{A.74a}$$

$$\begin{aligned} \|i_{a,k} - i_{b,k}\|_2 &\leq \frac{1}{4} \left\| W_i u_{a,k} + U_i h_{a,k} - W_i u_{b,k} - U_i h_{b,k} \right\|_2 \\ &\leq \frac{1}{4} \big(\|W_i\|_2 \|u_{a,k} - u_{b,k}\|_2 + \|U_i\|_2 \|h_{a,k} - h_{b,k}\|_2 \big) \end{aligned} \tag{A.74b}$$

$$\begin{aligned} \|z_{a,k} - z_{b,k}\|_2 &\leq \frac{1}{4} \left\| W_z u_{a,k} + U_z h_{a,k} - W_z u_{b,k} - U_z h_{b,k} \right\|_2 \\ &\leq \frac{1}{4} \big(\|W_z\|_2 \|u_{a,k} - u_{b,k}\|_2 + \|U_z\|_2 \|h_{a,k} - h_{b,k}\|_2 \big) \end{aligned} \tag{A.74c}$$

$$\begin{aligned} \|r_{a,k} - r_{b,k}\|_2 &\leq 1 \left\| W_r u_{a,k} + U_r h_{a,k} - W_r u_{b,k} - U_r h_{b,k} \right\|_2 \\ &\leq \|W_r\|_2 \|u_{a,k} - u_{b,k}\|_2 + \|U_r\|_2 \|h_{a,k} - h_{b,k}\|_2 \end{aligned} \tag{A.74d}$$

Taking the 2-norm of both sides of (A.73), and exploiting (A.74), one gets

$$\begin{aligned}\|c_{a,k+1}-c_{b,k+1}\|_2 \leq & \,\|\operatorname{diag}(f_{a,k})\|_2\,\|c_{a,k}-c_{b,k}\|_2\\ &+\|\operatorname{diag}(i_{a,k})\|_2\,\|r_{a,k}-r_{b,k}\|_2\\ &+\|\operatorname{diag}(c_{b,k})\|_2\,\|f_{a,k}-f_{b,k}\|_2\\ &+\|\operatorname{diag}(r_{b,k})\|_2\,\|i_{a,k}-i_{b,k}\|_2\\ \leq & \,\check{\sigma}_f\|c_{a,k}-c_{b,k}\|_2\\ &+\check{\sigma}_i\big(\|W_r\|_2\,\|u_{a,k}-u_{b,k}\|_2+\|U_r\|_2\,\|h_{a,k}-h_{b,k}\|_2\big)\\ &+\frac{1}{4}\check{c}\big(\|W_f\|_2\,\|u_{a,k}-u_{b,k}\|_2+\|U_f\|_2\,\|h_{a,k}-h_{b,k}\|_2\big)\\ &+\frac{1}{4}\check{\phi}_r\big(\|W_i\|_2\,\|u_{a,k}-u_{b,k}\|_2+\|U_i\|_2\,\|h_{a,k}-h_{b,k}\|_2\big)\end{aligned} \tag{A.75}$$

By collecting the common terms we thus get

$$\begin{aligned}\|c_{a,k+1}-c_{b,k+1}\|_2 \leq & \,\check{\sigma}_f\|c_{a,k}-c_{b,k}\|_2\\ &+\Big(\check{\sigma}_i\|U_r\|_2+\frac{1}{4}\check{c}\|U_f\|_2+\frac{1}{4}\check{\phi}_r\|U_i\|_2\Big)\|h_{a,k}-h_{b,k}\|_2\\ &+\Big(\check{\sigma}_i\|W_r\|_2+\frac{1}{4}\check{c}\|W_f\|_2+\frac{1}{4}\check{\phi}_r\|W_i\|_2\Big)\|u_{a,k}-u_{b,k}\|_2.\end{aligned} \tag{A.76}$$

Recalling the definition of $\check{c}$ in (3.12a) and the definition of $\check{\alpha}$ in (3.15b), letting

$$\check{k}_u=\check{\sigma}_i\|W_r\|_2+\frac{1}{4}\check{c}\|W_f\|_2+\frac{1}{4}\check{\phi}_r\|W_i\|_2, \tag{A.77}$$

the inequality (A.76) can be rewritten as

$$\|c_{a,k+1}-c_{b,k+1}\|_2 \leq \check{\sigma}_f\|c_{a,k}-c_{b,k}\|_2+\check{\alpha}\|h_{a,k}-h_{b,k}\|_2+\check{k}_u\|u_{a,k}-u_{b,k}\|_2. \tag{A.78}$$

Concerning the second state equation of (3.8a), one gets

$$h_{a,k+1}-h_{b,k+1}=z_{a,k}\circ\phi(c_{a,k+1})-z_{b,k}\circ\phi(c_{b,k+1}). \tag{A.79}$$

By adding and subtracting the term $z_{a,k}\circ\phi(c_{b,k+1})$ to the right-hand side of the equation

$$h_{a,k+1}-h_{b,k+1}=z_{a,k}\circ\big(\phi(c_{a,k+1})-\phi(c_{b,k+1})\big)-(z_{a,k}-z_{b,k})\circ\phi(c_{b,k+1}). \tag{A.80}$$

At this stage, let us notice that, since ϕ is strictly increasing, it holds that $\|\phi(c_{b,k+1})\|_\infty \leq \phi(\check{c})=\check{h}$, see (3.12b). Then, recalling the bounds (3.9) and (A.74), we now take the 2-norm of both sides of (A.80), which leads to

$$\begin{aligned}\|h_{a,k+1}-h_{b,k+1}\|_2 &\leq \|\operatorname{diag}(z_{a,k})\|_2\,\|\phi(c_{a,k+1})-\phi(c_{b,k+1})\|_2\\ &\qquad+\|\operatorname{diag}\big(\phi\big(c_{b,k+1}\big)\big)\|_2\,\|z_{a,k}-z_{b,k}\|\\ &\leq \breve{\sigma}_z\|c_{a,k+1}-c_{b,k+1}\|_2\\ &\qquad+\frac{1}{4}\breve{h}\big[\|W_z\|_2\,\|u_{a,k}-u_{b,k}\|_2+\|U_z\|_2\,\|h_{a,k}-h_{b,k}\|_2\big],\end{aligned} \tag{A.81}$$

where the 1-Lipschitzianity of ϕ has also been exploited. Applying the bound (A.78), and collecting the common terms, we thus get

$$\begin{aligned}\|h_{a,k+1}-h_{b,k+1}\|_2 &\leq \breve{\sigma}_z\breve{\sigma}_f\|c_{a,k}-c_{b,k}\|_2+\breve{\sigma}_z\breve{\alpha}\|h_{a,k}-h_{b,k}\|_2\\ &\quad+\breve{\sigma}_z\breve{k}_u\|u_{a,k}-u_{b,k}\|_2\\ &\quad+\frac{1}{4}\breve{h}\big(\|W_z\|_2\,\|u_{a,k}-u_{b,k}\|_2+\|U_z\|_2\,\|h_{a,k}-h_{b,k}\|_2\big)\\ &\leq \breve{\sigma}_z\breve{\sigma}_f\|c_{a,k}-c_{b,k}\|_2\\ &\quad+\left(\breve{\sigma}_z\breve{\alpha}+\frac{1}{4}\breve{h}\|U_z\|_2\right)\|h_{a,k}-h_{b,k}\|_2\\ &\quad+\left(\breve{\sigma}_z\breve{k}_u+\frac{1}{4}\breve{h}\|W_z\|_2\right)\|u_{a,k}-u_{b,k}\|_2.\end{aligned} \tag{A.82}$$

In light of (A.78) and (A.78), it follows

$$\begin{bmatrix}\|c_{a,k+1}-c_{b,k+1}\|_2\\ \|h_{a,k+1}-h_{b,k+1}\|_2\end{bmatrix}\leq \mathfrak{A}_\delta\begin{bmatrix}\|c_{a,k}-c_{b,k}\|_2\\ \|h_{a,k}-h_{b,k}\|_2\end{bmatrix}+\mathfrak{B}_\delta\|u_{a,k}-u_{b,k}\|_2, \tag{A.83a}$$

where the matrix $\mathfrak{A}_\delta$ is defined according to (3.15a) and the matrix $\mathfrak{B}_\delta$ reads as

$$\mathfrak{B}_\delta=\begin{bmatrix}\breve{k}_u\\ \breve{\sigma}_z\breve{k}_u+\frac{1}{4}\breve{h}\|W_z\|_2\end{bmatrix}. \tag{A.83b}$$

Iterating (A.83a) we hence obtain

$$\begin{bmatrix}\|c_{a,k}-c_{b,k}\|_2\\ \|h_{a,k}-h_{b,k}\|_2\end{bmatrix}\leq \mathfrak{A}_\delta^k\begin{bmatrix}\|c_{a,0}-c_{b,0}\|_2\\ \|h_{a,0}-h_{b,0}\|_2\end{bmatrix}+\sum_{\tau=0}^{k-1}\mathfrak{A}_\delta^{k-\tau-1}\mathfrak{B}_\delta\|u_{a,\tau}-u_{b,\tau}\|_2. \tag{A.84}$$

Noticing that

$$\left\|\begin{bmatrix}\|c_{a,\tau}-c_{b,\tau}\|_2\\ \|h_{a,\tau}-h_{b,\tau}\|_2\end{bmatrix}\right\|_2=\left\|\begin{bmatrix}c_{a,\tau}-c_{b,\tau}\\ h_{a,\tau}-h_{b,\tau}\end{bmatrix}\right\|_2=\|x_{a,\tau}-x_{b,\tau}\|_2,$$

the Schur stability of $\mathfrak{A}_\delta$ implies the existence of $\mu_{\mathfrak{A}_\delta} > 0$ and $\lambda_{\mathfrak{A}_\delta} \in (0,1)$ such that (A.84) can be bounded as

$$\begin{aligned} \|x_{a,k} - x_{b,k}\|_2 \leq \mu_{\mathfrak{A}_\delta} \|x_{a,0} - x_{b,0}\|_2 \, \lambda_{\mathfrak{A}_\delta}^k & \\ + \left\| \sum_{\tau=0}^{k-1} \mathfrak{A}_\delta^{k-\tau-1} \mathfrak{B}_\delta \|u_{a,\tau} - u_{b,\tau}\|_2 \right\|_2 & \\ \leq \mu_{\mathfrak{A}_\delta} \|x_{a,0} - x_{b,0}\|_2 \, \lambda_{\mathfrak{A}_\delta}^k & \\ + \left\| (I_{2,2} - \mathfrak{A}_\delta)^{-1} \mathfrak{B}_\delta \right\|_2 \|u_{a,0:k} - u_{b,0:k}\|_{2,\infty}. & \end{aligned} \tag{A.85}$$

Therefore, according to Definition 2.6, the system is ℓ_2-δISS with functions

$$\beta(\|x_{a,0} - x_{b,0}\|_2, k) = \mu_{\mathfrak{A}_\delta} \|x_{a,0} - x_{b,0}\|_2 \, \lambda_{\mathfrak{A}_\delta}^k, \tag{A.86a}$$

$$\gamma(\|u_{a,0:k} - u_{b,0:k}\|_{2,\infty}) = \left\| (I_{2,2} - \mathfrak{A}_\delta)^{-1} \mathfrak{B}_\delta \right\|_2 \|u_{a,0:k} - u_{b,0:k}\|_{2,\infty}. \tag{A.86b}$$

□

A.2.8 Proof of Proposition 3.3

Let us point out that, since $\breve{\sigma}_f > 0$, $\breve{\sigma}_z > 0$, $\breve{h} > 0$, and $\breve{\alpha} > 0$, it holds that

$$\text{trace}(\mathfrak{A}_\delta) = \breve{\sigma}_f + \breve{\sigma}_z \breve{\alpha} + \frac{1}{4} \breve{h} \|U_z\|_2 > 0. \tag{A.87}$$

Moreover,

$$\det(\mathfrak{A}_\delta) = \breve{\sigma}_f \breve{\sigma}_z \breve{\alpha} + \frac{1}{4} \breve{\sigma}_f \breve{h} \|U_z\|_2 - \breve{\sigma}_f \breve{\sigma}_z \breve{\alpha} = \frac{1}{4} \breve{\sigma}_f \breve{h} \|U_z\|_2. \tag{A.88}$$

Therefore, Lemma A.1 can be invoked, which guarantees that a necessary and sufficient condition for the Schur stability of $\mathfrak{A}_\delta$ is that the following inequalities hold

$$\breve{\sigma}_f + \breve{\sigma}_z \breve{\alpha} + \frac{1}{4} \breve{h} \|U_z\|_2 - 1 < \frac{1}{4} \breve{\sigma}_f \breve{h} \|U_z\|_2, \tag{A.89a}$$

$$\frac{1}{4} \breve{\sigma}_f \breve{h} \|U_z\|_2 < 1. \tag{A.89b}$$

Moving all the terms to the left-hand side, one obtains (3.16). □

A.2.9 Proof of Proposition 3.4

Consider (3.16a), and let us move the fourth term to the right-hand side.

$$\check{\sigma}_f + \check{\sigma}_z \check{\alpha} + \frac{1}{4}\check{h}\|U_z\|_2 - 1 < \frac{1}{4}\check{\sigma}_f \check{h}\|U_z\|_2. \tag{A.90}$$

Let us now replace $\check{\alpha}$ with its definition, given in (3.15b).

$$\begin{aligned}\check{\sigma}_f + \check{\sigma}_z &\left(\frac{1}{4}\check{c}\|U_f\|_2 + \check{\sigma}_i\|U_r\|_2 + \frac{1}{4}\check{\phi}_r\|U_i\|_2\right) + \frac{1}{4}\check{h}\|U_z\|_2 - 1 \\ &< \frac{1}{4}\check{\sigma}_f \check{h}\|U_z\|_2.\end{aligned} \tag{A.91}$$

By trivial manipulations we get

$$\begin{aligned}&\check{\sigma}_f + \check{\sigma}_z\check{\sigma}_i\|U_r\|_2 - 1 \\ &\quad < -\Big(\frac{1}{4}\check{c}\|U_f\|_2 + \frac{1}{4}\check{\phi}_r\|U_i\|_2\Big)\check{\sigma}_z - \frac{1}{4}(1-\check{\sigma}_f)\check{h}\|U_z\|_2,\end{aligned} \tag{A.92}$$

and since, owing to $\check{\sigma}_f \in (0, 1)$, the right hand side is surely negative, the inequality (A.92) implies

$$\check{\sigma}_f + \check{\sigma}_z\check{\sigma}_i\|U_r\|_2 - 1 < 0, \tag{A.93}$$

which is exactly the ISPS sufficient condition (3.14). □

A.2.10 Proof of Lemma 3.2

The proof is straightforward from Remark 3.1. Indeed, owing to the unity-boundedness of the input $u_k^{(l)}$ of each layer, Lemma 3.1 can be applied independently to each layer, which leads to (3.19) and (3.20). □

A.2.11 Proof of Proposition 3.5

Owing to Remark 3.1, the proof is straightforward. Since the input of each layer is unity-bounded, Proposition 3.1 can be applied to each layer $l \in \{1, ..., L\}$ of the deep LSTM individually and independently from the others. Thus, Proposition 3.5 yields, for each layer, the invariant set $\mathcal{X}^{(l)}$ defined in (3.21b). In light of the definition of the layer's state vector given in (3.17c), (3.21) is an invariant set of the deep LSTM model. □

A.2.12 Proof of Theorem 3.5

First, let us point out that owing to Remark 3.1, the layer-wise satisfaction of Proposition 3.2 – that is, the fulfillment of condition (3.23) $\forall l \in$

$\{1, ..., L\}$ – implies that each layer is ISPS. To ensure that (3.17) is ISPS, we need however to show that the ISPS definition applies.

To this end, let us consider the l-th layer. Owing to its ISPS, the bound (A.61) here reads as

$$\begin{bmatrix} \|c_{k+1}^{(l)}\|_2 \\ \|h_{k+1}^{(l)}\|_2 \end{bmatrix} \leq \mathfrak{A}^{(l)} \begin{bmatrix} \|c_k^{(l)}\|_2 \\ \|h_k^{(l)}\|_2 \end{bmatrix} + \mathfrak{B}_u^{(l)} \|u_k^{(l)}\|_2 + \mathfrak{B}_b^{(l)} \|b_r^{(l)}\|_2, \tag{A.94a}$$

where matrices $\mathfrak{A}^{(l)}$, $\mathfrak{B}_u^{(l)}$ and $\mathfrak{B}_b^{(l)}$ are defined analogously to (3.13) and (A.61b), i.e.

$$\begin{aligned} \mathfrak{A}^{(l)} &= \begin{bmatrix} \breve{\sigma}_f^{(l)} & \breve{\sigma}_i^{(l)} \|U_r^{(l)}\|_2 \\ \breve{\sigma}_z^{(l)} \breve{\sigma}_f^{(l)} & \breve{\sigma}_z^{(l)} \breve{\sigma}_i^{(l)} \|U_r^{(l)}\|_{2}, \end{bmatrix}, \\ \mathfrak{B}_u^{(l)} &= \begin{bmatrix} \breve{\sigma}_i^{(l)} \|W_r^{(l)}\|_2 \\ \breve{\sigma}_z^{(l)} \breve{\sigma}_i^{(l)} \|W_r^{(l)}\|_2 \end{bmatrix}, \quad \mathfrak{B}_b^{(l)} = \begin{bmatrix} \breve{\sigma}_i^{(l)} \\ \breve{\sigma}_z^{(l)} \breve{\sigma}_i^{(l)} \end{bmatrix}. \end{aligned} \tag{A.94b}$$

Recalling the definition of the input $u_k^{(l+1)}$ given in (3.17b), it holds that

$$\begin{aligned} \|u_k^{(l+1)}\|_2 \leq \|h_{k+1}^{(l)}\|_2 & \\ \leq \begin{bmatrix} 0 & 1 \end{bmatrix} \mathfrak{A}^{(l)} \begin{bmatrix} \|c_k^{(l)}\|_2 \\ \|h_k^{(l)}\|_2 \end{bmatrix} + \begin{bmatrix} 0 & 1 \end{bmatrix} \mathfrak{B}_u^{(l)} \|u_k^{(l)}\|_2 + \begin{bmatrix} 0 & 1 \end{bmatrix} \mathfrak{B}_b^{(l)} \|b_r^{(l)}\|_2. \end{aligned} \tag{A.95}$$

Thus, letting $\tilde{\mathfrak{A}}^{(l)} = \begin{bmatrix} 0 & 1 \end{bmatrix} \mathfrak{A}^{(l)}$, $\tilde{\mathfrak{B}}_u^{(l)} = \begin{bmatrix} 0 & 1 \end{bmatrix} \mathfrak{B}_u^{(l)}$ and $\tilde{\mathfrak{B}}_b^{(l)} = \begin{bmatrix} 0 & 1 \end{bmatrix} \mathfrak{B}_b^{(l)}$, by iterating (A.94a) over layers $l = 1, ..., L$ one gets

$$\begin{bmatrix} \|c_{k+1}^{(1)}\|_2 \\ \|h_{k+1}^{(1)}\|_2 \\ \vdots \\ \|c_{k+1}^{(L)}\|_2 \\ \|h_{k+1}^{(L)}\|_2 \end{bmatrix} \leq \mathfrak{A}_L \begin{bmatrix} \|c_k^{(1)}\|_2 \\ \|h_k^{(1)}\|_2 \\ \vdots \\ \|c_k^{(L)}\|_2 \\ \|h_k^{(L)}\|_2 \end{bmatrix} + \mathfrak{B}_{Lu} \|u_k\|_2 + \mathfrak{B}_{Lb} \begin{bmatrix} \|b_r^{(1)}\|_2 \\ \vdots \\ \|b_r^{(L)}\|_2 \end{bmatrix} \tag{A.96a}$$

where the matrices $\mathfrak{A}_L$, $\mathfrak{B}_{Lu}$, and $\mathfrak{B}_{Lb}$ read as follows

$$\mathfrak{A}_L = \begin{bmatrix} \mathfrak{A}^{(1)} & 0_{2,2} & \dots & 0_{2,2} \\ \mathfrak{B}_u^{(2)} \tilde{\mathfrak{A}}^{(1)} & \mathfrak{A}^{(2)} & \dots & 0_{2,2} \\ \vdots & & \ddots & \vdots \\ \mathfrak{B}_u^{(L)} \big(\prod_{j=L-1}^{2} \tilde{\mathfrak{B}}_u^{(j)} \big) \tilde{\mathfrak{A}}^{(1)} & \mathfrak{B}_u^{(L)} \big(\prod_{j=L-1}^{3} \tilde{\mathfrak{B}}_u^{(j)} \big) \tilde{\mathfrak{A}}^{(2)} & \dots & \mathfrak{A}^{(L)} \end{bmatrix}, \tag{A.96b}$$

$$\mathfrak{B}_{Lu} = \begin{bmatrix} \mathfrak{B}_u^{(1)} \\ \mathfrak{B}_u^{(2)}\tilde{\mathfrak{B}}_u^{(1)} \\ \vdots \\ \mathfrak{B}_u^{(L)}\left(\prod_{j=L-1}^{1}\tilde{\mathfrak{B}}_u^{(j)}\right) \end{bmatrix}, \tag{A.96c}$$

$$\mathfrak{B}_{Lb} = \begin{bmatrix} \mathfrak{B}_b^{(1)} & 0_{2,1} & \dots & 0_{2,1} \\ \mathfrak{B}_b^{(2)}\tilde{\mathfrak{B}}_b^{(1)} & \mathfrak{B}_b^{(2)} & \dots & 0_{2,1} \\ \vdots & & \ddots & 0_{2,1} \\ \mathfrak{B}_b^{(L)}\left(\prod_{j=L-1}^{1}\tilde{\mathfrak{B}}_b^{(j)}\right) & \mathfrak{B}_b^{(L)}\left(\prod_{j=L-1}^{2}\tilde{\mathfrak{B}}_b^{(j)}\right) & \dots & \mathfrak{B}_b^{(L)} \end{bmatrix}. \tag{A.96d}$$

Then, by iterating (A.96a) over time we get

$$\begin{bmatrix} \|c_k^{(1)}\|_2 \\ \|h_k^{(1)}\|_2 \\ \vdots \\ \|c_k^{(L)}\|_2 \\ \|h_k^{(L)}\|_2 \end{bmatrix} \leq \mathfrak{A}_L^k \begin{bmatrix} \|c_0^{(1)}\|_2 \\ \|h_0^{(1)}\|_2 \\ \vdots \\ \|c_0^{(L)}\|_2 \\ \|h_0^{(L)}\|_2 \end{bmatrix} + \sum_{\tau=0}^{k-1} \mathfrak{A}_L^{k-\tau-1} \left(\mathfrak{B}_{Lu}\|u_\tau\|_2 + \mathfrak{B}_{Lb} \begin{bmatrix} \|b_r^{(1)}\|_2 \\ \vdots \\ \|b_r^{(L)}\|_2 \end{bmatrix} \right) \tag{A.97}$$

Let us observe that

$$\left\| \begin{bmatrix} \|c_\tau^{(1)}\|_2 \\ \|h_\tau^{(1)}\|_2 \\ \vdots \\ \|c_\tau^{(L)}\|_2 \\ \|h_\tau^{(L)}\|_2 \end{bmatrix} \right\|_2 = \left\| \begin{bmatrix} x_\tau^{(1)} \\ \vdots \\ x_\tau^{(L)} \end{bmatrix} \right\|_2 = \|x_\tau\|_2$$

and that, letting $b_r = [b_r^{(1)}, ..., b_r^{(L)}]'$, it holds that

$$\left\| \begin{bmatrix} \|b_r^{(1)\prime}\|_2 \\ \vdots \\ \|b_r^{(L)\prime}\|_2 \end{bmatrix} \right\|_2 = \left\| \begin{bmatrix} b_r^{(1)} \\ \vdots \\ b_r^{(L)} \end{bmatrix} \right\|_2 = \|b_r\|_2.$$

Moreover, we point out that $\mathfrak{A}_L$ is Schur stable, since it is block-triangular with Schur stable blocks on the diagonal. This implies that there exist

$\mu_{\mathfrak{A}_L} > 0$ and $\lambda_{\mathfrak{A}_L} \in (0, 1)$ such that

$$\begin{aligned} \|x_k\|_2 \leq \mu_{\mathfrak{A}_L} \lambda_{\mathfrak{A}_L}^k \|x_0\|_2 + \big\|(I_{2L,2L} - \mathfrak{A}_L)^{-1} \mathfrak{B}_{Lu}\big\|_2 \|u_{0:k}\|_{2,\infty} \\ + \big\|(I_{2L,2L} - \mathfrak{A}_L)^{-1} \mathfrak{B}_{Lb}\big\|_2 \|b_r\|_2. \end{aligned} \tag{A.98}$$

According to Definition 2.4, the deep LSTM is hence ℓ_2-ISPS with

$$\beta(\|x_0\|_2, k) = \mu_{\mathfrak{A}_L} \lambda_{\mathfrak{A}_L}^k \|x_0\|_2, \tag{A.99a}$$

$$\gamma(\|u_{0:k}\|_{2,\infty}) = \big\|(I_{2L,2L} - \mathfrak{A}_L)^{-1} \mathfrak{B}_{Lu}\big\|_2 \|u_{0:k}\|_{2,\infty}, \tag{A.99b}$$

$$\varrho = \big\|(I_{2L,2L} - \mathfrak{A}_L)^{-1} \mathfrak{B}_{Lb}\big\|_2 \|b_r\|_2. \tag{A.99c}$$

□

A.2.13 Proof of Theorem 3.6

In light of Remark 3.1, the layer-wise satisfaction of Proposition 3.3, i.e. the fulfillment of condition (3.24) $\forall l \in \{1, ..., L\}$, entails the δISS of each layer. Therefore, the bound (A.83a) can be here written, for the generic layer l, as

$$\begin{bmatrix} \|c_{a,k+1}^{(l)} - c_{b,k+1}^{(l)}\|_2 \\ \|h_{a,k+1}^{(l)} - h_{b,k+1}^{(l)}\|_2 \end{bmatrix} \leq \mathfrak{A}_\delta^{(l)} \begin{bmatrix} \|c_{a,k}^{(l)} - c_{b,k}^{(l)}\|_2 \\ \|h_{a,k}^{(l)} - h_{b,k}^{(l)}\|_2 \end{bmatrix} + \mathfrak{B}_\delta^{(l)} \|u_{a,k}^{(l)} - u_{b,k}^{(l)}\|_2, \tag{A.100a}$$

where matrices $\mathfrak{A}_\delta^{(l)}$ and $\mathfrak{B}_\delta^{(l)}$ are defined analogously to (3.15a) and (A.83b), respectively, i.e.

$$\begin{aligned} \mathfrak{A}_\delta^{(l)} &= \begin{bmatrix} \check{\sigma}_f^{(l)} & \check{\alpha}^{(l)} \\ \check{\sigma}_z^{(l)} \check{\sigma}_f^{(l)} & \check{\sigma}_z^{(l)} \check{\alpha}^{(l)} + \frac{1}{4} \check{h}^{(l)} \|U_z^{(l)}\|_2 \end{bmatrix}, \\ \mathfrak{B}_\delta^{(l)} &= \begin{bmatrix} \check{k}_u^{(l)} \\ \check{\sigma}_z^{(l)} \check{k}_u^{(l)} + \frac{1}{4} \check{h}^{(l)} \|W_z^{(l)}\|_2 \end{bmatrix}. \end{aligned} \tag{A.100b}$$

The term $\check{k}_u^{(l)}$ appearing in (A.100b) is defined, accordingly to (A.77), as

$$\check{k}_u^{(l)} = \check{\sigma}_i^{(l)} \|W_r^{(l)}\|_2 + \frac{1}{4} \check{c}^{(l)} \|W_f^{(l)}\|_2 + \frac{1}{4} \check{\phi}_r^{(l)} \|W_i^{(l)}\|_2, \tag{A.100c}$$

and the term $\check{\alpha}^{(l)}$ is that defined in (3.25).

Let us now recall that, owing to (3.17b), the input to the layer $l + 1$ is defined, for the two trajectories, as $u_{a,k}^{(l+1)} = h_{a,k+1}^{(l)}$ and $u_{b,k}^{(l+1)} = h_{b,k+1}^{(l)}$.

Therefore, from (A.100) it follows that

$$\begin{aligned}\|u_{a,k}^{(l+1)} - u_{b,k}^{(l+1)}\|_2 &\leq \|h_{a,k+1}^{(l)} - h_{b,k+1}^{(l)}\|_2 \\ &\leq \begin{bmatrix}0 & 1\end{bmatrix} \mathfrak{A}_\delta^{(l)} \begin{bmatrix}\|c_{a,k}^{(l)} - c_{b,k}^{(l)}\|_2 \\ \|h_{a,k}^{(l)} - h_{b,k}^{(l)}\|_2\end{bmatrix} + \begin{bmatrix}0 & 1\end{bmatrix} \mathfrak{B}_\delta^{(l)} \|u_{a,k}^{(l)} - u_{b,k}^{(l)}\|_2.\end{aligned} \tag{A.101}$$

Let, for the sake of compactness, $\tilde{\mathfrak{A}}_\delta^{(l)} = \begin{bmatrix}0 & 1\end{bmatrix} \mathfrak{A}_\delta^{(l)}$, $\tilde{\mathfrak{B}}_\delta^{(l)} = \begin{bmatrix}0 & 1\end{bmatrix} \mathfrak{B}_\delta^{(l)}$. By iterating (A.100a) over layers $l \in \{1, ..., L\}$ one gets

$$\begin{bmatrix}\|c_{a,k+1}^{(1)} - c_{b,k+1}^{(1)}\|_2 \\ \|h_{a,k+1}^{(1)} - h_{b,k+1}^{(1)}\|_2 \\ \vdots \\ \|c_{a,k+1}^{(L)} - c_{b,k+1}^{(L)}\|_2 \\ \|h_{a,k+1}^{(L)} - h_{b,k+1}^{(L)}\|_2\end{bmatrix} \leq \mathfrak{A}_{L\delta} \begin{bmatrix}\|c_{a,k}^{(1)} - c_{b,k}^{(1)}\|_2 \\ \|h_{a,k}^{(1)} - h_{b,k}^{(1)}\|_2 \\ \vdots \\ \|c_{a,k}^{(L)} - c_{b,k}^{(L)}\|_2 \\ \|h_{a,k}^{(L)} - h_{b,k}^{(L)}\|_2\end{bmatrix} + \mathfrak{B}_{L\delta} \|u_{a,k} - u_{b,k}\|_2 \tag{A.102a}$$

where the matrices $\mathfrak{A}_{L\delta}$ and $\mathfrak{B}_{L\delta}$ read as

$$\mathfrak{A}_{L\delta} = \begin{bmatrix} \mathfrak{A}_\delta^{(1)} & 0_{2,2} & \dots & 0_{2,2} \\ \mathfrak{B}_\delta^{(2)} \tilde{\mathfrak{A}}_\delta^{(1)} & \mathfrak{A}_\delta^{(2)} & \dots & 0_{2,2} \\ \vdots & & \ddots & \vdots \\ \mathfrak{B}_\delta^{(L)} \left(\prod_{j=L-1}^{2} \tilde{\mathfrak{B}}_\delta^{(j)}\right) \tilde{\mathfrak{A}}_\delta^{(1)} & \mathfrak{B}_\delta^{(L)} \left(\prod_{j=L-1}^{3} \tilde{\mathfrak{B}}_\delta^{(j)}\right) \tilde{\mathfrak{A}}_\delta^{(2)} & \dots & \mathfrak{A}_\delta^{(L)} \end{bmatrix}, \tag{A.102b}$$

$$\mathfrak{B}_{L\delta} = \begin{bmatrix} \mathfrak{B}_\delta^{(1)} \\ \mathfrak{B}_\delta^{(2)} \tilde{\mathfrak{B}}_\delta^{(1)} \\ \vdots \\ \mathfrak{B}_\delta^{(L)} \left(\prod_{j=L-1}^{1} \tilde{\mathfrak{B}}_\delta^{(j)}\right) \end{bmatrix}. \tag{A.102c}$$

Iterating (A.102a) over time, we thus get

$$\begin{bmatrix}\|c_{a,k}^{(1)} - c_{b,k}^{(1)}\|_2 \\ \|h_{a,k}^{(1)} - h_{b,k}^{(1)}\|_2 \\ \vdots \\ \|c_{a,k}^{(L)} - c_{b,k}^{(L)}\|_2 \\ \|h_{a,k}^{(L)} - h_{b,k}^{(L)}\|_2\end{bmatrix} \leq \mathfrak{A}_L^k \begin{bmatrix}\|c_{a,0}^{(1)} - c_{b,0}^{(1)}\|_2 \\ \|h_{a,0}^{(1)} - h_{b,0}^{(1)}\|_2 \\ \vdots \\ \|c_{a,0}^{(L)} - c_{b,0}^{(L)}\|_2 \\ \|h_{a,0}^{(L)} - h_{b,0}^{(L)}\|_2\end{bmatrix} + \sum_{\tau=0}^{k-1} \mathfrak{A}_L^{k-\tau-1} \mathfrak{B}_{L\delta} \|u_{a,\tau} - u_{b,\tau}\|_2. \tag{A.103}$$

Observing that

$$\left\| \begin{bmatrix} \|c_{a,\tau}^{(1)} - c_{b,\tau}^{(1)}\|_2 \\ \|h_{a,\tau}^{(1)} - h_{b,\tau}^{(1)}\|_2 \\ \vdots \\ \|c_{a,\tau}^{(L)} - c_{b,\tau}^{(L)}\|_2 \\ \|h_{a,\tau}^{(L)} - h_{b,\tau}^{(L)}\|_2 \end{bmatrix} \right\|_2 = \left\| \begin{bmatrix} x_{a,\tau}^{(1)} - x_{b,\tau}^{(1)} \\ \vdots \\ x_{a,\tau}^{(L)} - x_{b,\tau}^{(L)} \end{bmatrix} \right\|_2 = \|x_{a,\tau} - x_{b,\tau}\|_2,$$

and noticing that $\mathfrak{A}_{L\delta}$ is Schur stable (since it it is block-triangular with Schur stable blocks on the main diagonal), there exist $\mu_{\mathfrak{A}_{L\delta}} > 0$ and $\lambda_{\mathfrak{A}_{L\delta}} \in (0,1)$ such that

$$\begin{aligned}\|x_{a,k} - x_{b,k}\|_2 \leq \mu_{\mathfrak{A}_{L\delta}} \lambda_{\mathfrak{A}_{L\delta}}^k \|x_{a,0} - x_{b,0}\|_2 \\ + \left\|(I_{2L,2L} - \mathfrak{A}_{L\delta})^{-1} \mathfrak{B}_{L\delta}\right\|_2 \|u_{a,0:k} - u_{b,0:k}\|_{2,\infty}.\end{aligned} \tag{A.104}$$

Therefore, in light of Definition 2.6, the deep LSTM is ℓ_2-δISS with

$$\beta(\|x_{a,0} - x_{b,0}\|_2, k) = \mu_{\mathfrak{A}_{L\delta}} \lambda_{\mathfrak{A}_{L\delta}}^k \|x_{a,0} - x_{b,0}\|_2, \tag{A.105a}$$

$$\gamma(\|u_{a,0:k} - u_{b,0:k}\|_{2,\infty}) = \left\|(I_{2L,2L} - \mathfrak{A}_{L\delta})^{-1} \mathfrak{B}_{L\delta}\right\|_2 \|u_{a,0:k} - u_{b,0:k}\|_{2,\infty}. \tag{A.105b}$$

□

A.2.14 Proof of Lemma 3.3

Consider the j-th state component of the GRU, and let $a = [z_k]_j$, the state equation associated to such component reads as

$$[x_{k+1}]_j = a[x_k]_j + (1-a)[r_k]_j. \tag{A.106}$$

In light of the boundedness of the activation functions, i.e. $\sigma(\cdot) \in (0,1)$ and $\phi(\cdot) \in (-1,1)$, it holds that $a \in (0,1)$ and $[r_k]_j \in (-1,1)$. Hence, (A.106) is a convex combination between the term $[x_k]_j$ and $[r_k]_j$. Since $|[r_k]_j| < 1 \leq \check{x}$, this implies that $|[x_{k+1}]_j| \leq \check{x}$. By applying this argument to all the components j, it follows that $\|x_{k+1}\|_\infty \leq \check{x}$, i.e. $x_{k+1} \in \mathcal{X}$. □

A.2.15 Proof of Lemma 3.4

First, let us note that in the trivial case $x_0 \in \tilde{\mathcal{X}}$, $x_k \in \tilde{\mathcal{X}}$ for any $k \geq 0$, since by Lemma 3.3 $\tilde{\mathcal{X}}$ is an invariant set. We hence focus on the case $x_0 \in \mathcal{X} \setminus \tilde{\mathcal{X}}$, i.e., $1 < \|x_0\|_\infty \leq \check{x}$.

We now prove the first claim of the lemma. To this end, let us consider the generic state component $j \in \{1, ..., n_c\}$. Then, the state equation of such component reads as

$$[x_{k+1}]_j = [z_k]_j [x_k]_j + (1 - [z_k]_j)[r_k]_j \tag{A.107}$$

Notably, owing to the boundedness of σ and ϕ, it holds that $[z_k]_j \in (0, 1)$ and $[r_k]_j \in (-1, 1)$.

More specifically, noticing that by Lemma 3.3 $[x_k]_j \leq \|x_k\|_\infty \leq \check{x}$, and recalling the input boundedness assumption, i.e. $\|u_\tau\|_\infty \leq \check{u}$ for any τ, we can ensure existence of some strictly positive $\underline{z}$, $\bar{z}$, and $\underline{\varepsilon}$ such that, for any $k \geq 0$ and any state component $j \in \{1, ..., n_c\}$,

$$\begin{align} 0 < \underline{z} \leq [z_k]_j \leq \bar{z} < 1, \tag{A.108a} \\ |[r_k]_j| \leq 1 - \underline{\varepsilon} < 1. \tag{A.108b} \end{align}$$

These bounds can be constructed leveraging the fact that σ and ϕ are strictly increasing and Lipschitz continuous. The following chain of inequality thus holds

$$\begin{aligned} [z_k]_j &\leq \|z_k\|_\infty \\ &\leq \max_u \left\| \sigma \left(\begin{bmatrix} W_z & U_z & b_z \end{bmatrix} \begin{bmatrix} u \\ x_k \\ 1_{n_x,1} \end{bmatrix} \right) \right\|_\infty \\ &\leq \max_u \sigma \left(\left\| W_z \quad U_z \|x_k\|_\infty \quad b_z \right\|_\infty \left\| \begin{matrix} u \\ 1_{n_x,1} \\ 1_{n_x,1} \end{matrix} \right\|_\infty \right) \\ &\leq \sigma \left(\check{u} \left\| W_z \quad U_z \check{x} \quad b_z \right\|_\infty \right) = \bar{z} \end{aligned} \tag{A.109a}$$

Moreover, owing to the simmetry of σ with respect to the point $(0, \frac{1}{2})$,

$$[z_k]_j \geq 1 - \sigma \left(\check{u} \left\| W_z \quad U_z \check{x} \quad b_z \right\|_\infty \right) = \underline{z} \tag{A.109b}$$

By similar arguments it is easy to show that

$$|[r_k]_j| \leq \phi(\check{u} \left\| W_r \quad U_r \check{x} \quad b_r \right\|_\infty) = 1 - \underline{\varepsilon}. \tag{A.109c}$$

Let us now take the absolute value of both sides of (A.107). We thus get

$$|[x_{k+1}]_j| \leq [z_k]_j \, |[x_k]_j| + (1 - [z_k]_j)|[r_k]_j|. \tag{A.110}$$

If the component j is such that $|[x_k]_j|\leq 1$, Lemma 3.3 guarantees that $|[x_\tau]_j|\leq 1$ for any $\tau \geq k$. If instead $1 < |[x_k]_j|\leq \check{x}$, by subtracting $|[x_k]_j|$ from both sides of (A.110), we get

$$|[x_{k+1}]_j|-|[x_k]_j|\leq -(1-[z_k]_j)|[x_k]_j|+(1-[z_k]_j)\,|[r_k]_j|. \tag{A.111}$$

Then, recalling that $|[r_k]_j|\leq 1-\underline{\varepsilon}$, since $|[x_k]_j|> 1$ implies $-|[x_k]_j|+1 < 0$, the following chain of inequalities holds

$$\begin{aligned} |[x_{k+1}]_j|-|[x_k]_j| &\leq \big(1-[z_k]_j\big)\,\big(-|[x_{k+1}]_j|+|[r_k]_j|\big) \\ &< \underline{z}\,\left(-|[x_k]_j|+1-\underline{\varepsilon}\right) \\ &< -\underline{z}\,\underline{\varepsilon}. \end{aligned} \tag{A.112}$$

This entails that $|[x_{k+1}]_j|< |[x_k]_j|$. Hence, applying this argument for any state component, it follows that as long as $\|x_k\|_\infty > 1$, $\|x_{k+1}\|_\infty < \|x_k\|_\infty$, which proves the first claim.

To prove the second claim we notice that, by iterating (A.112) over time, we get

$$\sum_{\tau=0}^{k-1}\Big(|[x_{\tau+1}]_j|-|[x_\tau]_j|\Big) \leq -k\underline{z}\underline{\varepsilon}. \tag{A.113}$$

Thus, letting

$$\bar{k}_j = \begin{cases} \left\lceil \frac{|[x_0]_j|-1}{\underline{z}\,\underline{\varepsilon}} \right\rceil & \text{if } |[x_0]_j|> 1 \\ 0 & \text{if } |[x_0]_j|\leq 1 \end{cases}, \tag{A.114}$$

we prove the second claim by taking

$$\bar{k} = \max_j \bar{k}_j \leq \left\lceil \frac{\check{x}-1}{\underline{z}\,\underline{\varepsilon}} \right\rceil. \tag{A.115}$$

Finally, we show that the convergence of each state component $[x_k]_j$ into $[\mathcal{X}]_j = [-1, 1]$ is exponential. To this end, let us write the evolution of (A.107) over the time index k as $[x_k]_j = [x_{a,k}]_j + [x_{b,k}]_j$, where

$$[x_{a,k}]_j = \left(\prod_{\tau=0}^{k-1}[z_\tau]_j\right)[x_0]_j, \tag{A.116a}$$

$$[x_{b,k}]_j = \sum_{\tau=0}^{k-1}\left(\prod_{h=\tau+1}^{k-1}[z_h]_j\right)(1-[z_\tau]_j)\,[r_\tau]_j, \tag{A.116b}$$

where the nonlinearity of the model is buried in the nonlinear dependency of z_k upon $x_k = x_{a,k} + x_{b,k}$ and u_k. Despite this nonlinear dependence, (A.116) can be exploited to characterize the state trajectories of the GRU, as explained below.

First, we point out that the term $[x_{a,k}]_j$ converges to zero. Indeed, owing to the bounds (A.109), by taking the absolute value of both sides of (A.116a) it follows that

$$|[x_{a,k}]_j| \leq \left| \prod_{\tau=0}^{k-1} [z_\tau]_j \right| |[x_0]_j| \leq \bar{z}^k |[x_0]_j|, \tag{A.117}$$

which clearly converges to zero as $k \to \infty$. Concerning $[x_{b,k}]_j$, by taking the absolute values of both sides of taking the absolute value of (A.116b), and applying once more the bounds (A.109), one gets

$$\begin{aligned} |[x_{b,k}]_j| &\leq \left| \sum_{\tau=0}^{k-1} \left[\prod_{h=\tau+1}^{k-1} [z_h]_j - \prod_{h=\tau}^{k-1} [z_h]_j \right] [r_\tau]_j \right| \\ &\leq \left[1 - \prod_{\tau=0}^{k-1} [z_\tau]_j \right] (1 - \underline{\varepsilon}) \\ &\leq \left(1 - \underline{z}^k\right) (1 - \underline{\varepsilon}). \end{aligned} \tag{A.118}$$

By the triangular inequality $|[x_k]_j| \leq |[x_{a,k}]_j| + |[x_{b,k}]_j|$, which leads to

$$|[x_k]_j| \leq \bar{z}^k \, |[x_0]_j| + \left(1 - \underline{z}^k\right) (1 - \underline{\varepsilon}). \tag{A.119}$$

Recalling (A.108), this proves the third and last claim of the Lemma. □

A.2.16 Proof of Lemma 3.5

First, let us compute the bound of the gate z_k. In light of the definition of z_k given in (3.26b), since the sigmoidal activation function is strictly increasing and Lipschitz continuous, the following chain of inequality holds for any component $j \in \{1, ..., n_c\}$

$$\begin{aligned}
[z_k]_j &\leq |[z_k]_j| \leq \|z_k\|_\infty \\
&\leq \left\| \max_{u\in\mathcal{U}} \sigma \left(\begin{bmatrix} W_z & U_z & b_z \end{bmatrix} \begin{bmatrix} u \\ x_k \\ 1_{n_c,1} \end{bmatrix} \right) \right\|_\infty \\
&\leq \max_{u\in\mathcal{U}} \sigma \left(\left\| \begin{bmatrix} W_z & U_z & b_z \end{bmatrix} \begin{bmatrix} u \\ x_k \\ 1_{n_c,1} \end{bmatrix} \right\|_\infty \right) \\
&\leq \max_{u\in\mathcal{U}} \sigma \left(\|W_z \quad U_z\check{x} \quad b_z\|_\infty \left\| \begin{bmatrix} u \\ 1_{n_c,1} \\ 1_{n_c,1} \end{bmatrix} \right\|_\infty \right) \\
&\leq \sigma(\|W_z \quad U_z\check{x} \quad b_z\|_\infty) = \check{\sigma}_z,
\end{aligned} \tag{A.120a}$$

where Assumption 3.1 and Lemma 3.3 have been exploited. Owing to the simmetry of σ with respect to point $(0, \frac{1}{2})$ it is easy to verify that

$$[z_k]_j \geq \sigma(-\|W_z \quad U_z\check{x} \quad b_z\|_\infty) = 1 - \check{\sigma}_z. \tag{A.120b}$$

By applying the same chain of inequalities to f_k, one gets

$$\begin{aligned}
&[f_k]_j \leq \sigma(\|W_f \quad U_f\check{x} \quad b_f\|_\infty) = \check{\sigma}_f, \\
&[f_k]_j \geq \sigma(-\|W_f \quad U_f\check{x} \quad b_f\|_\infty) = 1 - \check{\sigma}_f.
\end{aligned} \tag{A.121}$$

Concerning the squashed input r_k, by noticing that $\check{\sigma}_f < 1$, since ϕ is monotonically increasing and Lipschitz continuous it holds that

$$\begin{aligned}
[r_k]_j &\leq \|r_k\|_\infty \\
&\leq \left\| \max_{u\in\mathcal{U}} \phi \left(\begin{bmatrix} W_r & U_r & b_r \end{bmatrix} \begin{bmatrix} u \\ f_k \circ x_k \\ 1_{n_c,1} \end{bmatrix} \right) \right\|_\infty \\
&\leq \max_{u\in\mathcal{U}} \phi \left(\left\| \begin{bmatrix} W_r & U_r & b_r \end{bmatrix} \begin{bmatrix} u \\ f_k \circ x_k \\ 1_{n_c,1} \end{bmatrix} \right\|_\infty \right) \\
&\leq \max_{u\in\mathcal{U}} \phi \left(\|W_r \quad U_r\check{\sigma}_f\check{x} \quad b_r\|_\infty \left\| \begin{bmatrix} u \\ 1_{n_c,1} \\ 1_{n_c,1} \end{bmatrix} \right\|_\infty \right) \\
&\leq \phi(\|W_r \quad U_r\check{x} \quad b_r\|_\infty) = \check{\phi}_r.
\end{aligned} \tag{A.122a}$$

In light of the simmetry of ϕ with respect to the origin, we can also write

$$[r_k]_j \geq \phi(-\| W_r \quad U_r \check{x} \quad b_r \|_\infty) = -\check{\phi}_r. \tag{A.122b}$$

The Lemma is thus proven. □

A.2.17 Proof of Theorem 3.7

Case a. $x_0 \in \tilde{\mathcal{X}} \subseteq \mathcal{X}$

Consider the j-th component of x_k. From (3.26) it follows that

$$[x_{k+1}]_j = [z_k]_j \, [x_k]_j + (1 - [z_k]_j) \, [r_k]_j.$$

Taking the absolute value, and recalling that $[z_k]_j \in (0, 1)$, the previous equality becomes

$$|[x_{k+1}]_j| \leq [z_k]_j \, |[x_k]_j| + (1 - [z_k]_j) \, |[r_k]_j|. \tag{A.123}$$

Under Assumption 3.1, in light of (3.28), $x_0 \in \tilde{\mathcal{X}}$ implies $\|x_k\|_\infty \leq 1$ for any k. The gates' bounds reported in Remark 3.3 hence apply.

Since by definition $|[v]_j| \leq \|v\|_\infty$, inequality (A.123) can thus be recast as

$$|[x_{k+1}]_j| \leq \|x_{k+1}\|_\infty \leq [z_k]_j \|x_k\|_\infty + (1 - [z_k]_j) \|r_k\|_\infty \tag{A.124}$$

Recalling the definition of r_k in (3.26b), since ϕ is 1-Lipschitz

$$\begin{aligned} \|r_k\|_\infty &\leq \|W_r\|_\infty \|u_k\|_\infty + \|U_r\|_\infty \|f_k\|_\infty \|x_k\|_\infty + \|b_r\|_\infty \\ &\leq \|W_r\|_\infty \|u_k\|_\infty + \|U_r\|_\infty \tilde{\sigma}_f \|x_k\|_\infty + \|b_r\|_\infty. \end{aligned} \tag{A.125}$$

The inequality (A.124) can be thus bounded as

$$\begin{aligned} |[x_{k+1}]_j| \leq & \Big([z_k]_j + (1 - [z_k]_j) \|U_r\|_\infty \tilde{\sigma}_f \Big) \|x_k\|_\infty \\ & + (1 - [z_k]_j) \|W_r\|_\infty \|u_k\|_\infty + (1 - [z_k]_j) \|b_r\|_\infty \end{aligned} \tag{A.126}$$

In light of (3.32a), it holds that $[z_k]_j \in [1 - \tilde{\sigma}_z, \tilde{\sigma}_z] \subset (0, 1)$. Therefore, (3.33) implies that there exists some $\lambda \in (0, 1)$ such that

$$[z_k]_j + (1 - [z_k]_j) \|U_r\|_\infty \tilde{\sigma}_f \leq \lambda.$$

This allows us to re-write (A.126) as

$$\|x_{k+1}\|_\infty \leq \lambda \|x_k\|_\infty + \tilde{\sigma}_z \|W_r\|_\infty \|u_k\|_\infty + \tilde{\sigma}_z \|b_r\|_\infty. \tag{A.127}$$

Iterating (A.127), it is possible to derive that

$$\|x_k\|_\infty \leq \lambda^k \|x_0\|_\infty + \frac{\tilde{\sigma}_z}{1-\lambda} \|W_r\|_\infty \|u_{0:k}\|_{\infty,\infty} + \frac{\tilde{\sigma}_z}{1-\lambda} \|b_r\|_\infty, \quad \text{(A.128)}$$

where the coefficients of $\|u_{0:k}\|_{\infty,\infty}$ and of $\|b_r\|_\infty$ have been majorized by the geometric series' limit, i.e. $\sum_{t=0}^{k-1} \lambda^t \leq \frac{1}{1-\lambda}$. According to Definition 2.4, the system is thus ℓ_∞-ISPS with functions

$$\begin{aligned} \beta(\|x_0\|_\infty, k) &= \lambda^k \|x_0\|_\infty, \\ \gamma(\|u_{0:k}\|_{\infty,\infty}) &= \frac{\tilde{\sigma}_z}{1-\lambda} \|W_r\|_\infty \|u_{0:k}\|_{\infty,\infty}, \\ \varrho &= \frac{\tilde{\sigma}_z}{1-\lambda} \|b_r\|_\infty. \end{aligned}$$

Case b. $\bar{x} \in \mathcal{X} \setminus \tilde{\mathcal{X}}$

In light of Lemma 3.4, the state trajectory x_k converges into the invariant set $\tilde{\mathcal{X}}$ within a finite time instant $\bar{k}$. Since for $k < \bar{k}$ the convergence into $\tilde{\mathcal{X}}$ is exponential regardless of the input sequence applied, for any $\lambda \in (0, 1)$ there exists $\mu > 0$, sufficiently large, such that it holds

$$\|x_k\|_\infty \leq \mu \lambda^k \|x_0\|_\infty. \quad \text{(A.129)}$$

As soon as the state enters the invariant set $\tilde{\mathcal{X}}$, i.e. at $k = \bar{k}$, case *a* applies. Within such set, owing to (A.128), for $k \geq \bar{k}$ it holds that

$$\|x_k\|_\infty \leq \lambda^{k-\bar{k}} \|x_{\bar{k}}\|_\infty + \frac{\tilde{\sigma}_z}{1-\lambda} \|W_r\|_\infty \|u_{\bar{k}:k}\|_{\infty,\infty} + \frac{\tilde{\sigma}_z}{1-\lambda} \|b_r\|_\infty. \quad \text{(A.130)}$$

Noticing that $\|u_{\bar{k}:k}\|_{\infty,\infty} \leq \|u_{0:k}\|_{\infty,\infty}$, by combining (A.129) and (A.130), it follows that system is ℓ_∞-ISPS with functions

$$\beta(\|x_0\|_\infty, k) = \mu \lambda^k \|x_0\|_\infty \quad \text{(A.131a)}$$

$$\gamma(\|u_{0:k}\|_{\infty,\infty}) = \frac{\tilde{\sigma}_z}{1-\lambda} \|W_r\|_\infty \|u_{0:k}\|_{\infty,\infty}, \quad \text{(A.131b)}$$

$$\varrho = \frac{\tilde{\sigma}_z}{1-\lambda} \|b_r\|_\infty. \quad \text{(A.131c)}$$

□

A.2.18 Proof of Theorem 3.8

Consider two state trajectories $x_{a,k}$ and $x_{b,k}$ of the shallow GRU (3.26), defined as $x_{a,k} = x_k(x_{a,0}, u_{a,0:k})$ and $x_{b,k} = x_k(x_{b,0}, u_{b,0:k})$, where $x_{a,0} \in \mathcal{X}$

and $x_{b,0} \in \mathcal{X}$ denote the pair initial state, and $u_{a,0:k} \in \mathcal{U}_{0:k}$ and $u_{b,0:k} \in \mathcal{U}_{0:k}$ denote the pair of input sequences. Let, for the sake of compactness,

$$\begin{aligned} z_{a,k} &= \sigma(W_z u_{a,k} + U_z x_{a,k} + b_z) & z_{b,k} &= \sigma(W_z u_{b,k} + U_z x_{b,k} + b_z), \\ f_{a,k} &= \sigma(W_f u_{a,k} + U_f x_{a,k} + b_f) & f_{b,k} &= \sigma(W_f u_{b,k} + U_f x_{b,k} + b_f), \\ r_{a,k} &= \phi(W_r u_{a,k} + U_r f_{a,k} \circ x_{a,k} + b_r) & r_{b,k} &= \phi(W_r u_{b,k} + U_r f_{b,k} \circ x_{b,k} + b_r). \end{aligned} \tag{A.132}$$

In light of the state equation (3.26a), the j-th component of the difference $x_{a,k+1} - x_{b,k+1}$, with $j \in \{1, ..., n_c\}$, reads as

$$\begin{aligned} [x_{a,k+1}]_j - [x_{b,k+1}]_j = [z_{a,k}]_j [x_{a,k}]_j + (1 - [z_{a,k}]_j)[r_{a,k}]_j \\ - [z_{b,k}]_j [x_{b,k}]_j - (1 - [z_{b,k}]_j)[r_{b,k}]_j \end{aligned} \tag{A.133}$$

By summing and subtracting the terms $[z_{a,k}]_j [x_{b,k}]_j$ and $(1 - [z_{a,k}]_j)[r_{b,k}]_j$, and by trivial manipulations, we get

$$\begin{aligned} [x_{a,k+1}]_j - [x_{b,k+1}]_j = [z_{a,k}]_j \big([x_{a,k}]_j - [x_{b,k}]_j\big) \\ + (1 - [z_{a,k}]_j)\big([r_{a,k}]_j - [r_{b,k}]_j\big) \\ + \big([z_{a,k}]_j - [z_{b,k}]_j\big)[x_{b,k}]_j \\ + \big([z_{a,k}]_j - [z_{b,k}]_j\big)[r_{b,k}]_j. \end{aligned} \tag{A.134}$$

Recalling that $[z_{a,k}]_j \in (0, 1)$, and taking the absolute value of both sides of (A.134), one thus gets

$$\begin{aligned} \big|[x_{a,k+1}]_j - [x_{b,k+1}]_j\big| \leq [z_{a,k}]_j \,\big|[x_{a,k}]_j - [x_{b,k}]_j\big| \\ + (1 - [z_{a,k}]_j)\big|[r_{a,k}]_j - [r_{b,k}]_j\big| \\ + \big|[x_{b,k}]_j\big| \,\big|[z_{a,k}]_j - [z_{b,k}]_j\big| \\ + \big|[r_{b,k}]_j\big| \,\big|[z_{a,k}]_j - [z_{b,k}]_j\big|. \end{aligned} \tag{A.135}$$

Owing to Lemma 3.3, and by its definition (3.27), it holds that

$$|[x_{a,k}]_j| \leq \|x_{a,k}\|_\infty \leq \check{x} \qquad |[x_{b,k}]_j| \leq \|x_{b,k}\|_\infty \leq \check{x}. \tag{A.136a}$$

Moreover, in light of the unity-boundedness of the input, the bounds introduced in Lemma 3.5 hold, i.e.

$$|[z_{a,k}]_j| \leq \|z_{a,k}\|_\infty \leq \check{\sigma}_z \qquad |[z_{b,k}]_j| \leq \|z_{b,k}\|_\infty \leq \check{\sigma}_z, \tag{A.136b}$$

$$|[f_{a,k}]_j| \leq \|f_{a,k}\|_\infty \leq \check{\sigma}_f \qquad |[f_{b,k}]_j| \leq \|f_{b,k}\|_\infty \leq \check{\sigma}_f, \tag{A.136c}$$

$$|[r_{a,k}]_j| \leq \|r_{a,k}\|_\infty \leq \check{\phi}_r \qquad |[r_{b,k}]_j| \leq \|r_{b,k}\|_\infty \leq \check{\phi}_r, \tag{A.136d}$$

where $\check{\sigma}_z$, $\check{\sigma}_f$, and $\check{\phi}_r$ are defined in (3.30).

In order to bound the term $|[z_{a,k}]_j - [z_{b,k}]_j|$, we recall that the sigmoidal activation function is $\frac{1}{4}$-Lipschitz. Therefore, by standard norms arguments,

$$\begin{aligned} |[z_{a,k}]_j - [z_{b,k}]_j| &\leq \|z_{a,k} - z_{b,k}\|_\infty \\ &\leq \frac{1}{4}\big\|W_z(u_{a,k} - u_{b,k}) + U_z(x_{a,k} - x_{b,k})\big\|_\infty \\ &\leq \frac{1}{4}\|W_z\|_\infty \|u_{a,k} - u_{b,k}\|_\infty + \frac{1}{4}\|U_z\|_\infty \|x_{a,k} - x_{b,k}\|_\infty. \end{aligned} \tag{A.137}$$

Similarly, one gets that

$$\begin{aligned} |[f_{a,k}]_j - [f_{b,k}]_j| &\leq \|f_{a,k} - f_{b,k}\|_\infty \\ &\leq \frac{1}{4}\|W_f\|_\infty \|u_{a,k} - u_{b,k}\|_\infty + \frac{1}{4}\|U_f\|_\infty \|x_{a,k} - x_{b,k}\|_\infty. \end{aligned} \tag{A.138}$$

We now aim to bound the term $|[r_{a,k}]_j - [r_{b,k}]_j|$. Since ϕ is 1-Lipschitz, and in light of the bounds (A.136),(A.137), and (A.138), the following chain of inequalities holds

$$\begin{aligned} \big|[r_{a,k}]_j - [r_{b,k}]_j\big| &\leq \big\|r_{a,k} - r_{b,k}\big\|_\infty \\ &\leq \|W_r(u_{a,k} - u_{b,k}) + U_r(f_{a,k} \circ x_{a,k} - f_{b,k} \circ x_{b,k})\|_\infty \\ &\leq \|W_r\|_\infty \|u_{a,k} - u_{b,k}\|_\infty \\ &\qquad + \|U_r\|_\infty \|(f_{a,k} - f_{b,k}) \circ x_{a,k} + f_{b,k} \circ (x_{a,k} - x_{a,k})\|_\infty \\ &\leq \|W_r\|_\infty \|u_{a,k} - u_{b,k}\|_\infty \\ &\qquad + \|U_r\|_\infty \big(\|f_{a,k} - f_{b,k}\|_\infty \|x_{a,k}\|_\infty + \|f_{b,k}\|_\infty \|x_{a,k} - x_{b,k}\|_\infty\big) \\ &\leq \|W_r\|_\infty \|u_{a,k} - u_{b,k}\|_\infty + \frac{1}{4}\check{x}\|U_r\|_\infty \|W_f\|_\infty \|u_{a,k} - u_{b,k}\|_\infty \\ &\qquad + \frac{1}{4}\check{x}\|U_r\|_\infty \|U_f\|_\infty \|x_{a,k} - x_{b,k}\|_\infty \\ &\qquad + \check{\sigma}_f \|U_r\|_\infty \|x_{a,k} - x_{b,k}\|_\infty \\ &\leq \|U_r\|_\infty \left(\frac{1}{4}\check{x}\|U_f\|_\infty + \check{\sigma}_f\right) \|x_{a,k} - x_{b,k}\|_\infty \\ &\qquad + \left(\|W_r\|_\infty + \frac{1}{4}\check{x}\|U_r\|_\infty \|W_f\|_\infty\right) \|u_{a,k} - u_{b,k}\|_\infty. \end{aligned} \tag{A.139}$$

Applying the bounds (A.136)-(A.139) to the inequality (A.135) we obtain

$$|[x_{a,k+1}]_j - [x_{b,k+1}]_j| \leq \kappa_x \|x_{a,k} - x_{b,k}\|_\infty + \kappa_u \|u_{a,k} - u_{b,k}\|_\infty, \tag{A.140a}$$

where κ_x and κ_u are defined as

$$\begin{aligned}\kappa_x =&[z_{a,k}]_j + \frac{1}{4}(\check{x} + \check{\phi}_r)\|U_z\|_\infty + (1 - [z_{a,k}]_j)\|U_r\|_\infty\Big(\frac{1}{4}\check{x}\|U_f\|_\infty + \check{\sigma}_f\Big),\\ \kappa_u =&\frac{1}{4}(\check{x} + \check{\phi}_r)\|W_z\|_\infty + (1 - [z_{a,k}]_j)\Big(\|W_r\|_\infty + \frac{1}{4}\check{x}\|U_r\|_\infty\|W_f\|_\infty\Big).\end{aligned} \tag{A.140b}$$

We now show that the condition (3.34) implies the existence of some $\lambda_\delta \in (0, 1)$ such that $0 \leq \kappa_x \leq \lambda_\delta < 1$, and that this, in turn, implies the δISS of the system. To this end, let us consider the inequality $\kappa_x < 1$, i.e.

$$[z_{a,k}]_j + \frac{1}{4}(\check{x} + \check{\phi}_r)\|U_z\|_\infty + (1 - [z_{a,k}]_j)\|U_r\|_\infty\Big(\frac{1}{4}\check{x}\|U_f\|_\infty + \check{\sigma}_f\Big) < 1. \tag{A.141}$$

Moving $[z_{a,k}]_j$ to the right-hand side, and dividing by $1 - [z_{a,k}]_j$ (which is surely positive) one obtains

$$\frac{1}{4}\frac{\check{x} + \check{\phi}_r}{1 - [z_{a,k}]_j}\|U_z\|_\infty + \|U_r\|_\infty\Big(\frac{1}{4}\check{x}\|U_f\|_\infty + \check{\sigma}_f\Big) < 1. \tag{A.142}$$

We now observe that, since $[z_{a,k}]_j \in (1 - \check{\sigma}_z, \check{\sigma}_z)$, (A.142) surely holds if the following condition is fulfilled

$$\frac{1}{4}\frac{\check{x} + \check{\phi}_r}{1 - \check{\sigma}_z}\|U_z\|_\infty + \|U_r\|_\infty\Big(\frac{1}{4}\check{x}\|U_f\|_\infty + \check{\sigma}_f\Big) < 1, \tag{A.143}$$

which is equivalent to condition (3.34).

Therefore, since (A.140) holds for any component j, for any $[z_{a,k}]_j$ it holds that $\kappa_x \leq \lambda_\delta < 1$, we can write

$$\|x_{a,k+1} - x_{b,k+1}\|_\infty \leq \lambda_\delta\|x_{a,k} - x_{b,k}\|_\infty + \check{\kappa}_u\|u_{a,k} - u_{b,k}\|_\infty, \tag{A.144a}$$

where $\check{\kappa}_u$ is the supremum of κ_u, computed as

$$\check{\kappa}_u = \frac{1}{4}(\check{x} + \check{\phi}_r)\|W_z\|_\infty + \check{\sigma}_z\Big(\|W_r\|_\infty + \frac{1}{4}\check{x}\|U_r\|_\infty\|W_f\|_\infty\Big). \tag{A.144b}$$

By iterating (A.144a) over the time index k, it follows that

$$\begin{aligned}\|x_{a,k} - x_{b,k}\|_\infty \leq& \lambda_\delta^k\|x_{a,0} - x_{b,0}\|_\infty + \sum_{\tau=0}^{k-1}\lambda_\delta^{k-\tau-1}\check{\kappa}_u\|u_{a,\tau} - u_{b,\tau}\|_\infty\\ \leq& \lambda_\delta^k\|x_{a,0} - x_{b,0}\|_\infty + \frac{\check{\kappa}_u}{1 - \lambda_\delta}\|u_{a,0:k} - u_{b,0:k}\|_{\infty,\infty},\end{aligned} \tag{A.145}$$

where the geometric series limit has been exploited to bound the second term. Therefore, according to Definition 2.6, the system is ℓ_∞-δISS with functions

$$\beta(\|x_{a,0} - x_{b,0}\|_\infty, k) = \lambda_\delta^k \|x_{a,0} - x_{b,0}\|_\infty, \tag{A.146a}$$

$$\gamma(\|u_{a,0:k} - u_{b,0:k}\|_{\infty,\infty}) = \frac{\breve{\kappa}_u}{1-\lambda_\delta}\|u_{a,0:k} - u_{b,0:k}\|_{\infty,\infty}. \tag{A.146b}$$

□

A.2.19 Proof of Proposition 3.6

By definition of $\breve{\sigma}_z$ and $\tilde{\sigma}_z$, see (3.30) and (3.31), it follows that

$$0 < 1 - \breve{\sigma}_z \leq 1 - \tilde{\sigma}_z \leq \tilde{\sigma}_z \leq \breve{\sigma}_z < 1.$$

The same relationship holds between $\breve{\sigma}_f$ and $\tilde{\sigma}_f$. Hence, since $\|U_z\|_\infty \geq 0$ and $\breve{x} \geq 1$, from (3.34) and (3.35) it follows that

$$\begin{aligned}\|U_r\|_\infty\Big(\frac{1}{4}\|U_f\|_\infty + \tilde{\sigma}_f\Big) &\leq \|U_r\|_\infty\Big(\frac{1}{4}\breve{x}\|U_f\|_\infty + \breve{\sigma}_f\Big) \\ &< 1 - \frac{1}{4}\frac{\breve{x} + \breve{\phi}_r}{1 - \breve{\sigma}_z}\|U_z\|_\infty \\ &\leq 1 - \frac{1}{4}\frac{1 + \tilde{\phi}_r}{1 - \tilde{\sigma}_z}\|U_z\|_\infty < 1.\end{aligned} \tag{A.147}$$

Noting that $\frac{1}{4}\|U_f\|_\infty \geq 0$, (A.147) entails that

$$\|U_r\|_\infty \, \tilde{\sigma}_f < 1, \tag{A.148}$$

which is the ISPS condition (3.33). □

A.2.20 Proof of Lemma 3.6

Consider the first layer, $l = 1$. Since $x_k^{(1)} \in \mathcal{X}^{(1)}$, Lemma 3.3 implies that $\mathcal{X}^{(1)}$ is an invariant set of the state component $x_k^{(1)}$. The second layer, $l = 2$, is thus characterized by an input bounded by $\breve{x}^{(1)}$. Since $x_k^{(2)} \in \mathcal{X}^{(2)}$, Lemma 3.3 entails that $\mathcal{X}^{(2)}$ is an invariant set of $x_k^{(2)}$. Iterating this argument, we show that $\forall l \in \{1, ..., L\}$, $\mathcal{X}^{(l)}$ is the invariant set of $x_k^{(l)}$. By recalling (3.36b) and (3.37a), we can conclude that the Cartesian product of these sets, i.e. $\mathcal{X}$, is an invariant set of the state vector x_k. □

A.2.21 Proof of Lemma 3.7

The strategy adopted to prove this Lemma is to show that Lemma 3.4 can be applied layer-wise.

Let us therefore consider the first layer ($l = 1$). Then, since $u_k^{(1)} = u_k$ is finite, Lemma 3.4 can be straightforwardly applied. Concerning the second layer ($l = 2$), as pointed out in (3.38), its input is bounded as $\|u_k^{(2)}\|_\infty \leq \breve{x}^{(1)}$. Lemma 3.4 can therefore be applied to the second layer using such input bound. Iterating this argument to all layers we can easily prove all the claims of this Lemma.

Indeed, from the first claim of Lemma 3.4 we get that if $x_0^{(l)} \in \mathcal{X}^{(l)} \setminus \tilde{\mathcal{X}}^{(l)}$, then $\|x_k^{(l)}\|_\infty$ is strictly decreasing until $x_k^{(l)} \in \tilde{\mathcal{X}}^{(l)}$. By noting that $\|x_k\|_\infty = \max_l \|x_k^{(l)}\|_\infty$, this implies that $\|x_k\|_\infty$ is strictly decreasing until $x_k \in \tilde{\mathcal{X}}$, which proves the first claim.

From the second claim of Lemma 3.4 we get that the state of each layer $l \in \{1, ..., L\}$ converges into $\tilde{\mathcal{X}}^{(l)}$ in finite time, i.e., within $\bar{k}^{(l)}$ steps. Hence, letting $\bar{k} = \max_l \bar{k}^{(l)}$, for any $k \geq \bar{k}$ it surely holds that $x_k \in \tilde{\mathcal{X}}$, which proves the second claim.

Lastly, in light of (3.36b), the third claim of Lemma 3.4 straightforwardly implies the third claim of Lemma 3.7. □

A.2.22 Proof of Lemma 3.8

Let us consider the generic layer $l \in \{1, ..., L\}$. We observe that, in view of (3.38), it holds that $\|u_k^{(l)}\|_\infty \leq \breve{x}^{(l-1)}$, where for consistency we denote $\|u_k^{(1)}\|_\infty = \|u_k\|_\infty \leq \breve{x}^{(0)} = 1$, see Assumption 3.1. Therefore, since σ is strictly increasing and Lipschitz-continuous, each component $j \in \{1, ..., n_c^{(l)}\}$ can be bounded as follows

$$\begin{aligned}
[z_k^{(l)}]_j &\leq |[z_k^{(l)}]_j| \leq \|z_k^{(l)}\|_\infty \\
&\leq \left\| \max_{u: \|u\|_\infty \leq \breve{x}^{(l-1)}} \sigma \left(\begin{bmatrix} W_z^{(l)} & U_z^{(l)} & b_z^{(l)} \end{bmatrix} \begin{bmatrix} u \\ x_k^{(l)} \\ 1_{n_c^{(l)},1} \end{bmatrix} \right) \right\|_\infty \\
&\leq \max_{\tilde{u}: \|\tilde{u}\|_\infty \leq 1} \sigma \left(\| W_z^{(l)} \breve{x}^{(l-1)} \quad U_z^{(l)} \breve{x}^{(l)} \quad b_z^{(l)} \|_\infty \left\| \begin{bmatrix} \tilde{u} \\ 1_{n_c^{(l)},1} \\ 1_{n_c^{(l)},1} \end{bmatrix} \right\|_\infty \right) \\
&\leq \sigma(\| W_z^{(l)} \breve{x}^{(l-1)} \quad U_z^{(l)} \breve{x}^{(l)} \quad b_z^{(l)} \|_\infty) = \breve{\sigma}_z^{(l)},
\end{aligned} \tag{A.149a}$$

where Lemma 3.6 has also been exploited. Moreover, since σ is symmetric with respect to point $(0, \frac{1}{2})$,

$$[z_k^{(l)}]_j \geq \sigma(-\|W_z^{(l)}\check{x}^{(l-1)} \quad U_z^{(l)}\check{x}^{(l)} \quad b_z^{(l)}\|_\infty) = 1 - \check{\sigma}_z^{(l)}. \tag{A.149b}$$

By similar arguments, the gate $f_k^{(l)}$ can be bounded as

$$\begin{aligned} [f_k^{(l)}]_j &\leq \sigma(\|W_f^{(l)}\check{x}^{(l-1)} \quad U_f^{(l)}\check{x}^{(l)} \quad b_f^{(l)}\|_\infty) = \check{\sigma}_f^{(l)}, \\ [f_k^{(l)}]_j &\geq \sigma(-\|W_f^{(l)}\check{x}^{(l-1)} \quad U_f^{(l)}\check{x}^{(l)} \quad b_f^{(l)}\|_\infty) = 1 - \check{\sigma}_f^{(l)}. \end{aligned} \tag{A.150}$$

Therefore, being $\|f_k^{(l)}\|_\infty \leq \check{\sigma}_f^{(l)}$, the following chain of inequalities holds

$$\begin{aligned} [r_k^{(l)}]_j &\leq \|r_k^{(l)}\|_\infty \\ &\leq \left\| \max_{u:\|u\|_\infty \leq \check{x}^{(l-1)}} \phi\left([W_r^{(l)} \quad U_r^{(l)} \quad b_r^{(l)}] \begin{bmatrix} u \\ f_k^{(l)} \circ x_k^{(l)} \\ 1_{n_c^{(l)},1} \end{bmatrix} \right) \right\|_\infty \\ &\leq \max_{\tilde{u}:\|\tilde{u}\|_\infty \leq 1} \phi\left(\|W_r^{(l)}\check{x}^{(l)} \quad U_r^{(l)}\check{\sigma}_f^{(l)}\check{x}^{(l)} \quad b_r^{(l)}\|_\infty \left\| \begin{bmatrix} \tilde{u} \\ 1_{n_c^{(l)},1} \\ 1_{n_c^{(l)},1} \end{bmatrix} \right\|_\infty \right) \\ &\leq \phi(\|W_r^{(l)}\check{x}^{(l-1)} \quad U_r^{(l)}\check{x}^{(l)} \quad b_r^{(l)}\|_\infty) = \check{\phi}_r^{(l)}. \end{aligned} \tag{A.151a}$$

In light of the simmetry of ϕ with respect to the origin, we can also write

$$[r_k^{(l)}]_j \geq \phi(-\|W_r^{(l)}\check{x}^{(l-1)} \quad U_r^{(l)}\check{x}^{(l)} \quad b_r^{(l)}\|_\infty) = -\check{\phi}_r^{(l)}. \tag{A.151b}$$

□

A.2.23 Proof of Theorem 3.9

Consider the generic layer $l \in \{1, ..., L\}$. We observe that, in light of the definition of $u_k^{(l)}$, owing to Lemma 3.6 and Assumption 3.1,

$$u_k^{(l)} \in \mathcal{U}^{(l)} = \begin{cases} \mathcal{U} & \text{if } l = 1, \\ \mathcal{X}^{(l-1)} & \text{if } l \in \{2, ..., L\}. \end{cases} \tag{A.152}$$

That is, $\|u_k^{(l)}\|_\infty \leq \check{x}^{(l-1)}$, where for the sake of consistency, we denote $\|u_k^{(1)}\|_\infty = \|u_k\|_\infty \leq \check{x}^{(0)} = 1$. Along the lines of the proof of Theorem 3.7, we prove this theorem in two steps.

Case a. $x_0 \in \tilde{\mathcal{X}} \subseteq \mathcal{X}$

Since $\|x_0\|_\infty = \max_{l \in \{1,...,L\}} \|x_0^{(l)}\|_\infty \le 1$, Remark 3.5 applies, i.e. the gates of each layer can be bounded as in (3.42)-(3.43). Hence, along the lines of *Case a* of Theorem 3.7's proof, by adopting a suitable notation we get that

$$|[x_{k+1}^{(l)}]_j| \le \big([z_k^{(l)}]_j + (1-[z_k^{(l)}]_j)\|U_r^{(l)}\|_\infty \tilde{\sigma}_f^{(l)}\big)\|x_k^{(l)}\|_\infty \\ + (1-[z_k^{(l)}]_j)\|W_r^{(l)}\|_\infty \|u_k^{(l)}\|_\infty + (1-[z_k^{(l)}]_j)\|b_r^{(l)}\|_\infty \tag{A.153}$$

for any component $j \in \{1, ..., n_c^{(l)}\}$. Owing to (3.43a) we can write $[z_k^{(l)}]_j \in [1-\tilde{\sigma}_z^{(l)}, \tilde{\sigma}_z^{(l)}] \subset (0,1)$. The fulfillment of condition (3.44) thus implies the existence of $\lambda^{(l)} \in (0,1)$ such that

$$\|x_{k+1}^{(l)}\|_\infty \le \lambda^{(l)}\|x_k^{(l)}\|_\infty + \tilde{g}_u^{(l)}\|u_k^{(l)}\|_\infty + \tilde{g}_b^{(l)}\|b_r^{(l)}\|_\infty, \tag{A.154a}$$

where, for compactness,

$$\begin{aligned} \tilde{g}_u^{(l)} &= \tilde{\sigma}_z^{(l)}\|W_r^{(l)}\|_\infty, \\ \tilde{g}_b^{(l)} &= \tilde{\sigma}_z^{(l)}. \end{aligned} \tag{A.154b}$$

Since (A.154a) holds for each layer, recalling the definition of $u_k^{(l)}$ we can write

$$\begin{bmatrix} \|x_{k+1}^{(1)}\|_\infty \\ \|x_{k+1}^{(2)}\|_\infty \\ \vdots \\ \|x_{k+1}^{(L)}\|_\infty \end{bmatrix} \le \mathfrak{H}_L \begin{bmatrix} \|x_k^{(1)}\|_\infty \\ \|x_k^{(2)}\|_\infty \\ \vdots \\ \|x_k^{(L)}\|_\infty \end{bmatrix} + \mathfrak{G}_{Lu}\|u_k\|_\infty + \mathfrak{G}_{Lb} \begin{bmatrix} \|b_r^{(1)}\|_\infty \\ \|b_r^{(2)}\|_\infty \\ \vdots \\ \|b_r^{(L)}\|_\infty \end{bmatrix}, \tag{A.155a}$$

where, $\mathfrak{H}_L$, $\mathfrak{G}_{Lu}$, and $\mathfrak{G}_{Lu}$ read as

$$\mathfrak{H}_L = \begin{bmatrix} \lambda^{(1)} & 0 & \dots & 0 \\ \tilde{g}_u^{(2)}\lambda^{(1)} & \lambda^{(2)} & \dots & 0 \\ \vdots & & \ddots & \vdots \\ \big(\prod_{j=L}^{2}\tilde{g}_u^{(j)}\big)\lambda^{(1)} & \big(\prod_{j=L}^{3}\tilde{g}_u^{(j)}\big)\lambda^{(2)} & \dots & \lambda^{(L)} \end{bmatrix}, \tag{A.155b}$$

$$\mathfrak{G}_{Lu} = \begin{bmatrix} \tilde{g}_u^{(1)} \\ \tilde{g}_u^{(2)} \tilde{g}_u^{(1)} \\ \vdots \\ \prod_{j=L-1}^{1} \tilde{g}_u^{(j)} \end{bmatrix}, \quad \mathfrak{G}_{Lb} = \begin{bmatrix} \tilde{g}_u^{(1)} & 0 & \dots & 0 \\ \tilde{g}_u^{(2)} \tilde{g}_u^{(1)} & \tilde{g}_u^{(2)} & \dots & 0 \\ \vdots & & \ddots & \vdots \\ \prod_{j=L}^{1} \tilde{g}_u^{(j)} & \prod_{j=L}^{2} \tilde{g}_u^{(j)} & \dots & \tilde{g}_u^{(L)} \end{bmatrix}. \tag{A.155c}$$

By iterating (A.155) over the time index k we obtain

$$\begin{bmatrix} \|x_k^{(1)}\|_\infty \\ \|x_k^{(2)}\|_\infty \\ \vdots \\ \|x_k^{(L)}\|_\infty \end{bmatrix} \leq \mathfrak{H}_L^k \begin{bmatrix} \|x_0^{(1)}\|_\infty \\ \|x_0^{(2)}\|_\infty \\ \vdots \\ \|x_0^{(L)}\|_\infty \end{bmatrix} + \sum_{\tau=0}^{k-1} \mathfrak{H}_L^{\tau-k-1} \left(\mathfrak{G}_{Lu} \|u_\tau\|_\infty + \mathfrak{G}_{Lb} \begin{bmatrix} \|b_r^{(1)}\|_\infty \\ \|b_r^{(2)}\|_\infty \\ \vdots \\ \|b_r^{(L)}\|_\infty \end{bmatrix} \right). \tag{A.156}$$

At this stage, let us notice that

$$\left\| \begin{bmatrix} \|x_k^{(1)}\|_\infty \\ \|x_k^{(2)}\|_\infty \\ \vdots \\ \|x_k^{(L)}\|_\infty \end{bmatrix} \right\|_\infty = \|x_k\|_\infty, \qquad \left\| \begin{bmatrix} \|b_r^{(1)}\|_\infty \\ \|b_r^{(2)}\|_\infty \\ \vdots \\ \|b_r^{(L)}\|_\infty \end{bmatrix} \right\|_\infty = \|b_r\|_\infty,$$

where $b_r = \left[b_r^{(1)\prime}, ..., b_r^{(L)\prime}\right]'$. Then, since $\mathfrak{H}_L$ is diagonal, the elements on the main diagonal are the eigenvalues of the matrix, and thus $\mathfrak{H}_L$ is Schur stable. This implies that there exist $\mu_{\mathfrak{H}_L} > 0$ such that, letting $\lambda_{\mathfrak{H}_L} = \max_{l \in \{1,...,L\}} \lambda^{(l)} \in (0,1)$, from (A.156) it follows that

$$\begin{aligned} \|x_k\|_\infty \leq \mu_{\mathfrak{H}_L} \lambda_{\mathfrak{H}_L}^k \|x_0\|_\infty &+ \|(I_{L,L} - \mathfrak{H}_L)^{-1} \mathfrak{G}_{Lu}\|_\infty \|u_{0:k}\|_{\infty,\infty} \\ &+ \|(I_{L,L} - \mathfrak{H}_L)^{-1} \mathfrak{G}_{Lb}\|_\infty \|b_r\|_\infty. \end{aligned} \tag{A.157}$$

According to Definition 2.4, this implies that the system is ℓ_∞-ISPS with

$$\begin{aligned} \beta(\|x_0\|_\infty, k) &= \mu_{\mathfrak{H}_L} \lambda_{\mathfrak{H}_L}^k \|x_0\|_\infty \\ \gamma(\|u_{0:k}\|_\infty) &= \|(I_{L,L} - \mathfrak{H}_L)^{-1} \mathfrak{G}_{Lu}\|_\infty \|u_{0:k}\|_{\infty,\infty} \\ \varrho &= \|(I_{L,L} - \mathfrak{H}_L)^{-1} \mathfrak{G}_{Lb}\|_\infty \|b_r\|_\infty. \end{aligned}$$

Case b. $x_0 \in \mathcal{X} \setminus \tilde{\mathcal{X}}$

In light of Lemma 3.7, each component of $[x_k^{(l)}]_j$ into $[-1, 1]$ is exponential and happens in a finite number of time-steps, denoted by $\bar{k}$, regardless of the input sequence applied. Therefore, for any $\lambda_{\mathfrak{H}_L} \in (0,1)$ there exists a sufficiently large $\mu > 0$ such that, $\forall k < \bar{k}$,

$$\|x_k\|_\infty \leq \mu \lambda_{\mathfrak{H}_L}^k \|x_0\|_\infty. \tag{A.158}$$

Then, for any $k \geq \bar{k}$, since $x_k \in \tilde{\mathcal{X}}$, *Case a* applies, i.e.

$$\begin{aligned} \|x_k\|_\infty \leq \mu_{\mathfrak{H}_L} \lambda_{\mathfrak{H}_L}^{k-\bar{k}} \|x_{\bar{k}}\|_\infty + \|(I_{L,L} - \mathfrak{H}_L)^{-1} \mathfrak{G}_{Lu}\|_\infty \|u_{\bar{k}:k}\|_\infty \\ + \|(I_{L,L} - \mathfrak{H}_L)^{-1} \mathfrak{G}_{Lb}\|_\infty \|b_r\|_\infty. \end{aligned} \tag{A.159}$$

Thus, noticing that $\|u_{\bar{k}:k}\|_{\infty,\infty} \leq \|u_{0:k}\|_{\infty,\infty}$, and letting $\tilde{\mu}_{\mathfrak{H}_L} = \max(\mu_{\mathfrak{H}_L}, \mu)$, by combining (A.158) and (A.159) we get that the deep GRU is ℓ_∞-ISPS with functions

$$\beta(\|x_0\|_\infty, k) = \tilde{\mu}_{\mathfrak{H}_L} \lambda_{\mathfrak{H}_L}^k \|x_0\|_\infty \tag{A.160a}$$

$$\gamma(\|u_{0:k}\|_\infty) = \|(I_{L,L} - \mathfrak{H}_L)^{-1} \mathfrak{G}_{Lu}\|_\infty \|u_{0:k}\|_{\infty,\infty} \tag{A.160b}$$

$$\varrho = \|(I_{L,L} - \mathfrak{H}_L)^{-1} \mathfrak{G}_{Lb}\|_\infty \|b_r\|_\infty. \tag{A.160c}$$

□

A.2.24 Proof of Theorem 3.10

Consider two initial states $x_{a,0} \in \mathcal{X}$ and $x_{b,0} \in \mathcal{X}$, and two input sequences $u_{a,0:k} \in \mathcal{U}_{0:k}$ and $u_{b,0:k} \in \mathcal{U}_{0:k}$. The goal is to show that Definition 2.6 applies. To this end, consider the generic layer $l \in \{1, ..., L\}$. Along the lines of the proof reported in Appendix A.2.18, we define

$$\begin{aligned} z_{a,k}^{(l)} &= \sigma(W_z^{(l)} u_{a,k}^{(l)} + U_z^{(l)} x_{a,k}^{(l)} + b_z^{(l)}), \\ z_{b,k}^{(l)} &= \sigma(W_z^{(l)} u_{b,k}^{(l)} + U_z^{(l)} x_{b,k}^{(l)} + b_z^{(l)}), \\ f_{a,k}^{(l)} &= \sigma(W_f^{(l)} u_{a,k}^{(l)} + U_f^{(l)} x_{a,k}^{(l)} + b_f^{(l)}), \\ f_{b,k}^{(l)} &= \sigma(W_f^{(l)} u_{b,k}^{(l)} + U_f^{(l)} x_{b,k}^{(l)} + b_f^{(l)}), \\ r_{a,k}^{(l)} &= \phi(W_r^{(l)} u_{a,k}^{(l)} + U_r^{(l)} f_{a,k}^{(l)} \circ x_{a,k}^{(l)} + b_r^{(l)}), \\ r_{b,k}^{(l)} &= \phi(W_r^{(l)} u_{b,k}^{(l)} + U_r^{(l)} f_{b,k}^{(l)} \circ x_{b,k}^{(l)} + b_r^{(l)}). \end{aligned} \tag{A.161}$$

We recall Lemma 3.8, in which the bounds to the gates (A.161), denoted as $\check{\sigma}_z^{(l)}$, $\check{\sigma}_f^{(l)}$, and $\check{\phi}_z^{(l)}$, are established, see (3.40).

Hence, by following the steps illustrated in (A.133)-(A.140), one can easily get the following inequality holding $\forall j \in \{1, ..., n_c^{(l)}\}$

$$|[x_{a,k+1}^{(l)}]_j - [x_{b,k+1}^{(l)}]_j| \leq \kappa_x^{(l)} \|x_{a,k}^{(l)} - x_{b,k}^{(l)}\|_\infty + \kappa_u^{(l)} \|u_{a,k}^{(l)} - u_{b,k}^{(l)}\|_\infty, \tag{A.162a}$$

where $\kappa_x^{(l)}$ and $\kappa_u^{(l)}$ are defined as

$$\begin{aligned}\kappa_x^{(l)} =& [z_{a,k}^{(l)}]_j + \frac{1}{4}(\check{x}^{(l)} + \check{\phi}_r^{(l)})\|U_z^{(l)}\|_\infty \\ &\qquad + (1-[z_{a,k}^{(l)}]_j)\|U_r^{(l)}\|_\infty\Big(\frac{1}{4}\check{x}^{(l)}\|U_f^{(l)}\|_\infty + \check{\sigma}_f^{(l)}\Big), \\ \kappa_u^{(l)} =& \frac{1}{4}(\check{x}^{(l)} + \check{\phi}_r^{(l)})\|W_z^{(l)}\|_\infty \\ &\qquad + (1-[z_{a,k}^{(l)}]_j)\Big(\|W_r^{(l)}\|_\infty + \frac{1}{4}\check{x}^{(l)}\|U_r^{(l)}\|_\infty\|W_f^{(l)}\|_\infty\Big).\end{aligned} \tag{A.162b}$$

As shown in (A.141)-(A.143), if condition (3.45) is satisfied, there exists $\lambda_\delta^{(l)} \in (0,1)$ such that

$$0 < \kappa_x^{(l)} \leq \lambda_\delta^{(l)} < 1. \tag{A.163a}$$

Moreover, in light of (3.40a), by letting

$$\check{\kappa}_u^{(l)} = \frac{1}{4}(\check{x}^{(l)} + \check{\phi}_r^{(l)})\|W_z^{(l)}\|_\infty + \check{\sigma}_z^{(l)}\Big(\|W_r^{(l)}\|_\infty + \frac{1}{4}\check{x}^{(l)}\|U_r^{(l)}\|_\infty\|W_f^{(l)}\|_\infty\Big), \tag{A.163b}$$

it holds that

$$\kappa_u^{(l)} \leq \check{\kappa}_u^{(l)}. \tag{A.163c}$$

Applying (A.163) to (A.162) one gets that

$$\|x_{a,k+1}^{(l)} - x_{b,k+1}^{(l)}\|_\infty \leq \lambda_\delta^{(l)}\|x_{a,k}^{(l)} - x_{b,k}^{(l)}\|_\infty + \check{\kappa}_u^{(l)}\|u_{a,k}^{(l)} - u_{b,k}^{(l)}\|_\infty. \tag{A.164}$$

In light of the input vector definition (3.36c), by iterating (A.164) over $l \in \{1, ..., L\}$, we get

$$\begin{bmatrix} \|x_{a,k+1}^{(1)} - x_{b,k+1}^{(1)}\|_\infty \\ \|x_{a,k+1}^{(2)} - x_{b,k+1}^{(2)}\|_\infty \\ \vdots \\ \|x_{a,k+1}^{(L)} - x_{b,k+1}^{(L)}\|_\infty \end{bmatrix} \leq \mathfrak{H}_{L\delta} \begin{bmatrix} \|x_{a,k}^{(1)} - x_{b,k}^{(1)}\|_\infty \\ \|x_{a,k}^{(2)} - x_{b,k}^{(2)}\|_\infty \\ \vdots \\ \|x_{a,k}^{(L)} - x_{b,k}^{(L)}\|_\infty \end{bmatrix} + \mathfrak{G}_{L\delta}\|u_{a,k} - u_{b,k}\|_\infty, \tag{A.165a}$$

where $\mathfrak{H}_{L\delta}$ and $\mathfrak{G}_{L\delta}$ are defined as

$$\mathfrak{H}_{L\delta} = \begin{bmatrix} \lambda_\delta^{(1)} & 0 & \dots & 0 \\ \check{\kappa}_u^{(2)}\lambda_\delta^{(1)} & \lambda_\delta^{(2)} & \dots & 0 \\ \vdots & & \ddots & \vdots \\ \big(\prod_{j=L}^{2}\check{\kappa}_u^{(j)}\big)\lambda_\delta^{(1)} & \big(\prod_{j=L}^{3}\check{\kappa}_u^{(j)}\big)\lambda_\delta^{(2)} & \dots & \lambda_\delta^{(L)} \end{bmatrix}, \tag{A.165b}$$

$$\mathfrak{G}_{L\delta} = \begin{bmatrix} \check{\kappa}_u^{(1)} \\ \check{\kappa}_u^{(2)} \check{\kappa}_u^{(1)} \\ \vdots \\ \prod_{j=L-1}^{1} \check{\kappa}_u^{(j)} \end{bmatrix}. \tag{A.165c}$$

Then, by iterating (A.165) over the time index k, we obtain

$$\begin{bmatrix} \|x_{a,k}^{(1)} - x_{b,k}^{(1)}\|_\infty \\ \|x_{a,k}^{(2)} - x_{b,k}^{(2)}\|_\infty \\ \vdots \\ \|x_{a,k}^{(L)} - x_{b,k}^{(L)}\|_\infty \end{bmatrix} \leq \mathfrak{H}_{L\delta}^k \begin{bmatrix} \|x_{a,0}^{(1)} - x_{b,0}^{(1)}\|_\infty \\ \|x_{a,0}^{(2)} - x_{b,0}^{(2)}\|_\infty \\ \vdots \\ \|x_{a,0}^{(L)} - x_{b,0}^{(L)}\|_\infty \end{bmatrix} + \sum_{\tau=0}^{k-1} \mathfrak{H}_{L\delta}^{k-\tau-1} \mathfrak{G}_{L\delta} \|u_{a,\tau} - u_{b,\tau}\|_\infty. \tag{A.166}$$

Being triangular, the matrix $\mathfrak{H}_{L\delta}$ has $\lambda_\delta^{(1)}$, ..., $\lambda_\delta^{(L)}$ as eigenvalues, and hence it is Schur stable, since $\lambda_\delta^{(l)} \in (0,1)$. This implies that, letting $\lambda_{\mathfrak{H}_{L\delta}} = \max_{l \in \{1,\dots,L\}} \lambda_\delta^{(l)}$, there exists $\mu_{\mathfrak{H}_{L\delta}} > 0$ such that

$$\begin{aligned} \|x_{a,k} - x_{b,k}\|_\infty &\leq \mu_{\mathfrak{H}_{L\delta}} \lambda_{\mathfrak{H}_{L\delta}}^k \|x_{a,0} - x_{b,0}\|_\infty \\ &\qquad + \sum_{\tau=0}^{k-1} \mathfrak{H}_{L\delta}^{k-\tau-1} \mathfrak{G}_{L\delta} \|u_{a,\tau} - u_{b,\tau}\|_\infty. \\ &\leq \mu_{\mathfrak{H}_{L\delta}} \lambda_{\mathfrak{H}_{L\delta}}^k \|x_{a,0} - x_{b,0}\|_\infty \\ &\qquad + \|(I_{L,L} - \mathfrak{H}_{L\delta})^{-1} \mathfrak{G}_{L\delta}\|_\infty \|u_{a,0:k} - u_{b,0:k}\|_{\infty,\infty}. \end{aligned} \tag{A.167}$$

Therefore, in light of Definition 2.6, the deep GRU is ℓ_∞-δISS with functions

$$\beta(\|x_{a,0} - x_{b,0}\|_\infty, k) = \mu_{\mathfrak{H}_{L\delta}} \lambda_{\mathfrak{H}_{L\delta}}^k \|x_{a,0} - x_{b,0}\|_\infty, \tag{A.168a}$$

$$\gamma(\|u_{a,0:k} - u_{b,0:k}\|_\infty) = \|(I_{L,L} - \mathfrak{H}_{L\delta})^{-1} \mathfrak{G}_{L\delta}\|_\infty \|u_{a,0:k} - u_{b,0:k}\|_{\infty,\infty}. \tag{A.168b}$$

□

A.3 Proofs of Chapter 6

A.3.1 Proof of Theorem 6.1

The goal of the proof is to show that Definition 6.1 applies. To this end, we consider the shallow GRU system to be initialized in the unknown initial state $x_0 \in \mathcal{X}$ and fed by the input sequence $u_{0:k}$. We denote the resulting state of the system by $x_k = x_k(x_0, u_{0:k}; \Sigma(\Phi^\star))$, while $y_k = y_k(x_0, u_{0:k}; \Sigma(\Phi^\star))$ indicates its measured output. As discussed in Notation Addendum 6.1, $\hat{x}_k = \hat{x}_k(\hat{x}_0, u_{0:k}, y_{0:k}; \mathcal{O}(\Phi_o))$ denotes the state of observer (6.4) at time k when it is initialized in $\hat{x}_0 \in \mathcal{X}$, and it is fed with the known input sequence $u_{0:k}$ and the measured output $y_{0:k}$.

Let us therefore consider the j-th component of the state observation error at the generic time instant $k+1$, obtained by subtracting (6.4a) from (3.26a). By summing and subtracting the terms $[z_k]_j[\hat{x}_k]_j$ and $(1-[z_k]_j)[\hat{r}_k]_j$ we get

$$\begin{aligned}
[x_{k+1}]_j - [\hat{x}_{k+1}]_j &= [z_k]_j[x_k]_j + (1-[z_k]_j)[r_k]_j - [\hat{z}_k]_j[\hat{x}_k]_j - (1-[\hat{z}_k]_j)[\hat{r}_k]_j \\
&= [z_k]_j\big([x_k]_j - [\hat{x}_k]_j\big) + \big([z_k]_j - [\hat{z}_k]_j\big)[\hat{x}_k]_j \\
&\quad + (1-[z_k]_j)\big([r_k]_j - [\hat{r}_k]_j\big) + \big([z_k]_j - [\hat{z}_k]_j\big)[\hat{r}_k]_j
\end{aligned} \tag{A.169}$$

Along the lines of the Proof of Theorem 3.8, we take the absolute value of both sides of (A.169). Recalling that $[z_k]_j \in (0,1)$ and $[\hat{z}_k]_j \in (0,1)$, it follows that

$$\begin{aligned}
\big|[x_{k+1}]_j - [\hat{x}_{k+1}]_j\big| \leq{}& [z_k]_j\big|[x_k]_j - [\hat{x}_k]_j\big| + \big|[z_k]_j - [\hat{z}_k]_j\big|\,\big|[\hat{x}_k]_j\big| \\
&+ (1-[z_k]_j)\big|[r_k]_j - [\hat{r}_k]_j\big| + \big|[z_k]_j - [\hat{z}_k]_j\big|\,\big|[\hat{r}_k]_j\big|
\end{aligned} \tag{A.170}$$

Recalling that $\mathcal{X}$ is an invariant set for $\hat{x}$, and hence

$$|[\hat{x}_k]_j| \leq \|\hat{x}_k\|_\infty \leq \check{x}. \tag{A.171a}$$

Similarly, the term $[\hat{r}_k]_j$ can be bounded as in (A.122), i.e.

$$|[\hat{r}_k]_j| \leq \|\hat{r}_k\|_\infty \leq \check{\phi}_r. \tag{A.171b}$$

Exploiting the $\frac{1}{4}$-Lipschitzianity of σ and the linearity of output transformation, see (3.26a) and (6.4a), the following bound holds

$$\begin{aligned}
\big|[z_k]_j - [\hat{z}_k]_j\big| &\leq \|z_k - \hat{z}_k\|_\infty \\
&\leq \frac{1}{4}\|U_z(x_k - \hat{x}_k) - L_z(y_k - \hat{y}_k)\|_\infty \\
&\leq \frac{1}{4}\|U_z - L_z U_o\|_\infty \|x_k - \hat{x}_k\|_\infty.
\end{aligned} \tag{A.171c}$$

Analogously,

$$\begin{aligned}\big|[f_k]_j - [\hat{f}_k]_j\big| &\leq \|f_k - \hat{f}_k\|_\infty \\ &\leq \frac{1}{4}\|U_f(x_k - \hat{x}_k) - L_f(y_k - \hat{y}_k)\|_\infty \\ &\leq \frac{1}{4}\|U_f - L_f U_o\|_\infty \|x_k - \hat{x}_k\|_\infty.\end{aligned} \tag{A.171d}$$

Moreover, since ϕ is 1-Lipschitz, the following chain of inequalities holds true

$$\begin{aligned}\big|[r_k]_j - [\hat{r}_k]_j\big| &\leq \|r_k - \hat{r}_k\|_\infty \\ &\leq \|U_r\|_\infty \big\|\hat{f}_k \circ \hat{x}_k - f_k \circ x_k\big\|_\infty \\ &\leq \|U_r\|_\infty \big\|(f_k - \hat{f}_k) \circ \hat{x}_k + f_k \circ (x_k - \hat{x}_k)\big\|_\infty \\ &\leq \|U_r\|_\infty \Big[\check{x}\|f_k - \hat{f}_k\|_\infty + \check{\sigma}_f \|x_k - \hat{x}_k\|_\infty\Big] \\ &\leq \|U_r\|_\infty \Big(\frac{1}{4}\check{x}\|U_f - L_f U_o\|_\infty + \check{\sigma}_f\Big)\|x_k - \hat{x}_k\|_\infty\end{aligned} \tag{A.171e}$$

Therefore, in light of the bounds (A.171), the inequality (A.170) reads as

$$\big|[x_{k+1}]_j - [\hat{x}_{k+1}]_j\big| \leq \kappa_o \|x_k - \hat{x}_k\|_\infty, \tag{A.172}$$

where

$$\begin{aligned}\kappa_o = [z_k]_j + (1 - [z_k]_j)\|U_r\|_\infty \Big(\frac{1}{4}\check{x}\|U_f - L_f U_o\|_\infty + \check{\sigma}_f\Big) \\ + \frac{1}{4}(\check{x} + \check{\phi}_r)\|U_z - L_z U_o\|_\infty\end{aligned} \tag{A.173}$$

Since $[z_k]_j \in [1 - \check{\sigma}_z, \check{\sigma}_z]$, if condition (6.7) is fulfilled, it follows that (A.172) implies

$$\|x_{k+1} - \hat{x}_{k+1}\|_\infty \leq \lambda_o \|x_k - \hat{x}_k\|_\infty \tag{A.174}$$

Since $\lambda_o \in (0, 1)$, by iterating (A.174) over time we get

$$\|x_k - \hat{x}_k\|_\infty \leq \lambda_o^k \|x_0 - \hat{x}_0\|_\infty. \tag{A.175}$$

Therefore, the observer is nominally exponentially (ℓ_2) convergent with function

$$\beta_o(\|x_0 - \hat{x}_0\|_2, k) = \sqrt{n_x}\lambda_o^k \|x_0 - \hat{x}_0\|_2. \tag{A.176}$$

□

A.3.2 Proof of Proposition 6.1

First, let us point out that, as evident from equations (A.173)-(A.176), λ_o represents a bound on the observer's worst-case convergence rate. Therefore, the "optimal" gains of the observer are those that entail the smallest possible λ_o.

We therefore setup a min-max optimization problem, where the observer gains are selected as those that minimize the worst-case observer convergence rate

$$\lambda_o = \min_{L_z, L_f} \left\{ \max_{z \in [1-\breve{\sigma}_z, \breve{\sigma}_z]} \kappa_o(z, L_z, L_f) \right\}. \tag{A.177}$$

Notice that the δISS of the observed GRU model implies that the optimal solution of (A.177) satisfies Theorem 6.1, i.e. $\lambda_o \in (0, 1)$. Indeed, by taking the suboptimal gains $L_z = 0_{n_c,n_y}$ and $L_f = 0_{n_c,n_y}$, it can be easily notice that

$$\kappa_o(z, L_z, L_f) = \kappa_x \leq \lambda_\delta < 1 \tag{A.178}$$

where κ_x is defined in (A.140b) and $\kappa_x < \lambda_\delta$ is entailed by the δISS property, see (A.141)-(A.144). By definition, the optimal solution of (A.177) is characterized by $\lambda_o \leq \lambda_\delta < 1$, meaning that the optimal gains $L_z^\star$ and $L_f^\star$ satisfy Theorem 6.1.

In order to ease the solution of (A.177) we can notice that

$$\frac{\partial \kappa_o}{\partial z} = 1 - \left(\frac{1}{4} \breve{x} \| U_f - L_f U_o \|_\infty + \breve{\sigma}_f \right)$$

does not depend upon z, and hence

$$\max_{z \in [1-\breve{\sigma}_z, \breve{\sigma}_z]} \kappa_o(z, L_z, L_f) = \max \big(\kappa_o(\breve{\sigma}_z, L_z, L_f), \kappa_o(1-\breve{\sigma}_z, L_z, L_f) \big). \tag{A.179}$$

The optimization problem (A.177) can be therefore recast in the form (6.9), which – in light of the convexity of the $\max$ operator and of the ℓ_∞ norm – is convex. □

A.3.3 Proof of Theorem 6.2

The goal is to prove that the observed state $\hat{x}_k$ admits a Lyapunov function centered at the target equilibrium $\bar{x}$. To this end, we introduce the following auxiliary Lemma.

Lemma A.2. *If* (5.1) *is exponentially ℓ_∞-δISS with functions β, defined as in* (6.1), *and γ, then it is also exponentially ℓ_2-δISS with functions*

$$\begin{aligned} \beta_2(\|x_{a,0} - x_{b,0}\|_2, k) &= \sqrt{n_x} \mu_\delta \lambda_\delta^k \|x_{a,0} - x_{b,0}\|_2, \\ \gamma_2(\|u_{a,0:k} - u_{b,0:k}\|_{2,\infty}) &= \sqrt{n_u} \gamma(\|u_{a,0:k} - u_{b,0:k}\|_{\infty,\infty}). \end{aligned} \tag{A.180}$$

Proof. Recalling the definition of the ℓ_∞-δISS functions β and γ, see (A.146), and noticing that

$$\begin{aligned}\|x_k\|_2 \leq \sqrt{n_x}\|x_k\|_\infty \leq \sqrt{n_x}\|x_k\|_2,\\ \|u_k\|_2 \leq \sqrt{n_u}\|u_k\|_\infty \leq \sqrt{n_u}\|u_k\|_2,\end{aligned}$$

Lemma 2.1 can be invoked to guarantee that the shallow GRU is ℓ_2-δISS with functions β_2 and γ_2 defined as in (A.180). Hence, it satisfies[1]

$$\|x_{a,k} - x_{b,k}\|_2 \leq \beta_2(\|x_{a,0} - x_{b,0}\|_2, k) + \gamma_2(\|u_{a,0:k} - u_{b,0:k}\|_{2,\infty}). \tag{A.181}$$

□

Consider the optimal solution of (6.12) at time k. Let us denote the optimal control sequence as

$$u^\star_{k:k+N_c-1|k} = \{u^\star_{k|k}, ..., u^\star_{k+N_c-1|k}\},$$

and the corresponding state trajectories as

$$x^\star_{k:k+N|k} = \{x^\star_{k|k}, ..., x^\star_{k+N|k}\}.$$

Then, the optimal cost function $J^\star_k = J_k(u^\star_{k:k+N_c-1|k}, x^\star_{k:k+N|k})$ reads as

$$J^\star_k = \sum_{\tau=0}^{N_c-1}\Big(\|x^\star_{k+\tau|k} - \bar{x}\|^2_Q + \|u^\star_{k+\tau|k} - \bar{u}\|^2_R\Big) + \sum_{\tau=N_c}^{N}\|x^\star_{k+\tau|k} - \bar{x}\|^2_S \tag{A.182}$$

where $x^\star_{k|k} = \hat{x}_k$ due to (6.12b).

We now show that $J^\star_k$ is a Lyapunov function for the closed-loop system. To this end, let us point out that

$$J^\star_k \geq \|\hat{x}_k - \bar{x}\|^2_Q \geq \underline{\varsigma}_Q\|\hat{x}_k - \bar{x}\|^2_2. \tag{A.183}$$

We then consider a suboptimal – yet feasible – control sequence constantly equal to $\bar{u}$, i.e.

$$\tilde{u}_{k:k+N_c-1|k} = \{\bar{u}, ..., \bar{u}\},$$

and we denote by

$$\tilde{x}_{k:k+N|k} = \{\tilde{x}_{k|k}, ..., \tilde{x}_{k+N|k}\},$$

[1] We recall that $x_{a,k}$ and $x_{b,k}$ are a compact notation for $x_k(x_{a,0}, u_{a,0:k}; \Sigma(\Phi^\star))$ and $x_k(x_{b,0}, u_{b,0:k}, \Sigma(\Phi^\star))$, respectively.

the corresponding state trajectory. The suboptimality of $\tilde{u}_{k:k+N_c-1|k}$ and $\tilde{x}_{k:k+N|k}$ entails that

$$J_k^\star \leq \sum_{\tau=0}^{N_c-1} \|\tilde{x}_{k+\tau|k} - \bar{x}\|_Q^2 + \sum_{\tau=N_c}^{N_p} \|\tilde{x}_{k+\tau|k} - \bar{x}\|_S^2 \tag{A.184}$$

Since $\bar{\varsigma}_Q \leq \underline{\varsigma}_S \leq \bar{\varsigma}_S$, recalling that the GRU is ℓ_2-δISS by Lemma A.2, it holds that

$$J_k^\star \leq \bar{\varsigma}_S \sum_{\tau=0}^{N} \|\tilde{x}_{k+\tau|k} - \bar{x}\|_2^2 \leq \bar{\varsigma}_S \frac{n_x \mu_\delta^2}{1-\lambda_\delta^2} \|\hat{x}_k - \bar{x}\|_2^2, \tag{A.185}$$

where the geometric series bound has been applied. Combining (A.183) and (A.185) we get that the Lyapunov function candidate is bounded as

$$\underline{\varsigma}_Q \|\hat{x}_k - \bar{x}\|_2^2 \leq J_k^\star \leq \bar{\varsigma}_S \frac{n_x \mu_\delta^2}{1-\lambda_\delta^2} \|\hat{x}_k - \bar{x}_k\|_2^2. \tag{A.186}$$

At time $k+1$ the state observation $\hat{x}_{k+1}$ is available, and the optimization problem (6.12) is solved, yielding the optimal control sequence

$$u^\star_{k+1:k+N_c|k+1} = \{u^\star_{k+1|k+1}, \dots, u^\star_{k+N_c|k+1}\}$$

and the optimal state trajectory

$$x^\star_{k+1:k+N+1|k+1} = \{x^\star_{k+1|k+1}, \dots, x^\star_{k+N+1|k+1}\}.$$

The corresponding optimal cost function is denoted by

$$J^\star_{k+1} = J_{k+1}(x^\star_{k+1:k+N+1|k+1}, u^\star_{k+1:k+N_c|k+1}).$$

Let us notice that the optimal sequence computed at the previous iteration can be adopted as a suboptimal control sequence by letting

$$u_{k+1:k+N|k+1} = \{u^\star_{k+1|k}, \dots, u^\star_{k+N_c-1|k}, \bar{u}\}. \tag{A.187a}$$

The corresponding suboptimal state trajectory hence reads as

$$x_{k+1:k+N+1|k} = \{x_{k+1|k+1}, \dots, x_{k+N|k+1}, x_{k+N+1|k+1}\}, \tag{A.187b}$$

where it is worth stressing that $x_{k+1|k+1} = \hat{x}_{k+1} \neq x^\star_{k+1|k}$. Owing to the suboptimality of (A.187), $J^\star_{k+1}$ can be bounded as

$$\begin{aligned} J^\star_{k+1} \leq & \sum_{\tau=1}^{N_c-1} \big(\|x_{k+\tau|k+1} - \bar{x}\|_Q^2 + \|u^\star_{k+\tau|k} - \bar{u}\|_R^2\big) \\ & + \|x_{k+N_c|k+1} - \bar{x}\|_Q^2 + \sum_{\tau=N_c+1}^{N+1} \|x_{k+\tau|k+1} - \bar{x}\|_S^2 \end{aligned} \tag{A.188}$$

Then, letting $x^\star_{k+N+1|k} = f(x^\star_{k+N|k}, \bar{u})$, by defining $\forall \tau \in \{1, ..., N+1\}$

$$e_{k+\tau|k+1} = x_{k+\tau|k+1} - x^\star_{k+\tau|k}, \tag{A.189}$$

the bound (A.188) can be rewritten as

$$\begin{aligned} J^\star_{k+1} \leq & \sum_{\tau=1}^{N_c} \left\| \left(x^\star_{k+\tau|k} - \bar{x}\right) + e_{k+\tau|k+1} \right\|_Q^2 + \sum_{\tau=1}^{N_c-1} \left\| u^\star_{k+\tau|k} - \bar{u} \right\|_R^2 \\ & + \left\| \left(x^\star_{k+N_c+1|k} - \bar{x}\right) + e_{k+N_c|k+1} \right\|_Q^2 \\ & + \sum_{\tau=N_c+1}^{N+1} \left\| \left(x^\star_{k+\tau|k} - \bar{x}\right) + e_{k+\tau|k+1} \right\|_S^2 \end{aligned} \tag{A.190}$$

In light of (A.182) and (A.190), it follows that

$$J^\star_{k+1} - J^\star_k \leq -\|x^\star_{k|k} - \bar{x}\|_Q^2 - \|u^\star_{k|k} - \bar{u}\|_R^2 + \Delta J_a + \Delta J_b \tag{A.191a}$$

where

$$\begin{aligned} \Delta J_a = & \sum_{\tau=1}^{N_c-1} \left(\left\| \left(x^\star_{k+\tau|k} - \bar{x}\right) + e_{k+\tau|k+1} \right\|_Q^2 - \|x^\star_{k+t|k} - \bar{x}\|_Q^2 \right) \\ & + \sum_{\tau=N_c+1}^{N} \left(\left\| \left(x^\star_{k+\tau|k} - \bar{x}\right) + e_{k+\tau|k+1} \right\|_S^2 - \|x^\star_{k+t|k} - \bar{x}\|_S^2 \right) \end{aligned} \tag{A.191b}$$

and

$$\begin{aligned} \Delta J_b = & \left\| \left(x^\star_{k+N_c|k} - \bar{x}\right) + e_{k+N_c|k+1} \right\|_Q^2 - \|x^\star_{k+N_c|k} - \bar{x}\|_S^2 \\ & + \left\| \left(x^\star_{k+N+1|k} - \bar{x}\right) + e_{k+N+1|k+1} \right\|_S^2. \end{aligned} \tag{A.191c}$$

Let us derive a bound for the term ΔJ_a. By noticing that $\|v + w\|_Q^2 =$

$\|v\|_Q^2 + \|w\|_Q^2 + 2v'Qw$, it holds that

$$
\begin{aligned}
\Delta J_a \le & \sum_{\tau=1}^{N_c-1} \Big(\big\|x^\star_{k+\tau|k} - \bar{x}\big\|_Q^2 + \|e_{k+\tau|k+1}\|_Q^2 \\
& \qquad + 2(x^\star_{k+\tau|k} - \bar{x})'Qe_{k+\tau|k+1} - \|x^\star_{k+t|k} - \bar{x}\|_Q^2 \Big) \\
& + \sum_{\tau=N_c+1}^{N} \Big(\big\|x^\star_{k+\tau|k} - \bar{x}\big\|_S^2 + \|e_{k+\tau|k+1}\|_S^2 \\
& \qquad + 2(x^\star_{k+\tau|k} - \bar{x})'Se_{k+\tau|k+1} - \|x^\star_{k+t|k} - \bar{x}\|_S^2 \Big) \\
\le & \sum_{t=1}^{N_c-1} \Big(\|e_{k+\tau|k+1}\|_Q^2 + 2(x^\star_{k+\tau|k} - \bar{x})'Qe_{k+\tau|k+1} \Big) \\
& + \sum_{t=N_c+1}^{N} \Big(\|e_{k+\tau|k+1}\|_S^2 + 2(x^\star_{k+\tau|k} - \bar{x})'Se_{k+\tau|k+1} \Big).
\end{aligned}
\tag{A.192}
$$

In view of the boundedness of $x^\star_{k+t|k}$ in the invariant set $\mathcal{X}$, there exist finite $\mu_{a1} > 0$ and $\mu_{a2} > 0$ such that (A.192) can be upper bounded as

$$
\begin{aligned}
\Delta J_a \le \mu_{a1}\bar{\varsigma}_Q & \sum_{t=1}^{N_c-1} \big(\|e_{k+\tau|k+1}\|_2^2 + \|e_{k+\tau|k+1}\|_2\big) \\
& + \mu_{a2}\bar{\varsigma}_S \sum_{t=N_c+1}^{N} \big(\|e_{k+\tau|k+1}\|_2^2 + \|e_{k+\tau|k+1}\|_2\big).
\end{aligned}
\tag{A.193}
$$

Concerning the term ΔJ_b, it holds that

$$
\begin{aligned}
\Delta J_b \le & \|x^\star_{k+N_c|k} - \bar{x}\|_Q^2 - \|x^\star_{k+N_c|k} - \bar{x}\|_S^2 + \|x^\star_{k+N+1|k} - \bar{x}\|_S^2 \\
& + \|e_{k+N_c|k+1}\|_Q^2 + \|e_{k+N+1|k+1}\|_S^2 \\
& + 2(x^\star_{k+N_c|k} - \bar{x})'Qe_{k+N_c|k+1} + 2(x^\star_{k+N+1|k} - \bar{x})'Se_{k+N+1|k+1}
\end{aligned}
\tag{A.194}
$$

We now show that if (6.13) holds, then

$$
\|x^\star_{k+N_c|k} - \bar{x}\|_Q^2 - \|x^\star_{k+N_c|k} - \bar{x}\|_S^2 + \|x^\star_{k+N+1|k} - \bar{x}\|_S^2 < 0, \tag{A.195}
$$

or, equivalently,

$$
\|x^\star_{k+N+1|k} - \bar{x}\|_S^2 < \|x^\star_{k+N_c|k} - \bar{x}\|_{S-Q}^2, \tag{A.196}
$$

where $S - Q \succ 0$ due to (6.13a). Let us point out that if

$$
\bar{\varsigma}_S\|x^\star_{k+N+1|k} - \bar{x}\|_2^2 < \big(\underline{\varsigma}_S - \bar{\varsigma}_Q\big)\|x^\star_{k+N_c|k} - \bar{x}\|_2^2, \tag{A.197}
$$

then (A.196) surely holds. Since for $\tau \geq N_c$ the constant input $\bar{u}$ is applied, see (6.12d), Lemma A.2 can now be invoked to show that

$$\|x^\star_{k+N+1|k} - \bar{x}\|_2 \leq \sqrt{n_x}\mu_\delta \lambda_\delta^{N-N_c+1} \|x^\star_{k+N_c|k} - \bar{x}\|_2,$$

which implies that the following bound holds

$$\bar{\varsigma}_S \|x^\star_{k+N+1|k} - \bar{x}\|_2^2 \leq n_x \mu_\delta^2 \bar{\varsigma}_S \lambda_\delta^{2(N-N_c+1)} \|x^\star_{k+N_c|k} - \bar{x}\|_2^2. \tag{A.198}$$

Under (6.13b), one can therefore guarantee that

$$n_x \mu_\delta^2 \bar{\varsigma}_S \lambda_\delta^{2(N-N_c+1)} \|x^\star_{k+N_c|k} - \bar{x}\|_2^2 \leq (\underline{\varsigma}_S - \bar{\varsigma}_Q) \|x^\star_{k+N_c|k} - \bar{x}\|_2^2, \tag{A.199}$$

which, by means of a chain of inequalities, entails that (A.195) holds. Therefore, owing to the boundedness of $x_{k+N_c|k}$ in $\mathcal{X}$, there exist finite $\mu_{b1} > 0$ and $\mu_{b2} > 0$ such that (A.194) can be upper bounded by

$$\begin{aligned} \Delta J_b \leq \mu_{b1} \bar{\varsigma}_Q \big(\|e_{k+N_c|k+1}\|_2^2 + \|e_{k+N_c|k+1}\|_2 \big) \\ + \mu_{b2} \bar{\varsigma}_S \big(\|e_{k+N+1|k+1}\|_2^2 + \|e_{k+N+1|k+1}\|_2 \big). \end{aligned} \tag{A.200}$$

By combining (A.191), (A.193), and (A.200), and recalling (6.13a), it holds that there exists a finite $\mu_e > 0$ such that

$$J^\star_{k+1} - J^\star_k \leq - \|x^\star_{k|k} - \bar{x}\|_Q^2 + \underbrace{\mu_e \sum_{\tau=1}^{N+1} \left(\|e_{k+N_c|k+1}\|_2^2 + \|e_{k+N_c|k+1}\|_2 \right)}_{\varrho_{e,k}}. \tag{A.201}$$

To conclude the proof, we show that the term $\varrho_{e,k}$ exponentially converges to zero with k. To this end, let us point out that, by definition,

$$e_{k+1|k+1} = x_{k+1|k+1} - x^\star_{k+1|k} = f_o(\hat{x}_k, u^\star_{k|k}, y_k) - f(\hat{x}, u^\star_{k|k}), \tag{A.202}$$

where $f_o(\hat{x}_k, u_k, y_k)$ is the state update function of the observer $\mathcal{O}$, see (6.5). Since by Theorem 6.1 the observer is nominally converging, from (6.8) it follows that

$$\|e_{k+1|k+1}\|_2^2 \leq \mu_o^2 \lambda_o^2 \|\hat{x}_k - x_k\|_2^2. \tag{A.203}$$

Let now, for the sake of compactness,

$$u^\star_{k+1:k+N|k} = \{u^\star_{k+1|k}, ..., u^\star_{k+N_c|k}, \bar{u}, ..., \bar{u}\}.$$

For any $t \in \{2, ..., N+1\}$ by definition we have that

$$e_{k+t|k+1} = x_{k+t|k+1} - x^\star_{k+t|k}, \tag{A.204a}$$

where

$$x_{k+t|k+1} = x_{k+t}(x_{k+1|k+1}, u^{\star}_{k+1:k+t|k}; \Sigma(\Phi^{\star})), \tag{A.204b}$$

$$x^{\star}_{k+t|k} = x_{k+t}(x^{\star}_{k+1|k}, u^{\star}_{k+1:k+t|k}; \Sigma(\Phi^{\star})). \tag{A.204c}$$

Invoking Lemma A.2, $\|e_{k+t|k+1}\|_2$ can be bounded as

$$\begin{aligned} \|e_{k+t|k+1}\|_2^2 &\leq n_x \mu_\delta^2 \lambda_\delta^{2(t-1)} \|x_{k+1|k+1} - x^{\star}_{k+1|k}\|_2^2 \\ &\leq n_x \mu_\delta^2 \mu_o^2 \, \lambda_o^2 \lambda_\delta^{2(t-1)} \|\hat{x}_k - x_k\|_2^2. \end{aligned} \tag{A.205}$$

Therefore, owing to (A.203) and (A.205), and since the observer is exponentially converging, there exists $\mu_\varrho > 0$ such that

$$\varrho_{e,k} \leq \mu_\varrho \|\hat{x}_k - x_k\|_2^2 \leq \mu_\varrho \lambda_o^{2k} \|\hat{x}_0 - x_0\|_2^2. \tag{A.206}$$

That is, the perturbation term $\varrho_{e,k}$ of (A.201) exponentially converges to zero. The nominal closed-loop asymptotic stability can be therefore proven following [182]. □

A.3.4 Proof of Theorem 6.3

The goal of the proof is to show that the observer (6.22) satisfies Definition 6.2 with respect to the set $\mathcal{Z}$. In the following, consistently with the notation adopted so far, we denote by $\chi_k = \chi_k(\chi_0, v_{0:k}; \Sigma_a(\Phi^{\star}))$ and by $\hat{\chi}_k = \hat{\chi}_k(\hat{\chi}_0, v_{0:k}, \zeta_{0:k}, \bar{y}; \mathcal{O}_d(\Phi_d))$, where the state definition (6.17) is reminded.

Consider now the j-th component of the observation error of the state x at the generic time instant $k+1$, obtained by subtracting (6.22a) from (3.26a). By summing and subtracting the terms $[z_k]_j[\hat{x}_k]_j$ and $(1-[z_k]_j)[\hat{r}_k]_j$ we get

$$\begin{aligned} [x_{k+1}]_j - [\hat{x}_{k+1}]_j &= [z_k]_j[x_k]_j + (1-[z_k]_j)[r_k]_j - [\hat{z}_k]_j[\hat{x}_k]_j - (1-[\hat{z}_k]_j)[\hat{r}_k]_j \\ &= [z_k]_j\big([x_k]_j - [\hat{x}_k]_j\big) + \big([z_k]_j - [\hat{z}_k]_j\big)[\hat{x}_k]_j \\ &\quad + (1-[z_k]_j)\big([r_k]_j - [\hat{r}_k]_j\big) + \big([z_k]_j - [\hat{z}_k]_j\big)[\hat{r}_k]_j \end{aligned} \tag{A.207}$$

Along the lines of the Proof of Theorem 6.1, we take the absolute value of both sides of (A.207), which leads to

$$\begin{aligned} \big|[x_{k+1}]_j - [\hat{x}_{k+1}]_j\big| \leq{}& [z_k]_j\big|[x_k]_j - [\hat{x}_k]_j\big| + \big|[z_k]_j - [\hat{z}_k]_j\big|\,\big|[\hat{x}_k]_j\big| \\ &+ (1-[z_k]_j)\big|[r_k]_j - [\hat{r}_k]_j\big| + \big|[z_k]_j - [\hat{z}_k]_j\big|\,\big|[\hat{r}_k]_j\big| \end{aligned} \tag{A.208}$$

Recalling the bounds (3.30), since $(\chi_k, \upsilon_k) \in \mathcal{Z}$ it holds that

$$\begin{aligned} |[\hat{x}_k]_j| &\leq \|\hat{x}_k\|_\infty \leq \check{x}, \\ |[\hat{r}_k]_j| &\leq \|\hat{r}_k\|_\infty \leq \check{\phi}_r. \end{aligned} \tag{A.209a}$$

Moreover, since σ is $\frac{1}{4}$-Lipschitz, it holds that

$$\begin{aligned} \big|[z_k]_j - [\hat{z}_k]_j\big| &\leq \|z_k - \hat{z}_k\|_\infty \\ &\leq \frac{1}{4}\|W_z(\xi_k - \hat{\xi}_k) + U_z(x_k - \hat{x}_k) \\ &\qquad - L_{zy}(y_k - \hat{y}_k) - L_{z\xi}(\xi_k - \hat{\xi}_k)\|_\infty \\ &\leq \frac{1}{4}\Big(\|U_z - L_{zy}U_o\|_\infty \|x_k - \hat{x}_k\|_\infty \\ &\qquad + \|W_z - L_{z\xi}\|_\infty \|\xi_k - \hat{\xi}_k\|_\infty\Big) \end{aligned} \tag{A.209b}$$

and

$$\begin{aligned} \big|[f_k]_j - [\hat{f}_k]_j\big| &\leq \|f_k - \hat{f}_k\|_\infty \\ &\leq \frac{1}{4}\|W_f(\xi_k - \hat{\xi}_k) + U_f(x_k - \hat{x}_k) \\ &\qquad - L_{fy}(y_k - \hat{y}_k) - L_{f\xi}(\xi_k - \hat{\xi}_k)\|_\infty \\ &\leq \frac{1}{4}\Big(\|U_f - L_{fy}U_o\|_\infty \|x_k - \hat{x}_k\|_\infty \\ &\qquad + \|W_f - L_{f\xi}\|_\infty \|\xi_k - \hat{\xi}_k\|_\infty\Big) \end{aligned} \tag{A.209c}$$

Moreover, since ϕ is 1-Lipschitz, the following chain of inequalities holds true

$$\begin{aligned}\big|[r_k]_j - [\hat{r}_k]_j\big| &\leq \|r_k - \hat{r}_k\|_\infty \\ &\leq \|W_r\|_\infty \|\xi_k - \hat{\xi}_k\|_\infty \\ &\qquad + \|U_r\|_\infty \big\|\hat{f}_k \circ \hat{x}_k - f_k \circ x_k\big\|_\infty \\ &\leq \|W_r\|_\infty \|\xi_k - \hat{\xi}_k\|_\infty \\ &\qquad + \|U_r\|_\infty \big\|(f_k - \hat{f}_k) \circ \hat{x}_k + f_k \circ (x_k - \hat{x}_k)\big\|_\infty \\ &\leq \|W_r\|_\infty \|\xi_k - \hat{\xi}_k\|_\infty \\ &\qquad + \|U_r\|_\infty \Big[\check{x}\|f_k - \hat{f}_k\|_\infty + \check{\sigma}_f \|x_k - \hat{x}_k\|_\infty\Big] \\ &\leq \Big(\|W_r\|_\infty + \frac{1}{4}\check{x}\|U_r\|_\infty \|W_f - L_{f\xi}\|_\infty\Big)\|\xi_k - \hat{\xi}_k\|_\infty \\ &\qquad + \|U_r\|_\infty \Big(\frac{1}{4}\check{x}\|U_f - L_{fy}U_o\|_\infty + \check{\sigma}_f\Big)\|x_k - \hat{x}_k\|_\infty \end{aligned} \tag{A.209d}$$

Owing to (A.209), the inequality (A.208) reads as

$$\begin{aligned}\big|[x_{k+1}]_j - [\hat{x}_{k+1}]_j\big| \leq & \kappa_{dx}([z_k]_j, L_{zy}, L_{fy})\|x_k - \hat{x}_k\|_\infty \\ & + \kappa_{d\xi}([z_k]_j, L_{z\xi}, L_{f\xi})\|\xi_k - \hat{\xi}_k\|_\infty,\end{aligned} \tag{A.210}$$

where κ_{dx} and $\kappa_{d\xi}$ are defined as in (6.24). Then, noticing that the bounds in (6.23) satisfy

$$\begin{aligned}\check{\kappa}_{dx}(L_{zy}, L_{fy}) &= \sup_{z \in [1-\check{\sigma}_z, \sigma_z]} \kappa_{dx}(z, L_{zy}, L_{fy}) \\ \check{\kappa}_{d\xi}(L_{z\xi}, L_{f\xi}) &= \sup_{z \in [1-\check{\sigma}_z, \sigma_z]} \kappa_{d\xi}(z, L_{z\xi}, L_{f\xi})\end{aligned} \tag{A.211}$$

the inequality (A.210) entails

$$\|x_{k+1} - \hat{x}_{k+1}\|_\infty \leq \check{\kappa}_{d\xi}(L_{zy}, L_{fy})\|x_k - \hat{x}_k\|_\infty + \check{\kappa}_{d\xi}(L_{z\xi}, L_{f\xi})\|\xi_k - \hat{\xi}_k\|_\infty. \tag{A.212}$$

Moreover, by taking the infinity norm of the difference between ξ_{k+1} and $\hat{\xi}_{k+1}$ we get

$$\begin{aligned}\|\xi_{k+1} - \hat{\xi}_{k+1}\|_\infty \leq & \|U_o\|_\infty \|\mu_\xi - L_{\xi y}\|_\infty \|x_k - \hat{x}_k\|_\infty \\ & + \|I_{n_y, n_y} - L_{\xi\xi}\|_\infty \|\xi_k - \hat{\xi}_k\|_\infty\end{aligned} \tag{A.213}$$

From (A.212) and (A.213), it follows that

$$\begin{bmatrix}\|x_{k+1} - \hat{x}_{k+1}\|_\infty \\ \|\xi_{k+1} - \hat{\xi}_{k+1}\|_\infty\end{bmatrix} \leq \mathfrak{A}_d \begin{bmatrix}\|x_k - \hat{x}_k\|_\infty \\ \|\xi_k - \hat{\xi}_k\|_\infty\end{bmatrix}, \tag{A.214}$$

where $\mathfrak{A}_d$ is defined as in (6.25). Since the matrix $\mathfrak{A}_d$ is Schur stable, there exists a positive-definite matrix P that solves the Lyapunov equation

$$\mathfrak{A}_d' P \mathfrak{A}_d - \mathfrak{A}_d = -I_{2,2}. \tag{A.215}$$

Consider the candidate function

$$V_d(\chi_k, \hat{\chi}_k) = \left\| \begin{bmatrix} \|x_k - \hat{x}_k\|_\infty \\ \|\xi_k - \hat{\xi}_k\|_\infty \end{bmatrix} \right\|_P^2 . \tag{A.216}$$

By standard norm arguments, one can easily show that

$$\frac{\underline{\varsigma}_P}{n_x + n_y} \|\chi_k - \hat{\chi}_k\|_2^2 \leq V_d(\chi_k, \hat{\chi}_k) \leq \bar{\varsigma}_P (n_x + n_y) \|\chi_k - \hat{\chi}_k\|_2^2. \tag{A.217}$$

Moreover, it holds that

$$V_d(\chi_{k+1}, \hat{\chi}_{k+1}) \leq - \left\| \begin{bmatrix} \|x_k - \hat{x}_k\|_\infty \\ \|\xi_k - \hat{\xi}_k\|_\infty \end{bmatrix} \right\|_2^2 \leq - \frac{1}{n_x + n_y} \|\chi_k - \hat{\chi}_k\|_2^2. \tag{A.218}$$

Therefore, (6.22) is a weak detector of (6.16) with respect to the set $\mathcal{Z}$. □

List of Figures

List of Tables

Bibliography

[1] Z.-S. Hou and Z. Wang, “From model-based control to data-driven control: Survey, classification and perspective,” *Information Sciences*, vol. 235, pp. 3–35, 2013.

[2] J. Schoukens and L. Ljung, “Nonlinear system identification: A user-oriented road map,” *IEEE Control Systems Magazine*, vol. 39, no. 6, pp. 28–99, 2019.

[3] J. Rawlings and D. Mayne, *Model predictive control: theory and design*. Nob Hill Publishing, 2009.

[4] M. Korda and I. Mezić, “Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control,” *Automatica*, vol. 93, pp. 149–160, 2018.

[5] E. Terzi, L. Fagiano, M. Farina, and R. Scattolini, “Learning-based predictive control for linear systems: A unitary approach,” *Automatica*, vol. 108, p. 108473, 2019.

[6] L. Hewing, J. Kabzan, and M. N. Zeilinger, “Cautious model predictive control using gaussian process regression,” *IEEE Transactions on Control Systems Technology*, vol. 28, no. 6, pp. 2736–2743, 2019.

[7] A. Aswani, H. Gonzalez, S. S. Sastry, and C. Tomlin, “Provably safe and robust learning-based model predictive control,” *Automatica*, vol. 49, no. 5, pp. 1216–1226, 2013.

[8] M. C. Campi, A. Lecchini, and S. M. Savaresi, “Virtual reference feedback tuning: a direct method for the design of feedback controllers,” *Automatica*, vol. 38, no. 8, pp. 1337–1346, 2002.

[9] J. Coulson, J. Lygeros, and F. Dörfler, “Data-enabled predictive control: In the shallows of the deepc,” in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 307–312.

[10] K. Hornik, M. Stinchcombe, H. White *et al.*, “Multilayer feedforward networks are universal approximators.” *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[11] A. M. Schäfer and H. G. Zimmermann, “Recurrent neural networks are universal approximators,” in *International Conference on Artificial Neural Networks*. Springer, 2006, pp. 632–640.

[12] M. J. Willis, G. A. Montague, C. Di Massimo, M. T. Tham, and A. J. Morris, “Artificial neural networks in process estimation and control,” *Automatica*, vol. 28, no. 6, pp. 1181–1187, 1992.

[13] F. M. Bianchi, E. Maiorino, M. C. Kampffmeyer, A. Rizzi, and R. Jenssen, *Recurrent neural networks for short-term load forecasting: an overview and comparative analysis*. Springer, 2017.

[14] Y. Bengio, I. Goodfellow, and A. Courville, *Deep learning*. MIT press Massachusetts, USA, 2017, vol. 1.

[15] K. Hunt and D. Sbarbaro, "Neural networks for nonlinear internal model control," in *IEE Proceedings D (Control Theory and Applications)*, vol. 138, no. 5. IET, 1991, pp. 431–438.

[16] K. J. Hunt, D. Sbarbaro, R. Żbikowski, and P. J. Gawthrop, "Neural networks for control systems – a survey," *Automatica*, vol. 28, no. 6, pp. 1083–1112, 1992.

[17] A. Chiuso and G. Pillonetto, "System identification: A machine learning perspective," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, pp. 281–304, 2019.

[18] J. Xu, M. Kovatsch, D. Mattern, F. Mazza, M. Harasic, A. Paschke, and S. Lucia, "A review on ai for smart manufacturing: Deep learning challenges and solutions," *Applied Sciences*, vol. 12, no. 16, p. 8239, 2022.

[19] Z. Wu, A. Tran, D. Rincon, and P. D. Christofides, "Machine learning-based predictive control of nonlinear processes. part i: Theory," *AIChE Journal*, vol. 65, no. 11, 2019.

[20] F. Bonassi, M. Farina, J. Xie, and R. Scattolini, "On Recurrent Neural Networks for learning-based control: recent results and ideas for future developments," *Journal of Process Control*, vol. 114, pp. 92–104, 2022.

[21] A. U. Levin and K. S. Narendra, "Identification using feedforward networks," *Neural Computation*, vol. 7, no. 2, pp. 349–369, 1995.

[22] E. Terzi, F. Bonassi, M. Farina, and R. Scattolini, "Learning model predictive control with long short-term memory networks," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 18, pp. 8877–8896, 2021.

[23] Z. K. Nagy, "Model based control of a yeast fermentation bioreactor using optimally designed artificial neural networks," *Chemical engineering journal*, vol. 127, no. 1-3, pp. 95–109, 2007.

[24] N. Lanzetti, Y. Z. Lian, A. Cortinovis, L. Dominguez, M. Mercangöz, and C. Jones, "Recurrent neural network based MPC for process industries," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 1005–1010.

[25] W. C. Wong, E. Chee, J. Li, and X. Wang, "Recurrent neural network-based model predictive control for continuous pharmaceutical manufacturing," *Mathematics*, vol. 6, no. 11, p. 242, 2018.

[26] A. Karpatne, G. Atluri, J. H. Faghmous, M. Steinbach, A. Banerjee, A. Ganguly, S. Shekhar, N. Samatova, and V. Kumar, "Theory-guided data science: A new paradigm for scientific discovery from data," *IEEE Transactions on knowledge and data engineering*, vol. 29, no. 10, pp. 2318–2331, 2017.

[27] S. Pozzoli, M. Gallieri, and R. Scattolini, "Tustin neural networks: a class of recurrent nets for adaptive MPC of mechanical systems," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 5171–5176, 2020.

[28] M. Cranmer, S. Greydanus, S. Hoyer, P. Battaglia, D. Spergel, and S. Ho, "Lagrangian neural networks," in *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, 2020.

[29] M. A. Hosen, M. A. Hussain, and F. S. Mjalli, "Control of polystyrene batch reactors using neural network based model predictive control (NNMPC): An experimental investigation," *Control Engineering Practice*, vol. 19, no. 5, pp. 454–467, 2011.

[30] Z. Wu, D. Rincon, and P. D. Christofides, "Process structure-based recurrent neural network modeling for model predictive control of nonlinear processes," *Journal of Process Control*, vol. 89, pp. 74–84, 2020.

[31] G. Zhao, P. Zhang, G. Ma, and W. Xiao, "System identification of the nonlinear residual errors of an industrial robot using massive measurements," *Robotics and Computer-Integrated Manufacturing*, vol. 59, pp. 104–114, 2019.

[32] M. Forgione and D. Piga, "Model structures and fitting criteria for system identification with neural networks," in *2020 IEEE 14th International Conference on Application of Information and Communication Technologies (AICT)*. IEEE, 2020, pp. 1–6.

[33] A. Alessio and A. Bemporad, "A survey on explicit model predictive control," in *Nonlinear model predictive control*. Springer, 2009, pp. 345–369.

[34] T. Parisini and R. Zoppoli, "A receding-horizon regulator for nonlinear systems and a neural approximation," *Automatica*, vol. 31, no. 10, pp. 1443–1451, 1995.

[35] L. Cavagnari, L. Magni, and R. Scattolini, "Neural network implementation of nonlinear receding-horizon control," *Neural computing & applications*, vol. 8, no. 1, pp. 86–92, 1999.

[36] B. Karg and S. Lucia, "Approximate moving horizon estimation and robust nonlinear model predictive control via deep learning," *Computers & Chemical Engineering*, vol. 148, p. 107266, 2021.

[37] M. Hertneck, J. Köhler, S. Trimpe, and F. Allgöwer, "Learning an approximate model predictive controller with guarantees," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 543–548, 2018.

[38] B. Karg and S. Lucia, "Efficient representation and approximation of model predictive control laws via deep learning," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3866–3878, 2020.

[39] P. Kumar, J. B. Rawlings, and S. J. Wright, "Industrial, large-scale model predictive control with structured neural networks," *Computers & Chemical Engineering*, vol. 150, p. 107291, 2021.

[40] B. Karg and S. Lucia, "Stability and feasibility of neural network-based controllers via output range analysis," in *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 4947–4954.

[41] B. Karg, T. Alamo, and S. Lucia, "Probabilistic performance validation of deep learning-based robust nmpc controllers," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 18, pp. 8855–8876, 2021.

[42] A. Yeşildirek and F. L. Lewis, "Feedback linearization using neural networks," *Automatica*, vol. 31, no. 11, pp. 1659–1664, 1995.

[43] Y. Li, H. He, J. Wu, D. Katabi, and A. Torralba, "Learning compositional koopman operators for model-based control," in *International Conference on Learning Representations*, 2020.

[44] M. Tanaskovic, L. Fagiano, C. Novara, and M. Morari, "Data-driven control of nonlinear systems: An on-line direct approach," *Automatica*, vol. 75, pp. 1–10, 2017.

[45] P. Yan, D. Liu, D. Wang, and H. Ma, "Data-driven controller design for general mimo nonlinear systems via virtual reference feedback tuning and neural networks," *Neurocomputing*, vol. 171, pp. 815–825, 2016.

[46] M.-B. Radac and R.-E. Precup, "Data-driven MIMO model-free reference tracking control with nonlinear state-feedback and fractional order controllers," *Applied Soft Computing*, vol. 73, pp. 992–1003, 2018.

[47] W. D'Amico, M. Farina, and G. Panzani, "Advanced control based on recurrent neural networks learned using virtual reference feedback tuning and application to an electronic throttle body (with supplementary material)," *arXiv preprint arXiv:2103.02567*, 2021.

[48] L. Buşoniu, T. de Bruin, D. Tolić, J. Kober, and I. Palunko, "Reinforcement learning for control: Performance, stability, and deep approximators," *Annual Reviews in Control*, vol. 46, pp. 8–28, 2018.

[49] T. A. Badgwell, J. H. Lee, and K.-H. Liu, "Reinforcement learning–overview of recent progress and implications for process control," in *Computer Aided Chemical Engineering*. Elsevier, 2018, vol. 44, pp. 71–85.

[50] R. Özalp, N. K. Varol, B. Taşci, and A. Uçar, "A review of deep reinforcement learning algorithms and comparative results on inverted pendulum system," *Machine Learning Paradigms*, pp. 237–256, 2020.

[51] A. B. Martinsen, A. M. Lekkas, and S. Gros, "Combining system identification with reinforcement learning-based mpc," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 8130–8135, 2020.

[52] M. Zanon and S. Gros, "Safe reinforcement learning using robust mpc," *IEEE Transactions on Automatic Control*, vol. 66, no. 8, pp. 3638–3652, 2020.

[53] D. M. Himmelblau, "Accounts of experiences in the application of artificial neural networks in chemical engineering," *Industrial & Engineering Chemistry Research*, vol. 47, no. 16, pp. 5782–5796, 2008.

[54] J. Atuonwu, Y. Cao, G. Rangaiah, and M. Tadé, "Identification and predictive control of a multistage evaporator," *Control Engineering Practice*, vol. 18, no. 12, pp. 1418–1428, 2010.

[55] E. Terzi, T. Bonetti, D. Saccani, M. Farina, L. Fagiano, and R. Scattolini, "Learning-based predictive control of the cooling system of a large business centre," *Control Engineering Practice*, vol. 97, p. 104348, 2020.

[56] P. Zeng, H. Li, H. He, and S. Li, "Dynamic energy management of a microgrid using approximate dynamic programming and deep recurrent neural network learning," *IEEE Transactions on Smart Grid*, vol. 10, no. 4, pp. 4435–4445, 2018.

[57] F. Bonassi, M. Farina, and R. Scattolini, "Stability of discrete-time feed-forward neural networks in NARX configuration," in *19th IFAC Symposium on System Identification (SYSID 2021)*, 2021, pp. 547–552, IFAC-PapersOnLine 54.7.

[58] F. Bonassi, M. Farina, and R. Scattolini, "On the stability properties of gated recurrent units neural networks," *Systems & Control Letters*, vol. 157, p. 105049, 2021.

[59] F. Bonassi, A. La Bella, G. Panzani, M. Farina, and R. Scattolini, "Deep Long-Short Term Memory networks: Stability properties and Experimental validation," in *2023 European Control Conference (ECC)*, 2023, *(Under review)*.

[60] F. Bonassi, C. F. Oliveira da Silva, and R. Scattolini, "Nonlinear MPC for Offset-Free Tracking of systems learned by GRU Neural Networks," in *3rd IFAC Conference on Modelling, Identification and Control of Nonlinear Systems (MICNON 2021)*, 2021, pp. 54–59, IFAC-PapersOnLine 54.14.

[61] F. Bonassi, J. Xie, M. Farina, and R. Scattolini, "An Offset-Free Nonlinear MPC scheme for systems learned by Neural NARX models," in *2022 IEEE 61st Conference on Decision and Control (CDC)*, 2022, pp. 2123–2128.

[62] F. Bonassi and R. Scattolini, "Recurrent neural network-based Internal Model Control of unknown nonlinear stable systems," *European Journal of Control*, p. 100632, 2022.

[63] F. Bonassi, A. La Bella, M. Farina, and R. Scattolini, "Nonlinear MPC design for incrementally ISS systems with application to GRU networks," *Automatica*, 2023, *(In preparation)*.

[64] J. Xie, F. Bonassi, M. Farina, and R. Scattolini, "Robust offset-free nonlinear model predictive control learned by neural nonlinear autoregressive exogenous models," *International Journal of Robust and Nonlinear Control*, 2022, *(Under review, arXiv preprint 2210.06801)*.

[65] F. Bonassi, E. Terzi, M. Farina, and R. Scattolini, "LSTM neural networks: Input to state stability and probabilistic safety verification," in *Learning for Dynamics and Control*. PMLR, 2020, pp. 85–94.

[66] F. Bonassi, J. Xie, M. Farina, and R. Scattolini, "Towards lifelong learning of recurrent neural networks for control design," in *2022 European Control Conference (ECC)*, 2022, pp. 2018–2023.

[67] E. D. Sontag *et al.*, "Smooth stabilization implies coprime factorization," *IEEE transactions on automatic control*, vol. 34, no. 4, pp. 435–443, 1989.

[68] Z.-P. Jiang, A. R. Teel, and L. Praly, "Small-gain theorem for iss systems and applications," *Mathematics of Control, Signals and Systems*, vol. 7, no. 2, pp. 95–120, 1994.

[69] D. Angeli, "A lyapunov approach to incremental stability properties," *IEEE Transactions on Automatic Control*, vol. 47, no. 3, pp. 410–421, 2002.

[70] Z.-P. Jiang and Y. Wang, "Input-to-state stability for discrete-time nonlinear systems," *Automatica*, vol. 37, no. 6, pp. 857–869, 2001.

[71] M. Lazar, D. M. De La Pena, W. M. H. Heemels, and T. Alamo, "On input-to-state stability of min–max nonlinear model predictive control," *Systems & Control Letters*, vol. 57, no. 1, pp. 39–48, 2008.

[72] F. Bayer, M. Bürger, and F. Allgöwer, "Discrete-time incremental ISS: A framework for robust NMPC," in *2013 European Control Conference (ECC)*. IEEE, 2013, pp. 2068–2073.

[73] E. D. Sontag, "Input to state stability: Basic concepts and results," in *Nonlinear and optimal control theory*. Springer, 2008, pp. 163–220.

[74] M. Fazlyab, M. Morari, and G. J. Pappas, "Probabilistic verification and reachability analysis of neural networks via semidefinite programming," *arXiv preprint arXiv:1910.04249*, 2019.

[75] A. Lindholm, N. Wahlström, F. Lindsten, and T. B. Schön, *Machine Learning: A First Course for Engineers and Scientists*. Cambridge University Press, 2022.

[76] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[77] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*. PMLR, 2013, pp. 1310–1318.

[78] H. Jaeger, "Adaptive nonlinear system identification with echo state networks," *Advances in neural information processing systems*, vol. 15, 2002.

[79] H. Jaeger, "Echo state network," *Scholarpedia*, vol. 2, no. 9, p. 2330, 2007.

[80] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997.

[81] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, 2014.

[82] M. Sangiorgio, "Deep learning in multi-step forecasting of chaotic dynamics," in *Special Topics in Information Technology*. Springer, Cham, 2022, pp. 3–14.

[83] S. Viswanathan, M. Anand Kumar, and K. Soman, "A sequence-based machine comprehension modeling using lstm and gru," in *Emerging research in electronics, computer science and technology*. Springer, 2019, pp. 47–55.

[84] A. Rehmer and A. Kroll, "On using Gated Recurrent Units for Nonlinear System Identification," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 2504–2509.

[85] N. Mohajerin and S. L. Waslander, "Multistep prediction of dynamic systems with recurrent neural networks," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3370–3383, 2019.

[86] H. Zhang, Z. Wang, and D. Liu, "A comprehensive review of stability analysis of continuous-time recurrent neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 7, pp. 1229–1262, 2014.

[87] L. Jin, P. N. Nikiforuk, and M. M. Gupta, "Absolute stability conditions for discrete-time recurrent neural networks," *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 954–964, 1994.

[88] N. E. Barabanov and D. V. Prokhorov, "Stability analysis of discrete-time recurrent neural networks," *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 292–303, 2002.

[89] S. Hu and J. Wang, "Global stability of a class of discrete-time recurrent neural networks," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 49, no. 8, pp. 1104–1117, 2002.

[90] J. Miller and M. Hardt, "Stable recurrent models," in *International Conference on Learning Representations*, 2019, arXiv preprint arXiv:1805.10369.

[91] E. N. Sanchez and J. P. Perez, "Input-to-state stability (iss) analysis for dynamic neural networks," *IEEE Transactions on circuits and systems I: Fundamental Theory and Applications*, vol. 46, no. 11, pp. 1395–1398, 1999.

[92] L. Bugliari Armenio, E. Terzi, M. Farina, and R. Scattolini, "Echo State Networks: analysis, training and predictive control," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 799–804.

[93] L. Bugliari Armenio, E. Terzi, M. Farina, and R. Scattolini, "Model Predictive Control Design for Dynamical Systems Learned by Echo State Networks," *IEEE Control Systems Letters*, vol. 3, no. 4, pp. 1044–1049, 2019.

[94] D. M. Stipanović *et al.*, "Some local stability properties of an autonomous long short-term memory neural network model," in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2018, pp. 1–5.

[95] S. A. Deka, D. M. Stipanović, B. Murmann, and C. J. Tomlin, "Global asymptotic stability and stabilization of long short-term memory neural networks with constant weights and biases," *Journal of Optimization Theory and Applications*, vol. 181, no. 1, pp. 231–243, 2019.

[96] D. M. Stipanović, M. N. Kapetina, M. R. Rapaić, and B. Murmann, "Stability of gated recurrent unit neural networks: Convex combination formulation approach," *Journal of Optimization Theory and Applications*, pp. 1–16, 2020.

[97] C. Califano, S. Monaco, and D. Normand-Cyrot, "On the discrete-time normal form," *IEEE transactions on automatic control*, vol. 43, no. 11, pp. 1654–1658, 1998.

[98] H. Mhaskar, Q. Liao, and T. Poggio, "When and why are deep networks better than shallow ones?" in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, no. 1, 2017.

[99] G.-B. Zhou, J. Wu, C.-L. Zhang, and Z.-H. Zhou, "Minimal gated unit for recurrent neural networks," *International Journal of Automation and Computing*, vol. 13, no. 3, pp. 226–234, 2016.

[100] O. Nerrand, P. Roussel-Ragot, D. Urbani, L. Personnaz, and G. Dreyfus, "Training recurrent neural networks: Why and how? an illustration in dynamical process modeling," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 178–184, 1994.

[101] X. Wang and Y. Huang, "Convergence study in extended kalman filter-based training of recurrent neural networks," *IEEE Transactions on Neural Networks*, vol. 22, no. 4, pp. 588–600, 2011.

[102] T. Desell, S. Clachar, J. Higgins, and B. Wild, "Evolving deep recurrent neural networks using ant colony optimization," in *European Conference on Evolutionary Computation in Combinatorial Optimization*. Springer, 2015, pp. 86–98.

[103] S. Yang, Y. Tian, C. He, X. Zhang, K. C. Tan, and Y. Jin, "A gradient-guided evolutionary approach to training deep neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[104] A. Kag and V. Saligrama, "Training recurrent neural networks via forward propagation through time," in *International Conference on Machine Learning*. PMLR, 2021, pp. 5189–5200.

[105] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[106] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[107] M. Mehdipour Ghazi, M. Nielsen, A. Pai, M. Modat, M. J. Cardoso, S. Ourselin, and L. Sørensen, "On the initialization of long short-term memory networks," in *International Conference on Neural Information Processing*. Springer, 2019, pp. 275–286.

[108] S. Semeniuta, A. Severyn, and E. Barth, "Recurrent dropout without memory loss," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 1757–1766.

[109] O. Kuchaiev and B. Ginsburg, "Factorization tricks for lstm networks," *arXiv preprint arXiv:1703.10722*, 2017.

[110] R. C. Hall and D. E. Seborg, "Modelling and self-tuning control of a multivariable pH neutralization process part I: Modelling and multiloop control," in *1989 American Control Conference*. IEEE, 1989, pp. 1822–1827.

[111] M. A. Henson and D. E. Seborg, "Adaptive nonlinear control of a ph neutralization process," *IEEE transactions on control systems technology*, vol. 2, no. 3, pp. 169–182, 1994.

[112] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035.

[113] G. De Nicolao, L. Magni, and R. Scattolini, "Stabilizing predictive control of nonlinear arx models," *Automatica*, vol. 33, no. 9, pp. 1691–1697, 1997.

[114] L. Magni, D. M. Raimondo, and F. Allgöwer, "Nonlinear model predictive control," *Lecture Notes in Control and Information Sciences*, vol. 384, 2009.

[115] E. F. Camacho and C. Bordons, "Nonlinear model predictive control," in *Model Predictive control*. Springer, 2007, pp. 249–288.

[116] J. B. Rawlings, D. Q. Mayne, and M. Diehl, *Model predictive control: theory, computation, and design*. Nob Hill Publishing Madison, WI, 2017, vol. 2.

[117] L. Magni, G. De Nicolao, and R. Scattolini, "Output feedback and tracking of nonlinear systems with model predictive control," *Automatica*, vol. 37, no. 10, pp. 1601–1607, 2001.

[118] M. Morari and U. Maeder, "Nonlinear offset-free model predictive control," *Automatica*, vol. 48, no. 9, pp. 2059–2067, 2012.

[119] L. Magni and R. Scattolini, “Robustness and robust design of mpc for nonlinear discrete-time systems,” in *Assessment and future directions of nonlinear model predictive control*. Springer, 2007, pp. 239–254.

[120] D. Q. Mayne, E. C. Kerrigan, E. Van Wyk, and P. Falugi, “Tube-based robust nonlinear model predictive control,” *International journal of robust and nonlinear control*, vol. 21, no. 11, pp. 1341–1353, 2011.

[121] S. Lucia, T. Finkler, and S. Engell, “Multi-stage nonlinear model predictive control applied to a semi-batch polymerization reactor under uncertainty,” *Journal of process control*, vol. 23, no. 9, pp. 1306–1319, 2013.

[122] P. Li, H. Arellano-Garcia, and G. Wozny, “Chance constrained programming approach to process optimization under uncertainty,” *Computers & chemical engineering*, vol. 32, no. 1-2, pp. 25–45, 2008.

[123] S. J. Qin and T. A. Badgwell, “An overview of nonlinear model predictive control applications,” *Nonlinear model predictive control*, pp. 369–392, 2000.

[124] J. J. Song and S. Park, “Neural model predictive control for nonlinear chemical processes,” *Journal of chemical engineering of Japan*, vol. 26, no. 4, pp. 347–354, 1993.

[125] A. Draeger, S. Engell, and H. Ranke, “Model predictive control using neural networks,” *IEEE Control Systems Magazine*, vol. 15, no. 5, pp. 61–66, 1995.

[126] S. Piche, B. Sayyar-Rodsari, D. Johnson, and M. Gerules, “Nonlinear model predictive control using neural networks,” *IEEE Control Systems Magazine*, vol. 20, no. 3, pp. 53–62, 2000.

[127] Z. Wu, J. Luo, D. Rincon, and P. D. Christofides, “Machine learning-based predictive control using noisy data: evaluating performance and robustness via a large-scale process simulator,” *Chemical Engineering Research and Design*, vol. 168, pp. 275–287, 2021.

[128] K. Patan, “Neural network-based model predictive control: Fault tolerance and stability,” *IEEE Transactions on Control Systems Technology*, vol. 23, no. 3, pp. 1147–1155, 2014.

[129] G. Pannocchia, “Offset-free tracking mpc: A tutorial review and comparison of different formulations,” in *2015 European control conference (ECC)*. IEEE, 2015, pp. 527–532.

[130] P. Tatjewski and M. Ławryńczuk, “Algorithms with state estimation in linear and nonlinear model predictive control,” *Computers & Chemical Engineering*, vol. 143, p. 107065, 2020.

[131] A. Bamimore, N. Sobowale, A. Osunleke, and O. Taiwo, “Offset-free neural network-based nonlinear model predictive controller design using parameter adaptation,” *Neural Computing and Applications*, vol. 33, no. 16, pp. 10 235–10 257, 2021.

[132] S. H. Son, J. W. Kim, T. H. Oh, D. H. Jeong, and J. M. Lee, “Learning of model-plant mismatch map via neural network modeling and its application to offset-free model predictive control,” *Journal of Process Control*, vol. 115, pp. 112–122, 2022.

[133] C. G. Economou, M. Morari, and B. O. Palsson, “Internal model control: Extension to nonlinear system,” *Industrial & Engineering Chemistry Process Design and Development*, vol. 25, no. 2, pp. 403–411, 1986.

[134] M. Morari and E. Zafiriou, *Robust process control*. Morari, 1989.

[135] M. Morari, “Internal model control-theory and applications,” *IFAC Proceedings Volumes*, vol. 16, no. 21, pp. 1–18, 1983.

[136] M. A. Henson and D. E. Seborg, “An internal model control strategy for nonlinear systems,” *AIChE Journal*, vol. 37, no. 7, pp. 1065–1081, 1991.

[137] I. Rivals and L. Personnaz, “Nonlinear internal model control using neural networks: Application to processes with delay and design issues,” *IEEE transactions on neural networks*, vol. 11, no. 1, pp. 80–90, 2000.

[138] C. F. Oliveira da Silva, "Offset-free nonlinear MPC for systems learned by LSTM networks," 2021, Master thesis, Politecnico di Milano, Italy.

[139] L. Magni, G. De Nicolao, L. Magnani, and R. Scattolini, "A stabilizing model-based predictive control algorithm for nonlinear systems," *Automatica*, vol. 37, no. 9, pp. 1351–1362, 2001.

[140] A. Boccia, L. Grüne, and K. Worthmann, "Stability and feasibility of state constrained MPC without stabilizing terminal constraints," *Systems & control letters*, vol. 72, pp. 14–21, 2014.

[141] B. A. Francis and W. M. Wonham, "The internal model principle of control theory," *Automatica*, vol. 12, no. 5, pp. 457–465, 1976.

[142] R. Scattolini and N. Schiavoni, "A parameter optimization approach to the design of structurally constrained regulators for discrete-time systems," *International Journal of Control*, vol. 42, no. 1, pp. 177–192, 1985.

[143] L. Magni, G. De Nicolao, and R. Scattolini, "Model predictive control: output feedback and tracking of nonlinear systems," *IEE CONTROL ENGINEERING SERIES*, pp. 61–80, 2001.

[144] J. Köhler, M. A. Müller, and F. Allgöwer, "A nonlinear model predictive control framework using reference generic terminal ingredients," *IEEE Transactions on Automatic Control*, vol. 65, no. 8, pp. 3576–3583, 2019.

[145] K. Ohno, "A new approach to differential dynamic programming for discrete time systems," *IEEE Transactions on Automatic Control*, vol. 23, no. 1, pp. 37–47, 1978.

[146] M. Grant and S. Boyd, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control*, ser. Lecture Notes in Control and Information Sciences, V. Blondel, S. Boyd, and H. Kimura, Eds. Springer-Verlag Limited, 2008, pp. 95–110.

[147] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.2," http://cvxr.com/cvx, Mar. 2014.

[148] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[149] A. Alessandri, M. Baglietto, and G. Battistelli, "Moving-horizon state estimation for nonlinear discrete-time systems: New stability results and approximation schemes," *Automatica*, vol. 44, no. 7, pp. 1753–1765, 2008.

[150] H. K. Khalil, *Nonlinear systems; 3rd ed.* Prentice-Hall, 2002.

[151] I. Alvarado, D. Limon, D. M. De La Peña, J. M. Maestre, M. Ridao, H. Scheu, W. Marquardt, R. Negenborn, B. De Schutter, F. Valencia *et al.*, "A comparative analysis of distributed MPC techniques applied to the HD-MPC four-tank benchmark," *Journal of Process Control*, vol. 21, no. 5, pp. 800–815, 2011.

[152] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org.

[153] S. Wang, K. Pei, J. Whitehouse, J. Yang, and S. Jana, "Efficient formal safety analysis of neural networks," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[154] S. Dutta, S. Jha, S. Sankaranarayanan, and A. Tiwari, "Output range analysis for deep feedforward neural networks," in *NASA Formal Methods Symposium*. Springer, 2018, pp. 121–138.

[155] W. Ruan, X. Huang, and M. Kwiatkowska, "Reachability analysis of deep neural networks with provable guarantees," *arXiv preprint arXiv:1805.02242*, 2018.

[156] W. Xiang, H.-D. Tran, and T. T. Johnson, "Output reachable set estimation and verification for multilayer neural networks," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 11, pp. 5777–5783, 2018.

[157] K. Dvijotham, R. Stanforth, S. Gowal, T. A. Mann, and P. Kohli, “A dual approach to scalable verification of deep networks.” in *UAI*, vol. 1, no. 2, 2018, p. 3.

[158] M. Fazlyab, A. Robey, H. Hassani, M. Morari, and G. Pappas, “Efficient and accurate estimation of lipschitz constants for deep neural networks,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[159] C.-Y. Ko, Z. Lyu, L. Weng, L. Daniel, N. Wong, and D. Lin, “POPQORN: Quantifying robustness of recurrent neural networks,” in *International Conference on Machine Learning*. PMLR, 2019, pp. 3468–3477.

[160] A. Mironchenko and C. Prieur, “Input-to-state stability of infinite-dimensional systems: recent results and open questions,” *SIAM Review*, vol. 62, no. 3, pp. 529–614, 2020.

[161] M. C. Campi, S. Garatti, and M. Prandini, “The scenario approach for systems and control design,” *Annual Reviews in Control*, vol. 33, no. 2, pp. 149–157, 2009.

[162] L. Hewing and M. N. Zeilinger, “Scenario-based probabilistic reachable sets for recursively feasible stochastic model predictive control,” *IEEE Control Systems Letters*, vol. 4, no. 2, pp. 450–455, 2019.

[163] T. Alamo, R. Tempo, A. Luque, and D. R. Ramirez, “Randomized methods for design of uncertain systems: Sample complexity and sequential algorithms,” *Automatica*, vol. 52, pp. 160–172, 2015.

[164] D. Clarke and P. Gawthrop, “Self-tuning control,” in *Proceedings of the Institution of Electrical Engineers*, vol. 126, no. 6. IET, 1979, pp. 633–640.

[165] A. Cossu, A. Carta, V. Lomonaco, and D. Bacciu, “Continual learning for recurrent neural networks: An empirical evaluation,” *Neural Networks*, vol. 143, pp. 607–627, 2021.

[166] V. Losing, B. Hammer, and H. Wersing, “Incremental on-line learning: A review and comparison of state of the art algorithms,” *Neurocomputing*, vol. 275, pp. 1261–1274, 2018.

[167] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, “Continual lifelong learning with neural networks: A review,” *Neural Networks*, vol. 113, pp. 54–71, 2019.

[168] S. Sodhani, S. Chandar, and Y. Bengio, “Toward training recurrent neural networks for lifelong learning,” *Neural computation*, vol. 32, no. 1, pp. 1–35, 2020.

[169] M. Forgione, A. Muni, D. Piga, and M. Gallieri, “On the adaptation of recurrent neural networks for system identification,” *arXiv preprint arXiv:2201.08660*, 2022.

[170] J. Willard, X. Jia, S. Xu, M. Steinbach, and V. Kumar, “Integrating physics-based modeling with machine learning: A survey,” *arXiv preprint arXiv:2003.04919*, 2020.

[171] N. Thuerey, P. Holl, M. Mueller, P. Schnell, F. Trost, and K. Um, “Physics-based deep learning,” *arXiv preprint arXiv:2109.05237*, 2021.

[172] M. V. Egorchev and Y. V. Tiumentsev, “Semi-empirical neural network based approach to modelling and simulation of controlled dynamical systems,” *Procedia computer science*, vol. 123, pp. 134–139, 2018.

[173] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[174] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, “Neural ordinary differential equations,” *Advances in neural information processing systems*, vol. 31, 2018.

[175] A. Daw, R. Q. Thomas, C. C. Carey, J. S. Read, A. P. Appling, and A. Karpatne, “Physics-guided architecture (PGA) of neural networks for quantifying uncertainty in lake temperature modeling,” in *Proceedings of the 2020 siam international conference on data mining*. SIAM, 2020, pp. 532–540.

[176] N. Mertens, C. Kunde, A. Kienle, and D. Michaels, “Monotonic reformulation and bound tightening for global optimization of ideal multi-component distillation columns,” *Optimization and Engineering*, vol. 19, no. 2, pp. 479–514, 2018.

[177] M. S. Alhajeri, J. Luo, Z. Wu, F. Albalawi, and P. D. Christofides, “Process structure-based recurrent neural network modeling for predictive control: A comparative study,” *Chemical Engineering Research and Design*, 2022.

[178] P. O. Scokaert and D. Q. Mayne, “Min-max feedback model predictive control for constrained linear systems,” *IEEE Transactions on Automatic control*, vol. 43, no. 8, pp. 1136–1142, 1998.

[179] W. Langson, I. Chryssochoos, S. Raković, and D. Q. Mayne, “Robust model predictive control using tubes,” *Automatica*, vol. 40, no. 1, pp. 125–133, 2004.

[180] N. Bof, R. Carli, and L. Schenato, “Lyapunov theory for discrete time systems,” *arXiv preprint arXiv:1809.05289*, 2018.

[181] E. Jury, “A simplified stability criterion for linear discrete systems,” *Proceedings of the IRE*, vol. 50, no. 6, pp. 1493–1500, 1962.

[182] P. O. Scokaert, J. B. Rawlings, and E. S. Meadows, “Discrete-time stability with perturbations: Application to model predictive control,” *Automatica*, vol. 33, no. 3, pp. 463–470, 1997.

[183] F. Bonassi, “`ssnet`: a Python module for training State Space neural NETworks,” Available: https://github.com/bonassifabio/ssnet.

[184] F. Bonassi, A. La Bella, R. Lazzari, C. Sandroni, and R. Scattolini, “Supervised control of hybrid AC-DC grids for power balance restoration,” *Electric Power Systems Research*, vol. 196, p. 107107, 2021.

[185] R. Boffadossi, F. Bonassi, L. Fagiano, R. Scattolini, and A. Cataldo, “Safeguarded optimal policy learning for a smart discrete manufacturing plant,” in *14th IFAC Workshop on Intelligent Manufacturing Systems IMS 2022*. Elsevier, 2022, pp. 396–401, IFAC-PapersOnLine 55.2.

[186] F. Bonassi, A. La Bella, L. Fagiano, R. Scattolini, D. Zarrilli, and P. Almaleck, “Software-in-the-loop testing of a distributed optimal scheduling strategy for microgrids’ aggregators,” in *IEEE PES Innovative Smart Grid Technologies Europe*, 2020, pp. 985–989.

[187] A. La Bella, F. Bonassi, P. Klaus, and R. Scattolini, “A fully distributed control scheme for power balancing in distribution networks,” in *21st IFAC World Congress*, 2020, pp. 13 178–13 183, fAC-PapersOnLine 53.2.

[188] A. La Bella, F. Bonassi, C. Sandroni, L. Fagiano, and R. Scattolini, “A hierarchical approach for balancing service provision by microgrids aggregators,” in *21st IFAC World Congress*, 2020, pp. 12 930–12 935, iFAC-PapersOnLine 53.2.

[189] A. La Bella, F. Bonassi, M. Farina, and R. Scattolini, “Two-layer model predictive control of systems with independent dynamics and shared control resources,” in *15th IFAC Symposium on Large Scale Complex Systems (LSS)*, 2019, pp. 96–101, iFAC-PapersOnLine 52.3.

About the author

Fabio Bonassi was born in Brescia (BS), Italy, in 1994. He received his bachelor's and master's degrees in Automation and Control Engineering from Politecnico di Milano, Italy, in 2016 and 2018, respectively. In 2018 he completed, in collaboration with RSE S.p.A., his master's thesis entitled "Modeling and multi-layer optimal control of a mixed AC-DC grid", which received the "*Claudio Maffezzoni best thesis award*" from the Department of Electronics, Information, and Bioengineering, Politecnico di Milano, and the "*Si può fare di più*" best thesis award from Fondazione Cogeme.

In 2019, he was a research assistant at DEIB, Politecnico di Milano, and worked on predictive control applications in the context of managing prosumers' aggregators connected to the power system. Since November 2019, he is a PhD candidate in Information Technology at DEIB, Politecnico di Milano, Italy. In July 2021, he received the IFAC Young Author Award for the paper "Stability of discrete-time feed-forward neural networks in NARX configuration", presented at the 19th IFAC Symposium on System Identification. From August 2021 to December 2021 he was a visiting PhD student at the Uppsala University, Sweden.

During his PhD, he developed the Python module *ssnet* [183].

His main research interests lie in the identification methodologies and stability properties of recurrent neural networks used for learning dynamical systems, as well as their application in the context of model predictive control strategies. He is also interested in the application of machine learning and optimal control techniques in the context of *smart grid*.

Below is a list of all articles published, accepted for publication, or currently under review co-authored by Fabio Bonassi, in reversed chronological order.

Journal articles

- F. Bonassi, A. La Bella, M. Farina, and R. Scattolini, “Nonlinear MPC design for incrementally ISS systems with application to GRU networks,” *Automatica*, 2023, *(In preparation)*
- J. Xie, F. Bonassi, M. Farina, and R. Scattolini, “Robust offset-free nonlinear model predictive control learned by neural nonlinear autoregressive exogenous models,” *International Journal of Robust and Nonlinear Control*, 2022, *(Under review, arXiv preprint 2210.06801)*
- F. Bonassi, M. Farina, J. Xie, and R. Scattolini, “On Recurrent Neural Networks for learning-based control: recent results and ideas for future developments,” *Journal of Process Control*, vol. 114, pp. 92–104, 2022
- F. Bonassi and R. Scattolini, “Recurrent neural network-based Internal Model Control of unknown nonlinear stable systems,” *European Journal of Control*, p. 100632, 2022
- F. Bonassi, M. Farina, and R. Scattolini, “On the stability properties of gated recurrent units neural networks,” *Systems & Control Letters*, vol. 157, p. 105049, 2021
- E. Terzi, F. Bonassi, M. Farina, and R. Scattolini, “Learning model predictive control with long short-term memory networks,” *International Journal of Robust and Nonlinear Control*, vol. 31, no. 18, pp. 8877–8896, 2021
- F. Bonassi, A. La Bella, R. Lazzari, C. Sandroni, and R. Scattolini, “Supervised control of hybrid AC-DC grids for power balance restoration,” *Electric Power Systems Research*, vol. 196, p. 107107, 2021

Conference articles

- F. Bonassi, A. La Bella, G. Panzani, M. Farina, and R. Scattolini, “Deep Long-Short Term Memory networks: Stability properties and Experimental validation,” in *2023 European Control Conference (ECC)*, 2023, *(Under review)*
- F. Bonassi, J. Xie, M. Farina, and R. Scattolini, “An Offset-Free Nonlinear MPC scheme for systems learned by Neural NARX models,” in *2022 IEEE 61st Conference on Decision and Control (CDC)*, 2022, pp. 2123–2128
- F. Bonassi, J. Xie, M. Farina, and R. Scattolini, “Towards lifelong learning of recurrent neural networks for control design,” in *2022 European Control Conference (ECC)*, 2022, pp. 2018–2023
- R. Boffadossi, F. Bonassi, L. Fagiano, R. Scattolini, and A. Cataldo, “Safeguarded optimal policy learning for a smart discrete manufacturing plant,” in *14th IFAC Workshop on Intelligent Manufacturing Systems IMS 2022*. Elsevier, 2022, pp. 396–401, IFAC-PapersOnLine 55.2
- F. Bonassi, C. F. Oliveira da Silva, and R. Scattolini, “Nonlinear MPC for Offset-Free Tracking of systems learned by GRU Neural Networks,” in *3rd IFAC Conference on Modelling, Identification and Control of Nonlinear Systems (MICNON 2021)*, 2021, pp. 54–59, IFAC-PapersOnLine 54.14

- F. Bonassi, M. Farina, and R. Scattolini, “Stability of discrete-time feed-forward neural networks in NARX configuration,” in *19th IFAC Symposium on System Identification (SYSID 2021)*, 2021, pp. 547–552, IFAC-PapersOnLine 54.7
- F. Bonassi, E. Terzi, M. Farina, and R. Scattolini, “LSTM neural networks: Input to state stability and probabilistic safety verification,” in *Learning for Dynamics and Control*. PMLR, 2020, pp. 85–94
- F. Bonassi, A. La Bella, L. Fagiano, R. Scattolini, D. Zarrilli, and P. Almaleck, “Software-in-the-loop testing of a distributed optimal scheduling strategy for microgrids’ aggregators,” in *IEEE PES Innovative Smart Grid Technologies Europe*, 2020, pp. 985–989
- A. La Bella, F. Bonassi, P. Klaus, and R. Scattolini, “A fully distributed control scheme for power balancing in distribution networks,” in *21st IFAC World Congress*, 2020, pp. 13 178–13 183, fAC-PapersOnLine 53.2
- A. La Bella, F. Bonassi, C. Sandroni, L. Fagiano, and R. Scattolini, “A hierarchical approach for balancing service provision by microgrids aggregators,” in *21st IFAC World Congress*, 2020, pp. 12 930–12 935, iFAC-PapersOnLine 53.2
- A. La Bella, F. Bonassi, M. Farina, and R. Scattolini, “Two-layer model predictive control of systems with independent dynamics and shared control resources,” in *15th IFAC Symposium on Large Scale Complex Systems (LSS)*, 2019, pp. 96–101, iFAC-PapersOnLine 52.3