POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Development of a unitary approach for physics-based learning and decentralized control with recurrent neural network models

Tesi di Laurea Magistrale in
Automation and Control Engineering

Author: **Claudia Sbardi**

Student ID: 249440
Advisor: Prof. Marcello Farina
Co-advisors: Dott. Ing. Daniele Ravasio
Academic Year: 2024-2025

Abstract

Large-scale systems are widespread dynamical systems composed of several interconnected subsystems. They arise in complex industrial contexts such as chemical processes, energy networks, and transportation systems.
Their complexity comes from the high dimensionality, the presence of coupling among variables, and the frequent nonlinearities, which make the adoption of centralized control strategies difficult. To overcome the limitations related to high computational costs and communication requirements among physically coupled units, decentralized and distributed strategies represent effective solutions in terms of problem feasibility and control scalability.
However, their effectiveness depends on the availability of models capable of accurately reproducing the interconnection structure among the subsystems.
In this context, the need of a *physics-inspired* identification procedure arises and Recurrent Equilibrium Networks (RENs) allow the definition of nonlinear models that capture the interconnection topology. Structural constraints and physical properties of the system are enforced, so as to obtain models that are directly suitable for decentralized control.
This thesis focuses first on the identification of such control-oriented models, exploring different decomposition techniques and analyzing the impact of the sampling time. The methodology is validated on a nonlinear chemical plant: the objective is to identify and compare different identification procedures. Secondly, a decentralized nonlinear Model Predictive Control scheme for the class of structured REN models has been developed, in order to guarantee incremental stability properties, consistency among subsystems, and robustness with respect to bounded disturbances.

Keywords: large-scale nonlinear systems, RENs, physics-inspired system identification, decentralized NMPC design, control-oriented modeling

Abstract in lingua italiana

I sistemi a larga scala sono sistemi dinamici ampiamente diffusi, costituiti da numerosi sottosistemi dinamici interconnessi tra loro. Essi trovano applicazione in contesti industriali complessi, quali processi chimici, reti energetiche e sistemi di trasporto.
La loro complessità deriva dall'elevata dimensionalità, dalla presenza di forti interazioni tra le variabili e dalle frequenti nonlinearità, che rendono difficile l'adozione di strategie di controllo centralizzate. Per superare i limiti legati agli alti costi computazionali e alle esigenze di comunicazione tra unità fisicamente accoppiate, i sistemi di controllo decentralizzati e distribuiti rappresentano soluzioni efficaci in termini di fattibilità del problema e scalabilità del controllo.
Tuttavia, la loro efficacia dipende dalla disponibilità di modelli in grado di replicare fedelmente la struttura di interconnessione tra i sottosistemi; la derivazione di tali modelli fisici accurati risulta particolarmente complessa nel caso di sistemi nonlineari su larga scala.
In questo contesto nasce la necessità di proporre una procedura di identificazione *physics-inspired* basata su Recurrent Equilibrium Networks (RENs), che consenta di ottenere modelli nonlineari in grado di preservare la topologia delle interconnessioni. I vincoli strutturali e le proprietà fisiche del sistema sono imposti al fine di ottenere modelli direttamente utilizzabili per un controllo decentralizzato.
Questo lavoro di tesi si concentra in primo luogo sull'identificazione di tali modelli orientati al controllo, esplorando diverse tecniche di scomposizione, e analizzando l'impatto del tempo di campionamento in fase di discretizzazione. La metodologia è validata su un sistema chimico nonlineare con l'obiettivo di individuare e confrontare diverse procedure di identificazione. Parallelamente, è stato sviluppato uno schema di controllo predittivo robusto, decentralizzato e nonlineare applicabile alla classe di modelli REN strutturati, al fine di garantire proprietà di stabilità incrementale, consistenza tra sottosistemi e robustezza rispetto a disturbi limitati.

Keywords: sistemi su larga scala nonineari, RENs, identificazione physics-inspired, design NMPC decentralizzato, control-oriented modeling

Contents

Abstract **i**

Abstract in lingua italiana **iii**

Contents **v**

Introduction **1**

I.1 Motivations, objectives and state of the art 1

I.2 Contribution and structure of the thesis 5

I Physics-inspired learning of Recurrent Equilibrium Networks **7**

1 Recurrent Neural Networks **9**

1.1 Neural Networks in dynamical system modelling 9

1.2 The vanishing and exploding gradient problems and LSTM and GRU Recurrent Neural Networks 11

1.3 Gradient-free Recurrent Neural Networks 12

1.3.1 Neural Nonlinear Autoregressive models with eXogenous inputs (NNARXs) 12

1.3.2 Echo State Networks (ESNs) 13

1.3.3 Neural Nonlinear AutoRegressive models with eXogenous input Echo State Networks (NNARX-ESNs) 14

1.4 Physics-inspired learning of RNNs 15

2 Recurrent Equilibrium Networks and physics-inspired learning **19**

2.1 The RENs model 19

2.2 Hyperparameter definition of REN models 21

2.3 Learning REN Models 23

2.4 Structured Identification of REN models 24

3 Application of the learning approaches to the case study 33
3.1 Case study: Chemical Plant 33
3.2 Data acquisition and pre-processing 36
3.3 Non structured REN model 39
3.4 Structured REN models 43
3.4.1 Output-coupling Identification 43
3.4.2 Structured Matrices Identification 50
3.5 Comparative analysis of the models 55

II Decentralized control design for Structured Recurrent Equilibrium Network models 59

4 Centralized Robust Nonlinear Model Predictive Control 61
4.1 Statement of the problem 62
4.2 Incremental input-to-state stability 63
4.3 The nominal model 65
4.4 Constraint tightening 66
4.5 Online optimal control problem 66
4.5.1 The finite-horizon optimal control problem 67
4.5.2 Terminal ingredients 68

5 Decentralized Nonlinear Model Predictive Control of Structured Recurrent Equilibrium Network Models 71
5.1 Problem statement 71
5.2 Nominal model and tube-based definition 73
5.3 Online optimal control problem 75
5.4 Terminal ingredients 76

6 Offline design of Decentralized NMPC for Structured REN 79
6.1 Algorithm design 79
6.1.1 Design of K_x^i and K_s^i for δISS 80
6.1.2 Definition of the set of the coupling terms 84
6.1.3 Definition of the RPI set $\delta\mathbb{X}_i$ 86
6.1.4 Definition of the set $\delta\mathbb{S}_i$ 86
6.1.5 Consistency of the input and output constraints 87
6.2 Offline design procedure 88

6.3 Local δISS property 90
6.3.1 LMI for local δISS 90
6.3.2 Modification of the Offline Design Procedure using local δISS 92

7 Conclusions **95**

A Appendix - Structured Modelling **97**
A.1 Model Decomposition 97
A.2 The effect of the Sampling Time and Discretization in structured models . 100

Bibliography **103**

List of Figures **107**

List of Tables **109**

Introduction

I.1. Motivations, objectives and state of the art

Large-scale systems (LSSs) [18] are complex, dynamical, and often high-order systems composed of multiple interconnected subsystems. Typical examples include connected autonomous vehicles, transportation networks, power grids, and industrial or manufacturing processes. The structural and dynamical properties of these systems are the subject of active research, due to their significant practical importance in real-world applications. The interconnected structure, high dimensionality, distributed nature, uncertainties and frequent presence of nonlinearities make the centralized control of the LSSs significantly complex. From a computational perspective, centralized control design becomes impractical as the number of inputs and outputs increases; also, from the implementation perspective, the corresponding communication network may be infeasible as the physical distance between subsystems grows, as proper control requires synchronous and instantaneous communication among all units. To overcome centralized control limitations, decentralized and distributed control architectures [32] have been developed: both these schemes allow each subsystem or local controller to operate using partial information while still contributing to the global system objective.

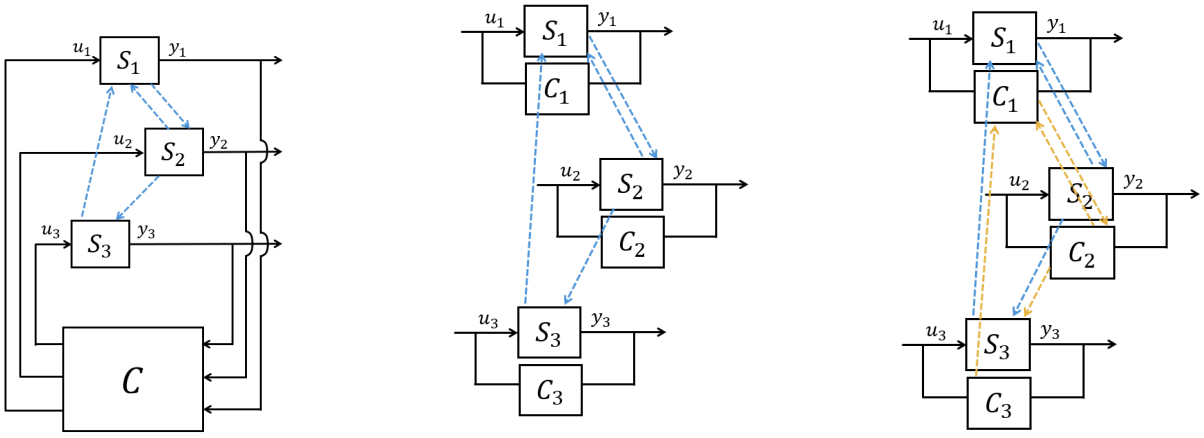


Figure 1: Comparison of control architectures: centralized, decentralized, and distributed.

In decentralized control [2], the system is structured into independent subsystems, each governed by a local controller. This structure enhances modularity and plug-and-play capabilities, but the absence of communication between controllers can limit overall stability and performance when subsystem interactions are significant.
Distributed control [19] concerns interacting local controllers attached to each input-output channel. In this framework, regulators are partially interconnected, ensuring coordination while maintaining a reduced communication burden. This advanced approach allows to define a trade-off between communication effort and performance, representing an intermediate solution between centralized and decentralized schemes.
Despite their advantages, both decentralized and distributed control architectures rely on the availability of accurate system models. However, deriving such models from physical equations for large-scale, nonlinear, and interconnected systems is often challenging, since conventional modeling approaches may be impractical and computationally expensive.

The recent advances in sensing technology, data acquisition, and machine learning have opened new opportunities for data-driven modeling and control design. The growing ability to collect measurements from the plants, the possibility to extract information from large data-sets thanks to advanced machine learning techniques, and the availability of open tools to train and deploy these algorithms are leading to a significant shift in data-driven control theory. In particular, Neural Networks (NNs) [1, 3] have emerged as powerful tools for modeling and control of complex nonlinear systems. They have shown remarkable performance in diverse domains such as computer vision, speech recognition, time-series forecasting, and autonomous systems, thanks to their ability to capture highly nonlinear input–output relationships.

Nevertheless, the implementation of purely data-driven neural network models, especially in the context of large-scale complex plants, presents a critical limitation: even when the model achieves good accuracy, it may fail to capture the underlying comprehensive physical/structural properties and complexities of the real system. This lack of physical consistency may result in unreliable or non-interpretable models, especially when the model is used for control design purposes. The first main objective of this thesis is therefore to propose a systematic physics-inspired approach to neural network modeling that includes basic selected features of the system while maintaining strong predictive capabilities: the proposed method aims to reconstruct not only the system outputs but also the intrinsic physical structure of the plant.

This framework is particularly suited for large-scale systems, where the number of interconnected states, inputs, and outputs makes both modeling and control extremely challenging. The proposed approach is structured around three main ideas:

- **Subsystem interaction modeling**: identifying the internal connections among subsystems and grouping variables that exhibit mutual influence. These "neighboring" relationships are explicitly encoded into the neural architecture to reflect the physical interconnection topology.
- **Physics-based constraints**: ensuring that the identified models preserve essential physical properties, enforcing contractivity and structural properties.
- **Control-oriented modeling** [17]: designing the model to be inherently suitable for decentralized or distributed control, thereby enhancing scalability, modularity, and computational efficiency while reducing complexity by neglecting non-relevant interconnections.

As better specified in the following, imposing a structured identification corresponds to imposing, to the learned models, features similar to the ones of the models commonly obtained by applying well-known model decomposition approaches [14]. In this framework it is possible to employ overlapping decompositions where each subsystem interacts with all others, providing rich coupling information but at high computational cost; in this case, the identification is more representative of the overall plant dynamics, but may fail to minimize the number of interconnections. On the other hand, by adopting a non-overlapping structure, the proposed model retains only the most significant interactions while maintaining feasible computational cost and interpretability, at a price of a possibly suboptimal dynamical precision.
For more details on model decompositions and their relationship with structural learning, please see Appendix A.1 and Chapter 2.4.

The selected model classes, in this work, are Recurrent Neural Networks (RNNs) [31]. Among the various classes of recurrent models — such as LSTM, GRU, NNARX, ESN, and hybrid NNARX-ESN architectures — this work focuses on Recurrent Equilibrium Networks (RENs) [28] due to their advantageous properties. Specifically:

- the identification problem can be formulated as a simple least-squares optimization,
- the modeling of nonlinear dynamics is possible in view of the presence of nonlinear activation functions,
- the replication of the original system's structure can be provided, thus producing a model directly usable for control design.

An additional but crucial aspect of this thesis is to study the influence of sampling time on the identified model's quality. Since neural network models depend on data sequences, the choice of the sampling time can significantly affect the model's ability to capture

dynamic behavior. A detailed analysis is carried out on a selected simulated case study to investigate how different sampling rates impact on the identified dynamics and on the overall network performance.

The second main objective of this thesis is the design of a robust decentralized MPC. As previously discussed, centralized NMPC formulations become impractical for large-scale systems. Decentralized and distributed control architectures [2, 32] overcome these limitations by letting local controllers operate with partial information while still pursuing the global control objective. These two control schemes introduce a trade-off between performance and communication effort [13, 14, 33], while enforcing modularity and plug-and-play capabilities [30]. However, these advantages require a model that explicitly reflects the interconnection structure of the plant.

This observation highlights a key aspect of the proposed model: the identification phase is not performed with the sole objective of minimizing the prediction error, but it is explicitly oriented toward control design [7]. For this reason, modeling and control design are not treated as separate steps, but as two complementary components of the same procedure. In particular, in this thesis REN models are learned in a structured, physics-inspired way so as to reflect the interconnection topology of large-scale systems and to be directly applicable within a decentralized robust MPC framework.

The decentralized controller discussed in this thesis relies on the tube-based robust MPC [22], similarly to [13]. The tube-based MPC formulation adopted in this work relies on the availability of a control law capable of confering to the system strong stability properties, i.e., δISS [25]. This property ensures that the effect of disturbances, modeling errors, and coupling terms remains bounded and can be compensated. The existence of robust positively invariant sets and the derivation of the corresponding constraint-tightening strategy can be formulated, in a rigorous way, in terms of LMIs, leading to a computationally feasible offline design procedure [20, 23, 35].

The second part of this thesis focuses indeed on the development of a suitable nonlinear Model Predictive Control framework for structured REN models. Starting from a centralized formulation, a decentralized tube-based control architecture is derived, where suitable LMI conditions are imposed to guarantee consistency among subsystems, preservation of the δISS property, and robust constraint satisfaction.

In particular, in this thesis, LMIs also play a crucial role in guaranteeing the consistency among the interconnected subsystems. Since each local controller is designed using partial information, suitable conditions must be imposed on the coupling terms. These consistency conditions can be expressed as local matrix inequalities involving only subsystem

parameters and neighboring interactions, thus preserving the scalability of the design procedure.

I.2. Contribution and structure of the thesis

The main contributions of this thesis can be summarized as follows:

- **Physics-Inspired RNN Identification for Large-Scale Systems**
  A theoretical framework for Recurrent Equilibrium Network (REN) models is presented, including their formulation, the definition of hyperparameters, and the learning procedure based on a least-squares optimization problem. The approach imposes physical and structural properties through Linear Matrix Inequalities, resulting in control-oriented nonlinear models that guarantee interpretability and modularity.
- **Decomposition strategies and validation on a benchmark chemical plant**
  Identification methods based on different model decomposition approaches are developed and compared. The effectiveness of the different approaches is assessed on a case study consisting in a nonlinear chemical plant. The objective is to highlight the trade-off between model accuracy, physical consistency, and computational complexity, as well as the impact of the sampling time on the quality of the learned dynamics.
- **Robust Decentralized NMPC Design for RENs**
  A decentralized controller design method is developed, based on robust tube-based NMPC. The controller design must guarantee that the LMI conditions ensuring well-posedness, δISS, recursive feasibility, and compliance with neighboring interconnections are fulfilled.

This thesis is divided in two main parts, highliting the two core components of the proposed framework.

Part I – Physics-inspired learning of Recurrent Equilibrium Networks focuses on the modeling and identification phase.

- Chapter 1 introduces recurrent neural networks for dynamical system modeling: gradient-based and gradient-free Recurrent Equilibrium Network models are presented.
- Chapter 2 presents the formulation of Recurrent Equilibrium Networks, defines the notions of contractivity and well-posedness, introduces the learning procedure, and the structured identification methodologies.

- Chapter 3 applies the proposed identification approaches to the chemical plant case study. The data acquisition and pre-processing steps are described, the identification procedures of non-structured and structured REN models are performed and compared in terms of prediction accuracy and computational complexity.

Part II – Decentralized control design for Recurrent Equilibrium Network models focuses on the control problem design using the REN models.

- Chapter 4 introduces the centralized robust nonlinear model predictive control framework, including the perturbed model description, the incremental stability properties, the constraint-tightening strategy, and the formulation of the finite-horizon optimal control problem.
- Chapter 5 develops the decentralized tube-based robust NMPC scheme for structured REN models. The problem statement, the definition of the nominal model, and the resulting online optimization problem with terminal ingredients are detailed.
- Chapter 6 develops the offline design algorithm for the decentralized NMPC defined in Chapter 5. The consistency conditions of the control architecture are derived through LMIs, and the offline design procedure is detailed.
- The Appendix provides additional material on model decomposition techniques and on the effect of sampling time and discretization on the models.

Part I

Physics-inspired learning of Recurrent Equilibrium Networks

1 | Recurrent Neural Networks

1.1. Neural Networks in dynamical system modelling

In recent decades, Artificial Neural Networls (ANNs) [1, 3] have emerged as powerful computational tools for addressing complex problems that are challenging to solve using traditional statistical or rule-based approaches. ANNs are often referred to as "neurocomputing" tools, as they work similarly to biological systems, e.g., in view of their characteristics, such as nonlinearity, high parallelism, robustness, ability to handle noisy information, and strong generalization capabilities.
Neural Networks can have two different structures: the traditional implementation, known as Feed Forward Neural Network (FFNN), and the dynamic form called Recurrent Neural Network (RNN).

Feed-Forward Neural Networks [34] are composed of "neurons", that receive an input signal, compute a response, and transmit it to the following layer. Multi-Layer FFNNs (MLFs) are the most widely used: they consist of neurons organized in layers and described by mapping functions. The importance of connections between neurons is captured by weights assigned to each link. The most commonly adopted training technique is the backpropagation algorithm [9], which adjusts synaptic weights to reduce the difference between predicted outputs and desired targets. Although commonly used in the past, FFNN predictors result in poor accuracy when representing long-term dynamics, due to their inherent lack of memory.

This thesis focuses on Recurrent Neural Networks [31], which, unlike FFNNs, include cycles that allow to transmit information back to the same nodes. These internal self-interconnections extend the functionalities of FFNNs, by training the network not only using the current input, but also past information.

Recurrent Neural Network models can be expressed as a dynamical Multi-Input Multi-

Output (MIMO) state-space model of the following general type

$$x(t+1) = f(x(t), u(t); \theta) \tag{1.1a}$$
$$y(t) = g(x(t), u(t); \theta) \tag{1.1b}$$

where

- $x(t) \in \mathbb{R}^{n_x}$ is the so-called hidden state vector, which captures the presence of internal loops in the RNN and the self-interconnection dynamics
- $u(t) \in \mathbb{R}^{n_u}$ is the input vector
- $y(t) \in \mathbb{R}^{n_y}$ is the output vector (i.e the vector of measurable variables of the system)

Equation (1.1a) represents the temporal dependency of the hidden (or internal) state on the input, expressed as a recursive difference equation, and Equation (1.1b) represents the output equation. Key features of RNN models are the presence of network weights, collected in θ, to be tuned during the training procedure, and suitable nonlinear activation functions, such as hyperbolic tangents or sigmoids.

The training algorithm for RRNs is an extension of the backpropagation method, known as Backpropagation Through Time (BPTT). Nevertheless, in case of RNNs, it presents two signifincant issues—the so called vanishing and exploding gradient problems—that can severely affect the learning process. The following sections will discuss specific RNN architectures designed to overcome these limitations.

Note that, in the following chapters, we specifically refer to on a novel class of RNNs, known as Recurrent Equilibrium Network, extensively used in physics-inspired learning and control design in this thesis. The class of RENs is a class of RNNs recently proposed in [28]. This new model class has built-in guarantees of contractivity (i.e., a strong form of nonlinear stability) and robustness. In this work, however, a slight abuse of terminology is adopted: for more generality, we define RENs as the networks that share the same formulation as the ones introduced in [28], but without necessarily satisfying the theoretical guarantees of contractivity and robustness. This is done to allow for describing a wider class of plant models. Features about REN's structure and training will be detailly provided in Chapter 2.

1.2. The vanishing and exploding gradient problems and LSTM and GRU Recurrent Neural Networks

RNNs are commonly trained using the Back Propagation Through Time [5] algorithm, that is an extension of the traditional backpropagation algorithm used, in case of FFNNs, for layer-by-layer training. The algorithm consists of unrolling the network across multiple time steps, transforming it into a FFNN with as many hidden states as the number of temporal iterations. Starting from available data $u(t)$ and $y(t)$, the input sequence $u(t)$ is sequentially processed step-by-step, the hidden state $x(t)$ is updated and the predicted output $\hat{y}(t)$ is generated.
The network is then trained by minimizing the loss function, defined as the Mean Square Error (MSE) between the RNN prediction and the measured output. The loss function takes the general form

$$\mathcal{L} = \sum_{t=1}^{T} \ell_t\big(y(t), \hat{y}(t)\big) \tag{1.2}$$

where ℓ_t is the individual loss at each time step. The training is performed by defining the value of the weight θ that minimizes the loss function [5]. To do this, the loss gradient must be computed. However, this algorithm is subject to two main issues, presented in [24]:

- **Vanishing gradient**: gradients can become exceedingly small as they propagate backward across many time steps, making it difficult for the network to learn long-range dependencies;
- **Exploding gradient**: gradients can sometimes grow exponentially, leading to very large updates and causing the network parameters to oscillate or diverge.

To overcome these problems, the Truncated Back-Propagation Through-Time (BPTT) algorithm has been introduced, where the sum of the loss function is truncated at a computationally convenient size. In order to preserve key dependencies, the upper bound of the time steps can be intended as a moving window over past time steps.
In addition, to address the issues of vanishing and exploding gradient problems, a number of RNN architectures [7] have been proposed, e.g., Long Short-Term Memory (LSTM) networks, Gated Recurrent Units (GRUs), Echo State Networks, and Neural Nonlinear Autoregressive Models with eXogenous Inputs (NNARX).
Long Short-Term Memory networks (LSTMs) were introduced to properly handle the vanishing gradient problem and to learn long-term dependencies over a long-range time-steps. To achieve this, LSTMs store and regulate information through specialized struc-

tures known as *gated cells*. Although they are powerful tools for system identification, their training can be computationally demanding due to the large number of free parameters involved. Gated Recurrent Units (GRUs) represent a simplified variant of LSTMs, offering a trade-off between complexity and modeling performance.
The training of both LSTMs and GRUs relies on the BPTT algorithm, which allows them to mitigate the vanishing gradient issue to some extent.
A description of the other (here denoted gradient-free) recurrent architectures will be provided in the following Section 1.3.

1.3. Gradient-free Recurrent Neural Networks

This section presents a different class of Recurrent Neural Network including NNARX, ESN, and the hybrid NNARX-ESN. These RNNs overcome the vanishing and exploding gradient problems by adopting an alternative identification approach, which provides the foundation of the learning procedure for Recurrent Equilibrium Networks (RENs) adopted in this work and thoroughly presented in Chapter 2.

1.3.1. Neural Nonlinear Autoregressive models with eXogenous inputs (NNARXs)

In NNARX models [6], the output is computed as a nonlinear regression over past d input and output data. For this reason, NNARXs are different to common RNNs, since recursivity of past data is introduced through lagged inputs and outputs, while the hidden state is absent. Specifically the output at time t is expressed as:

$$y(t) = \eta(u(t-d), ..., u(t-1), u(t), y(t-d-1), ..., y(t-1), \theta) \tag{1.3}$$

where d is the delay, $\eta(\cdot)$ is the nonlinear mapping function, typically implemented through a Multi-Layer Perceptron (MLP), and θ the set of free trainable parameters.

By adopting a prediction-error minimization approach, the Jacobian associated with the loss minimization can be derived analytically. Although the gradient is still employed to train the network, the learning procedure differs from the BPTT algorithm, as presented in [21] which presents gradient-related problems.

Note that, as described in [6], we can provide a state-space formulation of (1.3). To do

so, we define the state vector containing past data as:

$$x(i,t) = \begin{bmatrix} y(t-d+i) \\ u(t-d-1+i) \end{bmatrix}$$

for $i = 1, ..., d$. By defining $x(t) = \begin{bmatrix} x(1,t) \\ \vdots \\ x(d,t) \end{bmatrix} \in \mathbb{R}^{n_y+n_u}$, equation (1.3) can be rewritten in state-space form as:

$$x(t+1) = Ax(t) + B_u u(t) + B_x \eta(x(t), u(t); \theta) \tag{1.4a}$$
$$y(t) = Cx(t) \tag{1.4b}$$

where

- $A = \begin{bmatrix} 0 & I & 0 & \cdots & 0 \\ 0 & 0 & I & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \ddots & I \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix}$, where each block is in $\mathbb{R}^{(n_y+n_u)\times(n_y+n_u)}$
- $B_u = \begin{bmatrix} 0 & ... & 0 & \tilde{B}_u^\top \end{bmatrix}^\top$, where each block is in $\mathbb{R}^{(n_y+n_u)\times n_u}$ and $\tilde{B}_u = \begin{bmatrix} 0_{n_y \times n_u} \\ I_{n_u \times n_u} \end{bmatrix}$
- $B_x = \begin{bmatrix} 0 & ... & 0 & \tilde{B}_x^\top \end{bmatrix}^\top$, where each block is in $\mathbb{R}^{(n_y+n_u)\times n_y}$ and $\tilde{B}_x = \begin{bmatrix} I_{n_y \times n_y} \\ 0_{n_u \times n_y} \end{bmatrix}$
- $C = \begin{bmatrix} 0 & ... & 0 & \tilde{C} \end{bmatrix}$, where each block is in $\mathbb{R}^{n_y \times (n_y+n_u)}$ and $\tilde{C} = \begin{bmatrix} I_{n_y \times n_y} & 0_{n_y \times n_u} \end{bmatrix}$.

1.3.2. Echo State Networks (ESNs)

The ESN [8] model can take the following form

$$x(t+1) = \tanh\left(W_x x(t) + W_u u(t) + W_y y(t)\right) \tag{1.5a}$$
$$y(t) = W_{\text{out-x}} x(t) + W_{\text{out-u}} u(t) \tag{1.5b}$$

ESNs include the so-called hidden layer (1.5a), known as dynamical reservoir, through which the evolution of $x(t)$ can be defined. The connection weight matrices W_u, W_x, W_y are randomly generated and fixed, whereas $W_{\text{out-x}}, W_{\text{out-u}}$ must be identified in the training phase. The output is computed as a linear function of the free parameters according to

(1.5b), thus avoiding the need to handle nonlinearities. This allows to formulate the learning problem as a Quadratic Minimization problem, that avoids the vanishing and exploding gradient issues. In fact, the loss function (1.2) can be now expressed as a quadratic function:

$$\mathcal{L}(W_{\text{out}}) = \frac{1}{N_s}\sum_{t=1}^{N_s}\big(y(t) - (W_{\text{out-x}}x(t) + W_{\text{out-u}}u(t))\big)^2 = ||Y - \Psi W_{\text{out}}^T||^2 \tag{1.6}$$

where

$$Y = \begin{bmatrix} y(1)^T \\ \vdots \\ y(N_d)^T \end{bmatrix}, \qquad \Psi = \begin{bmatrix} x(1)^T & u(1)^T \\ \vdots & \vdots \\ x(N_d)^T & u(N_d)^T \end{bmatrix}, \qquad W_{\text{out}} = \begin{bmatrix} W_{\text{out-x}} \\ W_{\text{out-u}} \end{bmatrix}$$

Note that in (1.8), Ψ is known, since the values of $x(t)$ are provided by computing the solution to (1.5a), considering that $u(t)$ and $y(t)$ are given data.

The ESN hyperparameter W_x is designed in such a way that the Echo State Property [8] is fulfilled, according to which the current network state $x(t)$ gets asymptotically independent of its initial value when computing the recursive solution to (1.5a).
As shown in [8], a sufficient condition for the Echo State Property matrix W_x is diagonally Schur stable (i.e, there exists a diagonal $P > 0$ such that $W_x^T P W_x - P < 0$).

However, ESNs presents also some disadvantages: matrices W_x, W_u and W_y are fixed hyperparameters, which drastically reduces the number of degrees of freedom; moreover, these networks are not well suited for capturing complex nonlinear dynamics. Therefore, more general classes of neural networks are commonly required for this scope.

1.3.3. Neural Nonlinear AutoRegressive models with eXogenous input Echo State Networks (NNARX-ESNs)

Neural Nonlinear AutoRegressive models with eXogenous input Echo State Networks (NNARX-ESNs) [12] integrate the structure of NARX and ESN models.
As in ESNs, the network includes a randomly initialized resevoir (i.e., Equation (1.7a)), which allows to reduce the complexity of the training algorithm. The NNARX-ESN model can be expressed as:

$$x(t+1) = f_x\big(W_x x(t) + W_\phi \phi(t) + W_y y(t)\big) \tag{1.7a}$$

$$y(t+1) = W_{\text{out-}x}x(t) + W_{\text{out-}\phi}\phi(t-1) \tag{1.7b}$$

where $x(t)$ is the reservoir state vector at time t, y is the output vector at time t and ϕ is the regressor vector composed of past input and past and present output values, introducing the autoregressive component, i.e.,

$$\phi(t) = [y(t)^T \quad ... \quad y(t-d_y+1)^T \quad u(t)^T \quad ... \quad u(t-d_u+1)^T]$$

Moreover matrices W_x, W_ϕ and W_y are the weight matrices that are fixed by random initialization, while W_{out-x} and $W_{out-\phi}$ are the output weight matrices to be optimized. Lastly, $f_x(\cdot)$ is the reservoir vector function, typically a sigmoid or an hyperbolic tangent.

As in Section (1.3.2), the ESN-type reservoir equation (1.7a) introduces the possibility of solving the minimization of the loss function as a Least Squares problem, thereby avoiding limitations of the vanishing and exploding gradients. The loss function assumes the same form of (1.8). i.e.,

$$\mathcal{L}(W_{\text{out}}) = \frac{1}{N_s}\sum_{t=1}^{N_s}\left(y(t) - (W_{\text{out-}\chi}\chi(t) + W_{\text{out-}\phi}\phi(t))\right)^2 = ||Y - \Psi W_{\text{out}}^T||^2 \tag{1.8}$$

where

$$Y = \begin{bmatrix} y(1)^T \\ \vdots \\ y(N_d)^T \end{bmatrix}, \qquad \Psi = \begin{bmatrix} \chi(1)^T & \phi(1)^T \\ \vdots & \vdots \\ \chi(N_d)^T & \phi(N_d)^T \end{bmatrix}, \qquad W_{\text{out}} = \begin{bmatrix} W_{\text{out-}\chi} \\ W_{\text{out-}\phi} \end{bmatrix}$$

As also remarked in Section 1.3.2, the Echo State Property of (1.7a) ensures that reservoir states remain stable and are influenced by most recent inputs, rather than distant past inputs.

1.4. Physics-inspired learning of RNNs

Modeling complex dynamic systems purely from data can be challenging, especially when the system is affected by strong nonlinearities, long-term dependencies, or interactions among multiple interconnected subsystems. Purely data-driven approaches can capture nonlinear dynamics, but they often suffer from limited interpretability and may produce models that are inconsistent with known physical laws. These limitations become particularly critical in complex industrial applications, where prior physical knowledge of the system and subsystem interactions should be used to guide the modeling process.
To address these challenges, this work investigates the opportunities offered by the so-

called physics-inspired approaches to RNN learning. The key idea is to integrate physical knowledge of the system with the model architecture and the training procedure, resulting in models that are accurate, interpretable, and physically consistent. In this thesis, this approach enables the model to capture dynamic couplings among subsystems, making the model directly applicable within a distributed control framework;

The training methodology proposed in this thesis explicitly combines data-driven learning with prior knowledge about the system topology and interactions. Connections between subsystems are established according to known physical interactions—such as material flows, energy transfers, or information dependencies—while negligible links are discarded. The system architecture is retained by training a sparsely connected RNN, where each block represents either an isolated subsystem or a cluster of interconnected subsystems. Rather than decomposing a centralized plant model into multiple parts, this approach directly determines the decomposed model by exploiting the dataset and the plant's sparsity. The adopted structure ensures that the model captures the effects of dynamic couplings among neighboring subsystems, and it enhances interpretability and specific performances.

The benefits of this physics-inspired approach can be summarized from both the modeling and the control perspectives as follows.

- From a modeling perspective, this approach improves the reliability of the model under varying operating conditions, and ensures consistency with both the structural properties and the dynamic features of the system.
- From a control design perspective, it provides manageable control-oriented models, which can be directly used in the design of distributed and decentralized control schemes.

It is important to make a further comment about two fundamental (and partially interlaced) aspects of this framework, i.e.,

- the relationship between structural system interconnection network and the model complexity;
- the relationship between the sampling time and the model approximation.

Regarding the former point, note that the interconnection network between subsystems is a priori defined in some cases, but also strongly depends upon the modeling choice that we do. At the same time, it strongly impacts on the complexity of the resulting model. This lesson can be learned, indeed, by considering the possible decisions that can be made (see Appendix A.1) about model decomposition. For instance, the choice of decomposing a

given system model using a non-overlapping decomposition makes it possible to minimize the number of interconnections betweeen subsystems (by considering, in a few words, only the physical "direct" connections between the different parts of the plant, which are often "state couplings"), and the dimensionality of the subsystems' models. On the other hand, the choice of adopting an overlapping decomposition requires to explicitly represent also the "indirect" interconnections, to the point that, in case of completely overlapping decompositions, interconnections must be considered (and in particular "input couplings") in all cases when non-zero transfer functions between subsystems (i.e., from an input u_j to an output y_i, for $j \neq i$) exist. This, as a byproduct, leads also to an increased dimensionality of the obtained models (i.e., in view of the presence of overlapping states).

In this thesis we regard this issue from a different viewpoint–that of learning–but the lesson learned in the "model decomposition realm" has an impact on our modeling choices. More specifically, in case we opt to define a sparse interconnection network (based on just the main physical interconnections between the parts of the plant) through states, we are implicitly enforcing a modeling choice consistent with non-overlapping decompositions, implying that minimal (also in terms of number of variables) models may be obtained.

On the other hand, if we need to impose couplings through inputs (as done, e.g., in [22]), we are implicitly adopting an approach consistent with overlapping decompositions, characterized by possibly very dense (e.g., all-to-all) interconnection networks and non-minimal models.

As far as the sampling time is concerned, we need to defer the reader to the Appendix again. Indeed, In Appendix A.2 it is clearly indicated that, while physical plant models in continuous-time may display a clear sparse structure, exact discretization (i.e., ZOH) leads to corresponding discrete-time models where such structure is not well defined anymore. In [15], in order to devise structured discrete-time models suitable for decentralized and distributed controller design, it is clarified that approximation methods (e.g., the so-called mixed Euler-ZOH) must be adopted. These methods clearly induce approximations which approach to the real model only if $T_s \to 0^+$, where T_s is the sampling time. This consideration sheds some light also in learning structured models, since it clarifies very well that any learned structured discrete-time model is inherently affected by approximations, and that the approximation error can be reduced only by reducing the sampling time. Therefore, the choice of T_s becomes particularly important and critical.

2 | Recurrent Equilibrium Networks and physics-inspired learning

2.1. The RENs model

The class of Recurrent Equilibrium Networks (RENs) [28, 29] is a novel class of models designed to represent nonlinear dynamical systems. Their construction is based on the feedback interconnection between a linear dynamical system G and an implicit function, defined by non-linear operator σ, as represented in Figure 2.1. Depending on σ, RENs may represent a deep neural network, while retaining a clear and compact definition.

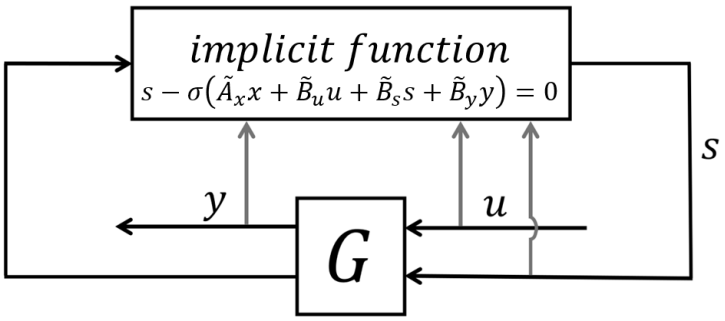


Figure 2.1: REN as a feedback interconnection of a linear system G and a nonlinear activation σ

The REN structure is a state-space representation expressed as:

$$x(t+1) = A_x x(t) + B_u u(t) + B_s s(t) + B_y y(t) \tag{2.1a}$$
$$s(t) = \sigma(\tilde{A}_x x(t) + \tilde{B}_u u(t) + \tilde{B}_s s(t) + \tilde{B}_y y(t)) \tag{2.1b}$$
$$y(t) = C x(t) + D_u u(t) + D_s s(t) \tag{2.1c}$$

where

- $x \in \mathbb{R}^{n_x}$ denotes the internal state vector of the network. The latter is characterized by matrices $A_x \in \mathbb{R}^{n_x \times n_x}$,$B_u \in \mathbb{R}^{n_x \times n_u}$, $B_s \in \mathbb{R}^{n_x \times n_s}$, and $B_y \in \mathbb{R}^{n_x \times n_y}$.
- $s \in \mathbb{R}^{n_s}$ is the solution to the equilibrium equation (2.1b), also known as "implicit network". The latter is characterized by matrices $\tilde{A}_x \in \mathbb{R}^{n_s \times n_x}$,$\tilde{B}_u \in \mathbb{R}^{n_s \times n_u}$, $\tilde{B}_s \in \mathbb{R}^{n_s \times n_s}$, $\tilde{B}_y \in \mathbb{R}^{n_s \times n_y}$, and by the activation function $\sigma = [\sigma_1(\cdot) \ ... \ \sigma_{n_s}]^T$, that is a decentralized vector of scalar sigmoid functions.
- $u \in \mathbb{R}^{n_u}$ denotes the input vector.
- $y \in \mathbb{R}^{n_y}$ denotes the output vector.

The term "equilibrium" derives from the property according to which any solution of the nonlinear equation (2.1b), is also an equilibrium point of $s(t+1)$. In fact, REN models can be interpreted as two-timescale or singular perturbation models: the *fast* dynamics in s are assumed to reach the equilibrium within one step of the *slow* dynamics represented by x.

We define *Explicit RENs* as the class of RENs having $\tilde{D}_s = 0$. These models have been studied, e.g., in [27]. In this case the network (2.1b) reduces to a single-layer neural network. These models are certainly more effective than a single-layer feed forward neural networks and than linear models. However, they are less flexible and representative of RENs having $\tilde{D}_s \neq 0$, which are referred to as *Implicit RENs*. From this point onward and throughout this work, the latter will be considered.
Depending on the value of $\tilde{D}_s$, the implicit Equation (2.1a) may or may not admit a unique solution $s(t)$ for a given $x(t), u(t)$.

The following assumption is made on the function σ.

Assumption 2.1.1. Each component $\sigma_i(\cdot) : \mathbb{R} \to \mathbb{R}, i = 1, ..., n_s$ is a sigmoid function such that $\sigma_i(v_i(t)) \in [-1, 1], \forall v_i \in \mathbb{R}$. The activation function must be bounded, piecewise differentiable and slope-restricted in $[0, 1]$, i.e.,

$$0 \leq \frac{\sigma(b) - \sigma(a)}{b - a} \leq 1, \forall a, b \in \mathbb{R}, a \neq b \tag{2.2}$$

Also, $\sigma_i(\cdot)$ is Lipschitz continuous with unitary Lipschitz constant and one and only one inflection point in $v_i = 0$.

■

An example of activation function is represented in Figure 2.2, showing $\sigma_i(v_i) = \tanh(v_i)$.

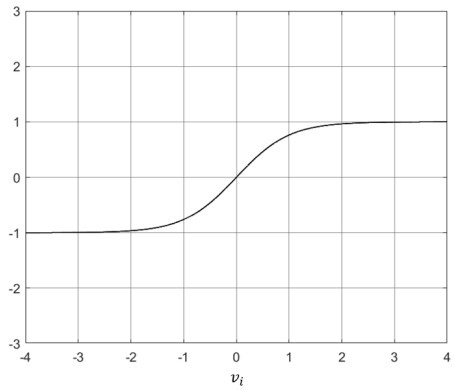


Figure 2.2: Function $\sigma_i(v_i) = tanh(v_i)$

A key property used in this work to derive useful Linear Matrix Inequalities (LMIs) is the incremental sector condition, provided by the following lemma [26].

Lemma 2.1. *Under Assumption 2.1.1,* $\forall h_i > 1, \exists \bar{v}_i(h_i) \geq 0$ *such that* $\forall (v_i, v_i + \Delta v_i) \in [-\bar{v}_i(h_i), \bar{v}_i(h_i)]^2$*, it holds that*

$$\Delta s_i(\Delta v_i - h_i \Delta s_i) \geq 0 \tag{2.3}$$

where $\Delta s_i = s_i(v_i + \Delta v_i) - s_i(v_i)$

A model (2.1) is said to be *well-posed* if it provides a unique solution of $s(t)$ for a given $x(t), u(t)$ (i.e., if it yelds a unique response to any initial conditions and input). As stated in [29], the equilibrium network is well-posed if $\exists \Lambda > 0$ diagonal such that

$$2\Lambda - \Lambda \tilde{B}_s - \tilde{B}_s{}^T \Lambda > 0 \tag{2.4}$$

Note that, when $\tilde{D}_s$ has a triangular structure (i.e. in the case of the so-called *acyclic RENs*), the model is well-posed by definition. As detailed in [29], acyclic RENs are very practical since the elements of $s(t)$ can be explicitely computed row-by-row from (2.1b). Compared to the general-type RENs, the acyclic RENs are easier to implement and, in most cases, they provide models of excellent quality.

2.2. Hyperparameter definition of REN models

The first objective of this work is to identify a suitable model from a set of candidates using data. In this thesis, in order to simplify the learning process of REN models, we adopt an approach similar to the one described in Chapter 1.3.2 and we consider the

matrices in (2.1a) and (2.1b) as hyperparameters.
Namely, the training procedure begins by defining matrices $B_u, \tilde{B}_u, B_y, \tilde{B}_y, B_s$ and $\tilde{B}_s$ as fixed random hyperparameters. Matrices A_x and $\tilde{A}_x$ are also treated as hyperparameters, but they are subjected to specific constraints.
In particular, it is possible to impose contractivity to the untrained model, which guarantees that trajectories of the system converge exponentially towards each other, independently of their initial conditions, thus ensuring the well-posedness (see Theorem 2.1 from [29]) of the network dynamics.

Definition 2.2.1. A model (2.1) is said to be *contracting with rate* $\alpha \in (0, 1)$ [29] if, for any two initial conditions x_0^a, $x_0^b \in \mathbb{R}^{n_x}$, given the same input sequence $u : \mathbb{N} \to \mathbb{R}^{n_u}$, the state sequences x_t^a, x_t^b satisfy for some $K > 0$ the following

$$|x_t^a - x_t^b| \leq K\alpha^t |x_0^a - x_0^b| \tag{2.5}$$

Under Assumption 2.1.1 described by (2.2), the following theorem gives the conditions for contracting RENs [29].

Theorem 2.1. *If there exists a matrix* $P = P^T > 0$ *and* $\Lambda \in \mathbb{D}_+$ *, where* $\mathbb{D}_+$ *is the set of diagonal positive definite matrices, such that*

$$\begin{bmatrix} \bar{\alpha}^2 P & -\tilde{A}_x^T \Lambda \\ -\Lambda \tilde{A}_x & W \end{bmatrix} - \begin{bmatrix} A_x^T \\ B_s^T \end{bmatrix} P \begin{bmatrix} A_x^T \\ B_s^T \end{bmatrix}^T > 0 \tag{2.6}$$

where $W = 2\Lambda - \Lambda \tilde{B}_s - \tilde{B}_s^T \Lambda$*, then the network is well-posed and contracting with some rate* $\alpha < \bar{\alpha}$*.*

The maximum rate of convergence $\bar{\alpha}$, for instance, can be chosen as the spectral radius of the system (i.e the maximum eigenvalue of the discrete-time linearized system), which can be inferred from the settling time of the system trajectories, e.g., in response to step-wise inputs.
The complete LMI is derived in [29] by combining the Lyapunov condition (2.6) with the sector condition, derived in Lemma 2.1. Applying the Schur complement, the resulting LMI for the contractivity of the untrained model is the following:

$$\begin{bmatrix} \bar{\alpha}^2 P & -\tilde{A}_x^T \Lambda & A_x^T P \\ -\Lambda \tilde{A}_x & W & B_s^T P \\ P A_x & P B_s & P \end{bmatrix} > 0 \tag{2.7}$$

Depending on the problem, it is possible to influence the position of the eigenvalues of matrices A_x and $\tilde{A}_x$ by adding some constraints on the trace of the square matrix A_x and on the rectangular-trace[1] of the non-square matrix $\tilde{A}_x$.

2.3. Learning REN Models

After the definition of the model hyperparameters, the only free identifiable parameters are matrices C, D_u and D_s of (2.1c). For notational reasons, we define $\theta = [C \quad D_u \quad D_s]$. The identification of θ is performed as follows:

1. Letting N_s be the number of the available data samples, the data $u(t)$ and $y(t)$ for $t = 1, ..., N_s$ are used to feed equations (2.1a, 2.1b), from a random initial condition $x(1)$. At this point all values of internal states $x(t)$ and nonlinearities $s(t)$ are available for $t = 2, ..., N_s$.

2. Similarly to Section 1.2, the goal is to select the value of θ that minimizes the quadratic error between the data and the predicted output:

$$J(\theta) = \sum_{t=1}^{N_s} \left\| y(t) - \hat{y}(t) \right\|^2 = \sum_{t=1}^{N_s} \left\| y(t) - \begin{bmatrix} C & D & Ds \end{bmatrix} \begin{bmatrix} x(t) \\ u(t) \\ s(t) \end{bmatrix} \right\|^2 = \sum_{t=1}^{N_s} \left\| y(t) - \theta \begin{bmatrix} x(t) \\ u(t) \\ s(t) \end{bmatrix} \right\|^2 \tag{2.8}$$

Note that the optimization problem is quadratic. In fact, it reduces to a Least-Squares formulation, avoiding the vanishing and exploding gradient issues. Additionally, Linear Matrix Inequalities (LMIs) can be incorporated into the optimization framework to enforce desired physical properties of the modeled system. Consequently, the overall identification procedure may adopt a physics-inspired perspective, enhancing the interpretability and physical consistency of the resulting model.

Note that the identification procedure used in this work can be interpretated as an hybrid having features of both the Prediction Error Method (PEM) and the Simulation Error Method (SEM) approaches. Note that, one the one hand, PEM provides accurate one-step-ahead predictions but may fail to reproduce the long-term dynamics of the system, for

[1] **Rectanguar-trace**: For a rectangular matrix $M \in \mathbb{R}^{n \times m}$, we define the generalized trace as

$$\mathrm{tr}_r(M) = \sum_{i=1}^{\min(m,n)} M_{ii},$$

i.e., the sum of the diagonal entries up to the smallest dimension. This quantity is basis–dependent and differs from the nuclear norm, which is instead defined as the sum of the singular values.

example when the identified model structure is wrong; on the other hand, SEM focuses on reproducing the overall system behavior through multi-step simulations, but may provide incorrect internal parametrization. By mixing the two approaches, the proposed method exploits the short-term accuracy of PEM and the global consistency of SEM, capturing both the local and long-term dynamics of the system.

At this point, the learned REN model is:

$$x(t+1) = (A_x + B_y C)x(t) + (B_u + B_y D)u(t) + (B_s + B_y D_s)s(t) \tag{2.9a}$$

$$s(t) = \sigma\big((\tilde{A}_x + \tilde{B}_y C)x(t) + (\tilde{B}_u + \tilde{B}_y D)u(t) + (\tilde{B}_s + \tilde{B}_y D_s)s(t)\big) \tag{2.9b}$$

$$y(t) = Cx(t) + D_u u(t) + D_s s(t) \tag{2.9c}$$

Note that, in case no structural constraints are imposed in the definition of hyperparameters and free parameters, the learned model may not retain the structural interconnections between subsystems, required for decentralized and distributed controller design. In particular, although the model may be accurate and dynamically consistent, modularity and scalability that characterize network-based structures may be absent.
In the following section we discuss how physical information can be enforced on the learning process of RENs.

2.4. Structured Identification of REN models

The centralized identification process described in Section 2.3 presents several challenges such as lack of modularity and scalability, high computational cost and dense network representation. As specified in Section A.1, it is possible to define decomposed models in a non-overlapping or partially-overlapping forms, suitable for decentralized and distributed control frameworks. In this section, different identification approaches inspired by available decomposition approaches are described.

In any case, the preliminary step relies on the definition of the input-output pairs (u_i, y_i), each representing the input/output channel of the i-th subsystem $\mathscr{S}_i$. In particular, in a physics-inspired modeling framework, the input-output i-th pair (u_i, y_i) is defined first by subdividing the plant $\mathscr{S}$ in subplants $\mathscr{S}_i$, for $i = 1, ..., N$, and then identifying

- the specific manipulable inputs u_i of the subplant $\mathscr{S}_i$
- the measured variables y_i of $\mathscr{S}_i$

Namely, the channel is identified in such a way that input u_i is selected (for structural

reasons) as the one defined by the i-th controller to regulate the output y_i in a decentralized/distributed architecture.

As a second step, the interconnections between the subsystems $\mathscr{S}_i$, with $i = 1, ..., N$, must be defined by physical inspection. This allows to define a physical graph $\mathscr{G}^0$, whose nodes are the subsystems, and whose edges represent the interconnections. More specifically, an edge (i, j) between $\mathscr{S}_j$ and $\mathscr{S}_i$ is present if some variables (e.g., the input u_j, the measured output y_j, or some elements of the internal non-measured state x_j) of $\mathscr{S}_j$ have a direct known physical influence on the dynamics of $\mathscr{S}_i$.
As an alternative representation, we can define, for each $i = 1, ..., N$, the set $\mathscr{N}_i^0$ as the set of indices j such that (i, j) is an edge of the graph $\mathscr{G}^0$ and which does not include i.

At this point it is possible to define the interconnected model. Three possible approaches can be defined, based on different assumptions and ideas.

1) Completely overlapping-inspired identification
This approach, previously adopted in [22], allows to obtain input-coupled and state-decoupled models similar to the ones obtained, in a model-based framework, from a completely overlapping decomposition approach.

First, the model is obtained by imposing the hyperparameters as follows:

1. Define n_x (dimension of internal states vector $x(t)$) and n_s (dimension of non-linearities vector $s(t)$).
2. Define $B_u, \tilde{B}_u, B_y, \tilde{B}_y$ as full random matrices.
3. Define B_s as a sparse matrix, meaning that most of its entries are zero, while a small fraction of elements are randomly-assigned nonzero values. This allows random nonlinear interactions to be included in the internal state vector $s(t)$, while keeping the model computationally manageable.
4. Define $\tilde{B}_s$ as a lower triangular matrix, as suggested in [29]. In this configuration, the elements of $s(t)$ can be computed explicitly *row-by-row*, avoiding algebraic loops. Compared to a general REN, the acyclic version is simpler to implement and often produces models of comparable quality.
5. Define A_x and $\tilde{A}_x$ as full matrices, subject to the contractivity constraint (2.6).

Secondly, the model identification procedure defined in Section 2.3 must be executed. Indeed, the identification of the full matrices C, D and D_s, should be carried out by minimizing the cost function given in Equation (2.8). Note that this does not explicitly enforce a structured sparsity pattern on the system matrices.

Therefore, the presence of state-dependent input couplings leads, for the learned model, to a possibly fully connected representation, represented by Equations (2.9a, 2.9b, 2.9c).

Consistently with the decomposition presented in Appendix A.1 and denoting with N the number of subsystems, the decomposed model is obtained by partitioning matrices $B_u, B_y, \tilde{B}_u$ and $\tilde{B}_y$ in N block-column matrices, D_s in N block-rows matrices, an D_u in $N \times N$ blocks. The obtained decomposed model is:

$$x(t+1) = A_x x(t) + \sum_{i=1}^{N} B_{u_i} u_i(t) + B_s s(t) + \sum_{i=1}^{N} B_{y_i} y_i(t) \tag{2.10a}$$

$$s(t) = \sigma\Big(\tilde{A} x(t) + \sum_{i=1}^{N} \tilde{B}_{u_i} u_i(t) + \tilde{B}_s s(t) + \sum_{i=1}^{N} \tilde{B}_{y_i} y_i(t)\Big) \tag{2.10b}$$

$$y_i(t) = C_i x(t) + \sum_{j=1}^{N} D_{u_{ij}} u_j(t) + D_s s_i(t) \tag{2.10c}$$

As previously specified, the edges of the interconnection graph $\mathscr{G}^0$ correspond to physical interactions among subsystems. In this case, however, the identified model is input-coupled meaning that the input coupling is enforced by introducing interaction terms at the input level, which results in an all-to-all connected interconnection graph, exactly how we obtained in model-based fully overlapping decompositions.

For this procedure, the validation step is performed by simulating predictions generated by the centralized model. Equations (2.9a), (2.9b), and (2.9c) are used to generate the predictions, that are then compared with the validation dataset and the selected model is the one, among the candidate models, that minimizes the cost function.

2) Output-coupling decomposition

This approach can be adopted when the physical couplings among subsystems occur through measurable inputs and outputs variables, in such a way that the model of each subsystem $\mathscr{S}_i$ can be written in the following general form:

$$x_i(t+1) = f\big(x_i(t), u_i(t), \{u_j(t), y_j(t)\}_{j \in \mathscr{N}_i^0 \cup \{i\}}\big) \tag{2.11}$$

Consistently with (2.11), the resulting untrained REN i-th model assumes the following

form:

$$x_i(t+1) = A_{x_i}x_i(t) + B_{u_{ii}}u_i(t) + B_{s_i}s_i(t) + B_{y_{ii}}y_i(t) + \sum_{j\in\mathcal{N}_i^0}\left(B_{y_{ij}}y_j(t) + B_{u_{ij}}u_j(t)\right) \tag{2.12a}$$

$$s_i(t) = \sigma\Big(\tilde{A}_{x_i}x_i(t) + \tilde{B}_{u_{ii}}u_i(t) + \tilde{B}_{s_i}s_i(t) + \tilde{B}_{y_{ii}}y_i(t) + \sum_{j\in\mathcal{N}_i^0}\left(\tilde{B}_{y_{ij}}y_j(t) + \tilde{B}_{u_{ij}}u_j(t)\right)\Big) \tag{2.12b}$$

$$y_i(t) = C_i x_i(t) + D_{u_i}u_i(t) + D_{s_i}s_i(t) \tag{2.12c}$$

In this case, for each subsystem $\mathscr{S}_i$, the identification procedure is performed using a composite input signal. Besides the local control input u_i, the composite input vector of (2.12a) and (2.12b) also includes the outputs and inputs of neighboring subsystems enter $\mathscr{S}_i$, consistently with the interconnection structure described by the graph $\mathscr{G}^0$.

Therefore, N separate identification procedures need to be performed, resulting in N distinct models. The training procedure follows the one described by Sections 2.2 and 2.3 where, for each subsystem, we perform an independent centralized identification.

First of all, the hyperparameters need to be defined as follows, for all $i = 1, \ \dots \ , N$:

1. Define $n_{x,i}$ (dimension of internal states vector $x_i(t)$) and $n_{s,i}$ (dimension of non-linearities vector $s_i(t)$).
2. Define matrices $B_{u,ii}, \tilde{B}_{u,ii}, B_{y,i}, \tilde{B}_{y,ii}, B_{s,i}$ and $\tilde{B}_{s,i}$, and $B_{u,ij}, \tilde{B}_{u,ij}, B_{y,ij}$ and $\tilde{B}_{y,ij}$, for $j \in \mathcal{N}_i^0$, as full random matrices.
3. Define $A_{x,i}$ and $\tilde{A}_{x,i}$ in such a way that contractivity constraint

$$\begin{bmatrix} \bar{\alpha}^2 P_i & -\tilde{A}_{x,i}^T\Lambda & A_{x,i}^T P_i \\ -\Lambda\tilde{A}_{x,i} & W_i & B_{s,i}^T P_i \\ P_i A_{x,i} & P_i B_{s,i} & P_i \end{bmatrix} > 0$$

   is enforced.

Secondly, as presented in Section 2.3, matrices C_i, D_i and D_{s_i} can be learned by minimizing the quadratic error between simulated data and the predicted output, according to Equation (2.8).

In this way, the learned submodels take the following structured form, for all $i = 1, ...N$

$$x_i(t+1) = (A_{x_i} + B_{y_{ii}}C_i)x_i(t) + (B_{u_{ii}} + B_{y_{ii}}D_i)u_i(t) + (B_{s_i} + B_{y_{ii}}D_{s_i})s_i(t) + \sum_{j\in\mathcal{N}_i^0} \left(B_{y_{ij}}C_j x_j(t) + (B_{u_{ij}} + B_{y_{ij}}D_j)u_j(t) + B_{y_{ij}}D_{s_j}s_j(t)\right) \tag{2.13a}$$

$$s_i(t) = \sigma\Big((\tilde{A}_{x_i} + \tilde{B}_{y_{ii}}C_i)x_i(t) + (\tilde{B}_{u_{ii}} + \tilde{B}_{y_{ii}}D_i)u_i(t) + (\tilde{B}_{s_i} + \tilde{B}_{y_{ii}}D_{s_i})s_i(t) + \sum_{j\in\mathcal{N}_i^0} \left(\tilde{B}_{y_{ij}}C_j x_j(t) + (\tilde{B}_{u_{ij}} + \tilde{B}_{y_{ij}}D_j)u_j(t) + \tilde{B}_{y_{ij}}D_{s_j}s_j(t)\right)\Big) \tag{2.13b}$$

For the validation of the learned models (where an ad-hoc validation dataset is commonly used), two approaches can be adopted, denoted as "validation using data" and "validation using predictions" below.

1. **Validation using data**

   As a first approach to perform the model validation, we can compute, in a decentralized fashion, the predictions $\hat{y}_i(t)$ of the outputs $y_i(t)$, for all $i = 1, ...N$, through the following equations. They are obtained from the output-coupled decomposition, but using (2.12c) only for the "local" subsystem, in (2.12a) and (2.12b), i.e.,

   $$\hat{x}_i(t+1) = (A_{x_i} + B_{y_{ii}}C_i)\hat{x}_i(t) + (B_{u_{ii}} + B_{y_{ii}}D_i)u_i(t) + (B_{s_i} + B_{y_{ii}}D_{s_i})\hat{s}_i(t) + \sum_{j\in\mathcal{N}_i^0} \left(B_{y_{ij}}y_j(t) + B_{u_{ij}}u_j(t)\right) \tag{2.14a}$$

   $$\hat{s}_i(t) = \sigma\Big((\tilde{A}_{x_i} + \tilde{B}_{y_{ii}}C_i)\hat{x}_i(t) + (\tilde{B}_{u_{ii}} + \tilde{B}_{y_{ii}}D_i)u_i(t) + (\tilde{B}_{s_i} + \tilde{B}_{y_{ii}}D_{s_i})\hat{s}_i(t) + \sum_{j\in\mathcal{N}_i^0} \left(\tilde{B}_{y_{ij}}y_j(t) + \tilde{B}_{u_{ij}}u_j(t)\right)\Big) \tag{2.14b}$$

   $$\hat{y}_i(t) = C_i\hat{x}_i(t) + D_{u_i}u_i(t) + D_{s_i}\hat{s}_i(t) \tag{2.14c}$$

   Note that, in this approach, each submodel uses the validation data sequences $u_i(t), \{u_j(t), y_j(t)\}_{j\in\mathcal{N}_i^0}$ to compute $\hat{y}_i(t)$, for all $i = 1, ..., N$.

2. **Validation using predictions**

   The second validation approach, which is more specifically tailored to validate the system-wide capability of the obtained identified model to represent the system dynamics, consists of obtaining the predictions $\hat{y}_i(t)$ of the outputs $y_i(t)$, for all

$i = 1, ...N$, from Equations (2.13a) and (2.13b), i.e.,

$$\hat{x}_i(t+1) = (A_{x_i} + B_{y_{ii}}C_i)\hat{x}_i(t) + (B_{u_{ii}} + B_{y_{ii}}D_i)u_i(t) + (B_{s_i} + B_{y_{ii}}D_{s_i})\hat{s}_i(t) + \sum_{j\in\mathcal{N}_i^0} \left(B_{y_{ij}}\hat{y}_j(t) + B_{u_{ij}}u_j(t)\right) \tag{2.15a}$$

$$\hat{s}_i(t) = \sigma\Big((\tilde{A}_{x_i} + \tilde{B}_{y_{ii}}C_i)\hat{x}_i(t) + (\tilde{B}_{u_{ii}} + \tilde{B}_{y_{ii}}D_i)u_i(t) + (\tilde{B}_{s_i} + \tilde{B}_{y_{ii}}D_{s_i})\hat{s}_i(t) + \sum_{j\in\mathcal{N}_i^0} \left(\tilde{B}_{y_{ij}}\hat{y}_j(t) + \tilde{B}_{u_{ij}}u_j(t)\right)\Big) \tag{2.15b}$$

$$\hat{y}_i(t) = C_i\hat{x}_i(t) + D_{u_i}u_i(t) + D_{s_i}\hat{s}_i(t) \tag{2.15c}$$

Note that, in this approach, each submodel uses the validation data sequances $u_i(t), \{u_j(t), \hat{y}_j(t)\}_{j\in\mathcal{N}_i^0}$ to compute $\hat{y}_i(t)$, for all $i = 1, ..., N$.

In both cases, to evaluate the model quality, the following RMSE is used, i.e.,

$$\text{RMSE} = \frac{1}{N}\sum_{i=1}^{N}\sqrt{\frac{1}{N_s - N_\text{w}}\sum_{t=N_\text{w}+1}^{N_s}\left(y_i(t) - \hat{y}_i(t)\right)^2} \tag{2.16}$$

where $\hat{y}_i(t)$ can be computed either with (2.14c) or with (2.15c), depending on the used validation approach, and where N_w is the so-called washout period, i.e., the time interval that is discarded, as it is required for the system to settle from the initial condition.

Some remarks are in order at this point.
The validation approach 1 is the most computationally lightweight since it can be performed in a purely decentralized fashion (i.e., system-by-system) but, at the same time, it is closer to the identification approach, and it is meant to evaluate the precision with which the single submodel i reacts to external stimuli, i.e. $u_i(t)$ and $\{u_j(t), y_j(t)\}$ for all $j \in \mathcal{N}_i^0$.
On the other hand, the evaluation performed through the validation approach 2 aims to test the system-wide representation capabilities of the overall model, which is the main and most significant goal of the identification procedure.

In light of this, the validation approach 2 will be considered as the most relevant one. Finally note that, in view of the discussion made in Section 1.4 about the impact of the sampling time, one of the main reasons of the possible mismatch between results obtained using the two validation approaches lies in the choice of the sampling time. Notably, in fact, Validation from Data assumes that the model (2.14a), (2.14b), (2.14c) is fed by sample-and-hold signals $u_i(t), \{u_j(t), y_j(t)\}_{j\in\mathcal{N}_i^0}$, but $y_j(t)$ are outputs which are not

constant during the sampling intervals, leading to significant approximation when T_s is large.

3) Structured matrix decomposition

The core idea of this approach is to perform the identification described in Sections 2.2 and 2.3, while imposing a particular structure on the model matrices (i.e., both the hyperparameters and free ones) to reflect the interconnections between systems. In this way, the influence of each subsystem on its neighbors is naturally represented within the model, without the need to explicitly introduce coupling variables, making this approach applicable even in case the interconnection variables are not measurable, contrarily with the "Output-coupling decomposition" presented above.

To include the desired interconnection structure to the model through the matrix sparsity, in this work we propose, first to define the hyperparameters as follows.

1. Define n_x (dimension of internal states vector $x(t)$) and n_s (dimension of non-linearities vector $s(t)$). At this point, it is also necessary to define n_{x_i} and n_{s_i}, for $i = 1, ..., N$, as they represent the dimensions of the blocks of the structured matrices, introduced in the following points.
2. $B_u, \tilde{B}_u, B_y, \tilde{B}_y, B_s$ and $\tilde{B}_s$ are defined as block diagonal matrices, composed of blocks $B_{u_i} \in \mathbb{R}^{n_{x_i} \times n_{u_i}}$, $\tilde{B}_{u_i} \in \mathbb{R}^{n_{s_i} \times n_{u_i}}$, $B_{y_i} \in \mathbb{R}^{n_{x_i} \times n_{y_i}}$, $\tilde{B}_{u_i} \in \mathbb{R}^{n_{s_i} \times n_{y_i}}$. $B_{s_i} \in \mathbb{R}^{n_{x_i} \times n_{s_i}}$ has sparse random blocks and $\tilde{B}_{s_i} \in \mathbb{R}^{n_{s_i} \times n_{s_i}}$ has random lower triangular blocks.
3. A_x and $\tilde{A}_x$ as block diagonal matrices, with blocks $A_{x_i} \in \mathbb{R}^{n_{x_i} \times n_{x_i}}$ and $\tilde{A}_{x_i} \in \mathbb{R}^{n_{s_i} \times n_{x_i}}$, respectively, generated randomly, but subjected to the contractivity constraint (2.6).

A given structure is imposed also on the free parameters C, D and D_s, so as to reflect the interconnections represented by the graph $\mathscr{G}^0$. All matrices must have suitable dimensions and must coherent with each other and with overall system's interconnections.

The resulting untrained REN i-th submodel assumes the following form:

$$x_i(t+1) = A_{x_i} x_i(t) + B_{u_i} u_i(t) + B_{s_i} s_i(t) + B_{y_i} y_i(t) \tag{2.17a}$$

$$s_i(t) = \sigma\big(\tilde{A}_{x_i} x_i(t) + \tilde{B}_{u_i} u_i(t) + \tilde{B}_{s_i} s_i(t) + \tilde{B}_{y_i} y_i(t)\big) \tag{2.17b}$$

$$y_i(t) = C_i x_i(t) + D_{u_i} u_i(t) + D_{s_i} s_i(t) + \sum_{j \in \mathcal{N}_i^0} \big(C_{ij} x_j(t) + D_{u_{ij}} u_j(t)\big) \tag{2.17c}$$

In this structured identification approach for interconnected systems, the overall state dimension of the identified model is therefore given by the aggregation of the state vari-

ables associated with each subsystem, rather than by the minimal order of the global input–output behavior.

The identification of C, D and D_s is carried out by minimizing the cost function given in equation (2.8).
Considering the trained model equations derived by using (2.17c) into (2.17a) and (2.17b), the structure of all trained matrices is dictated by that of C, D, and D_s. This is a crucial result, as the tunable matrices C, D, and D_s encode the interconnection relationships within the system. Importantly, the admissible interaction structure is informed by the underlying physics of the system, while the level of interconnections is identified directly from data rather than being arbitrarily imposed as hyperparameter. As a consequence, the trained model inherits a structured representation that is both physics-consistent and data-driven, with matrices that preserve the same interconnection pattern as $\mathscr{G}^0$, ensuring a faithful representation of the system's physical and structural properties.

In this procedure, the validation step is performed using the model obtained by plugging (2.17c) into (2.17a) and (2.17b). The generated data are then compared with the validation dataset, and the resulting model is the one, among the candidate ones, that minimizes the cost function.
After training, we define matrices $A_i = A_{x_i} + B_{y_i}C_i, B_i = B_{u_i} + B_{y_i}D_{u_i}, B^0_{s_i} = B_{s_i} + B_{y_i}D_{u_i}, A_{ij} = B_{y_i}C_i, B_{ij} = B_{y_i}D_{u_{ij}}, \tilde{A}_i = \tilde{A}_{x_i} + \tilde{B}_{y_i}C_i, \tilde{B}_i = \tilde{B}_{u_i} + \tilde{B}_{y_i}D_{u_i}, \tilde{B}^0_{s_i} = \tilde{B}_{s_i} + \tilde{B}_{y_i}D_{u_i}, \tilde{A}_{ij} = \tilde{B}_{y_i}C_i, \tilde{B}_{ij} = \tilde{B}_{y_i}D_{u_{ij}}$.
In this way, the learned i-th submodel takes the form

$$x_i(t+1) = A_i x_i(t) + B_i u_i(t) + B^0_{s_i} s_i(t) + \sum_{i \in \mathcal{N}_i^0} \big(A_{ij} x_j(t) + B_{ij} u_j(t)\big) \tag{2.18a}$$

$$s_i(t) = \sigma\Big(\tilde{A}_i x_i(t) + \tilde{B}_i u_i(t) + \tilde{B}^0_{s_i} s_i(t) + \sum_{i \in \mathcal{N}_i^0} \big(\tilde{A}_{ij} x_j(t) + \tilde{B}_{ij} u_j(t)\big)\Big) \tag{2.18b}$$

$$y_i(t) = C_i x_i(t) + D_i u_i(t) + D_{s_i} s_i(t) + \sum_{j \in \mathcal{N}_i^0} \big(C_{ij} x_j(t) + D_{ij} u_j(t)\big) \tag{2.18c}$$

3 | Application of the learning approaches to the case study

3.1. Case study: Chemical Plant

The proposed methodologies are validated through simulation tests on a benchmark chemical plant proposed in [33], demonstrating their feasibility and effectiveness in real-world applications. This configuration provides an ideal environment for testing the proposed methodologies and for evaluating the trade-offs between model complexity, physical consistency, and control performance. In fact, the plant consists of three interconnected subsystems, including a recycle line, making it particularly suitable for analyzing interdependencies and the effect of information sharing among subsystems. Additionally, the presence of strong nonlinearities makes it suitable to be modeled using Recurrent Equilibrium Networks.

Since distributed and decentralized control architectures are widely adopted in the petrochemical industry for the control of large-scale, multi-variable processes, this case study provides a good benchmark for testing effectiveness of developed concepts and methodologies.

The considered framework is a chemical plant consisting of two reactors and a separator, illustrated in Figure 3.1 and in detail described in [33].

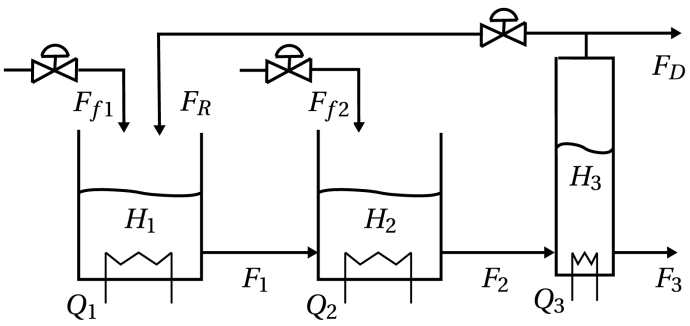


Figure 3.1: Two reactors in series with separator and recycle.

A stream of pure reactant A enters the two reactors (subsystems 1 and 2), and converted into the desired product B. The product is lost by a reaction to side product C. The latter passes to the separator (subsystem 3), whose distillate is split between the downstream process and a recycle stream directed back to the first reactor. For each subsystem $i \in \{1, 2, 3\}$ the state is composed of the level H_i, the temperature T_i, and the concentrations of the three components x_{Ai}, x_{Bi}, and x_{Ci}, while the flow rates between the vessels are denoted as F_i.
The system's dynamics can be described by the following nonlinear differential equations:

$$\begin{cases} \frac{dH_1}{dt} &= \frac{1}{\rho A_1}\left(F_{f1} + F_R - F_1\right) \\ \frac{dx_{A1}}{dt} &= \frac{1}{\rho A_1 H_1}\left(F_{f1}x_{A0} + F_R x_{AR} - F_1 x_{A1}\right) - k_{A1}x_{A1} \\ \frac{dx_{B1}}{dt} &= \frac{1}{\rho A_1 H_1}\left(F_R x_{BR} - F_1 x_{B1}\right) + k_{A1}x_{A1} - k_{B1}x_{B1} \\ \frac{dT_1}{dt} &= \frac{1}{\rho A_1 H_1}\left(F_{f1}T_0 + F_R T_R - F_1 T_1\right) - \frac{1}{C_p}\left(k_{A1}x_{A1}\Delta H_A + k_{B1}x_{B1}\Delta H_B\right) + \frac{Q_1}{\rho A_1 C_p H_1} \\ \frac{dH_2}{dt} &= \frac{1}{\rho A_2}\left(F_{f2} + F_1 - F_2\right) \\ \frac{dx_{A2}}{dt} &= \frac{1}{\rho A_2 H_2}\left(F_{f2}x_{A0} + F_1 x_{A1} - F_2 x_{A2}\right) - k_{A2}x_{A2} \\ \frac{dx_{B2}}{dt} &= \frac{1}{\rho A_2 H_2}\left(F_1 x_{B1} - F_2 x_{B2}\right) + k_{A2}x_{A2} - k_{B2}x_{B2} \\ \frac{dT_2}{dt} &= \frac{1}{\rho A_2 H_2}\left(F_{f2}T_0 + F_1 T_1 - F_2 T_2\right) - \frac{1}{C_p}\left(k_{A2}x_{A2}\Delta H_A + k_{B2}x_{B2}\Delta H_B\right) + \frac{Q_2}{\rho A_2 C_p H_2} \\ \frac{dH_3}{dt} &= \frac{1}{\rho A_3}\left(F_2 - F_D - F_R - F_3\right) \\ \frac{dx_{A3}}{dt} &= \frac{1}{\rho A_3 H_3}\left(F_2 x_{A2} - (F_D + F_R)x_{AR} - F_3 x_{A3}\right) \\ \frac{dx_{B3}}{dt} &= \frac{1}{\rho A_3 H_3}\left(F_2 x_{B2} - (F_D + F_R)x_{BR} - F_3 x_{B3}\right) \\ \frac{dT_3}{dt} &= \frac{1}{\rho A_3 H_3}\left(F_2 T_2 - (F_D + F_R)T_R - F_3 T_3\right) + \frac{Q_3}{\rho A_3 C_p H_3} \end{cases} \tag{3.1}$$

where, for all $i \in \mathbb{I}_{1:3}$:

$$F_i = k_{vi}H_i, \qquad k_{Ai} = k_A \exp\left(-\frac{E_A}{RT_i}\right), \qquad k_{Bi} = k_B \exp\left(-\frac{E_B}{RT_i}\right) \tag{3.2}$$

The recycle flow and weight rates satisfy:

$$F_D = 0.01F_R, \quad x_{AR} = \frac{\alpha_A x_{A3}}{\bar{x}_3}, \quad x_{BR} = \frac{\alpha_B x_{B3}}{\bar{x}_3},$$

$$\bar{x}_3 = \alpha_A x_{A3} + \alpha_B x_{B3} + \alpha_C x_{C3}, \quad x_{C3} = 1 - x_{A3} - x_{B3}.$$

The system parameters are provided in Tables 3.1

Parameter	A_1	A_2	A_3	ρ	C_p	k_{v1}	k_{v2}	k_{v3}	x_{a0}	T_0
Value	3	3	1	0.15	25	2.5	2.5	2.5	1	313
Units	m^2	m^2	m^2	kg/m^3	kJ/kgK	kg/ms	kg/ms	kg/ms	$wt_{\%}$	K

Parameter	k_a	k_b	E_A/R	E_B/R	ΔH_A	ΔH_B	α_A	α_B	α_C
Value	0.02	0.018	-100	-150	-40	-50	3.5	1.1	0.5
Units	1/s	1/s	K	K	kJ/kg	kJ/kg	—	—	—

Table 3.1: Chemical Plant System's Parameters

The output and input are denoted, respectively, by

$$y = \begin{bmatrix} H_1 & x_{A1} & x_{B1} & T_1 & H_2 & x_{A2} & x_{B2} & T_2 & H_3 & x_{A3} & x_{B3} & T_3 \end{bmatrix}^T$$
$$u = \begin{bmatrix} F_{f1} & Q_1 & F_{f2} & Q_2 & F_R & Q_3 \end{bmatrix}^T.$$

Flow rates and heatings in input have the saturation limits represented in Table 3.2:

Parameter	F_{f1}	Q_1	F_{f2}	Q_2	F_R	Q_3
Lower Bound	0	0	0	0	0	0
Upper Bound	10	50	10	50	75	50
Units	kg/s	kJ/s	kg/s	kJ/s	kg/s	kJ/s

Table 3.2: Chemical Plant Input constraints

Furthermore, the states are assumed to be measurable, then they coincide with the outputs. As inputs, also all states are constrained to be positive and, in particular, for concentrations we have to $x_{Ai} \leq 1$ and $x_{Bi} \leq 1$, by definition. The state vector does not include the concentration of the product denoted as C, since it is constrained by the other

two by:

$$\begin{cases} x_{Ai} + x_{Bi} + x_{Ci} = 1 \\ x_{Ai} + x_{Bi} < 1 \end{cases} \tag{3.3}$$

The system model is implemented in MATLAB and subsequently simulated in Simulink. The constrained optimization problem implementing the control laws defined in the following chapters, is solved using the YALMIP-Mosek toolbox, which is well-suited for handling both linear and nonlinear optimization problems.

3.2. Data acquisition and pre-processing

Before performing model identification, data are acquired from the simulator and pre-processed to ensure both consistency with physical constraints and effective training of the neural network.
Specifically, the training, validation, and test datasets are generated by simulating the system (3.1) in Simulink. The system is excited with a Multilevel Pseudo-Random Signal (MPRS) to effectively cover different operating regions of the system. MPRS with both high and low frequency components are used: at high frequency, input changes occur every 2 seconds whereas, at low frequency, they occur every 10 seconds. The input profiles used for the test are shown in Figure 3.2.

For the considered chemical plant, it is crucial to excite the system with inputs within the physical limits reported in Table 3.2. For each input combination, also output profiles must remain within specified bounds and respect the physical limitations on the concentration of fluids A, B and C, as expressed in (3.3). Figures 3.2, 3.3 and 3.4 show clearly that all imposed physical limits are satisfied by inputs and outputs profiles. As suggested by [33] and based on the considerations presented in Section 1.4, 2.4, and A.2, the selected sampling time in the simulation is $T_s = 0.1s$.

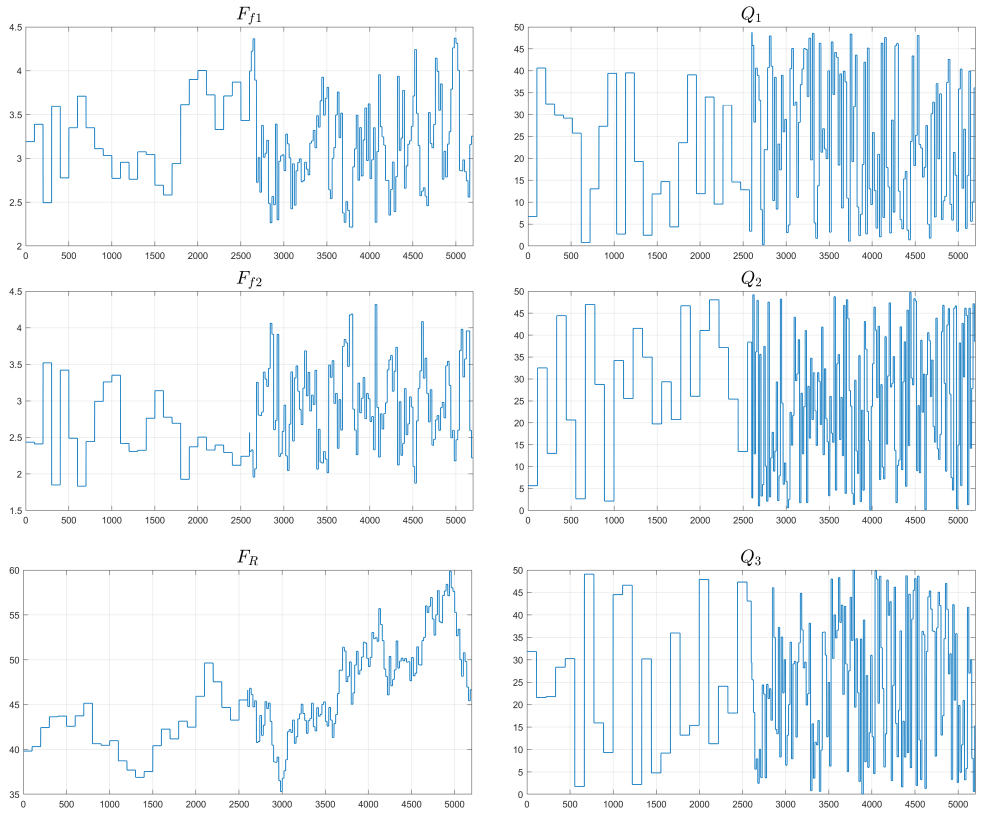


Figure 3.2: Plots of simulated inputs

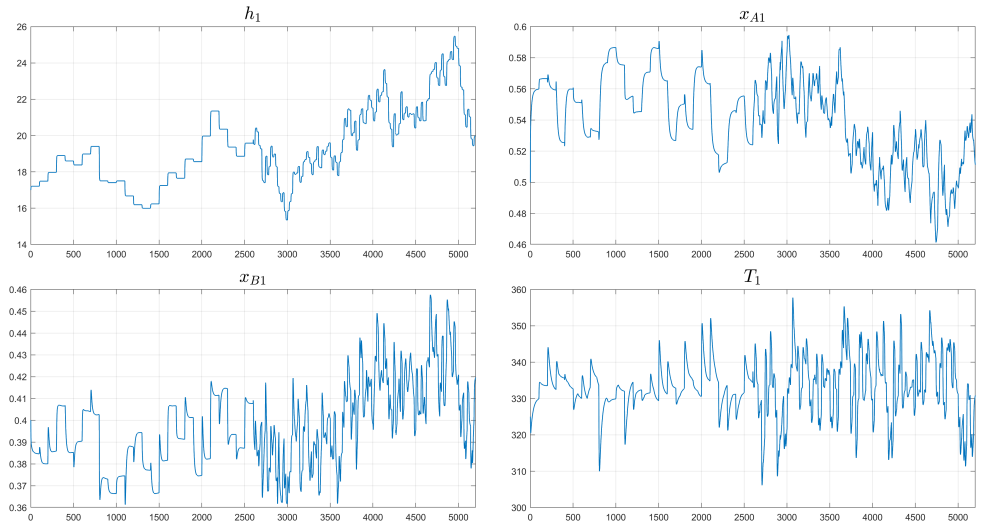


Figure 3.3: Plots of non-scaled simulated outputs

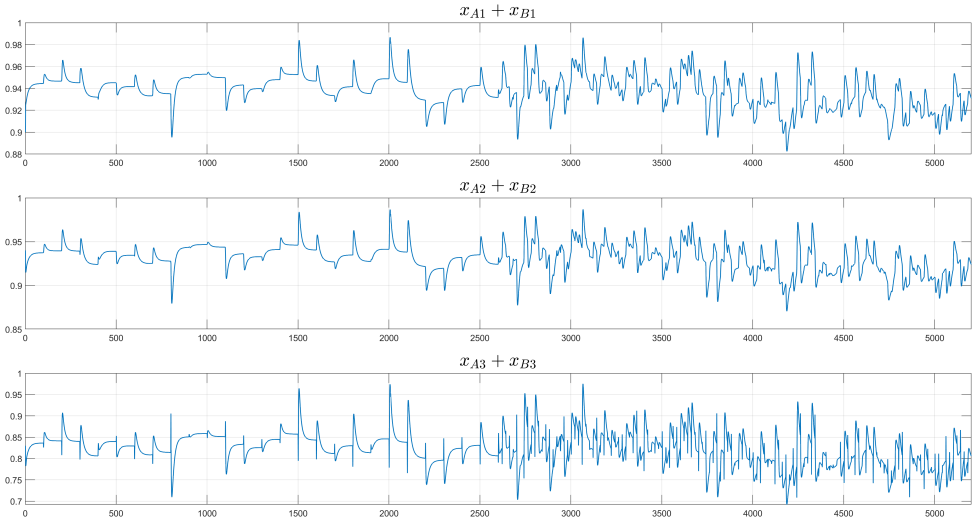


Figure 3.4: Plots of concentrations' sum for each subsystem

Acquired data are divided as follows: approximately 45% of data are used for the training set, $\mathcal{I}_t$, 30% for the validation set $\mathcal{I}_v$ and 25% for the test set $\mathcal{I}_f$. In the training set, about 65% of datapoints correspond to low frequency excitations and 35% to high frequency excitations. Low-frequency data are prioritized because they capture the dominant, slower dynamics of the system, which are essential for accurate modeling and reliable long-term predictions. High-frequency data, on the other hand, represent fast transients, which are less influential on the overall system behavior but still necessary to accurately capture the system's rapid dynamics.
Data acquisition is followed by data pre-processing: data are normalized in order to ensure consistency with the system constraints. Indeed, in this systems, features have very different scales (for example, concentrations range between 0 and 1, while temperatures assumes values around 300K). Normalization guarantees balanced gradients during training, improves numerical stability, and supports the REN in capturing the couplings between variables. Input and output data are scaled as follows:

$$y^{(i)} \in [0,1], \text{ for } i = 1, ..., n_y \qquad u^{(i)} \in [0,1], \text{ for } i = 1, ..., n_u.$$

where $y^{(i)}$ and $u^{(i)}$ represent the i-th entries of vectors y and u, respectively.

3.3. Non structured REN model

In this section we show the results of the identification procedure of a non structured (i.e., centralized) REN model, following the procedure described in Section 2.4.

The resulting model quality depends on random hyperparameters: in order to explore a large set of candidates, the identification procedure consists of 1000 iterations. At each iteration, a different candidate model is generated by randomly selecting the model dimensions n_x and n_s and by randomly initializing the fixed hyperparameters. These quantities are then kept fixed during the identification of that model. Each iteration produces a distinct identified model, which is evaluated on a validation dataset. The performance of each candidate model is quantified by the Root Mean Square Error (RMSE), used to measure the discrepancy between the model outputs and the simulated outputs. The best model is selected as the one achieving the minimum RMSE on the validation set.

In this case:

1. $n_x = 12$ and $n_s = 1$ give the best model among the candidates. Different tests (data not shown for brevity) have been conducted for $n_x \in [7, 12]$ and $n_s \in [1, n_x - 3]$.
2. $B_u, \tilde{B}_u, B_y, \tilde{B}_y$, B_s and $\tilde{B}_s$ are matrices of appropriate dimensions randomly gener-

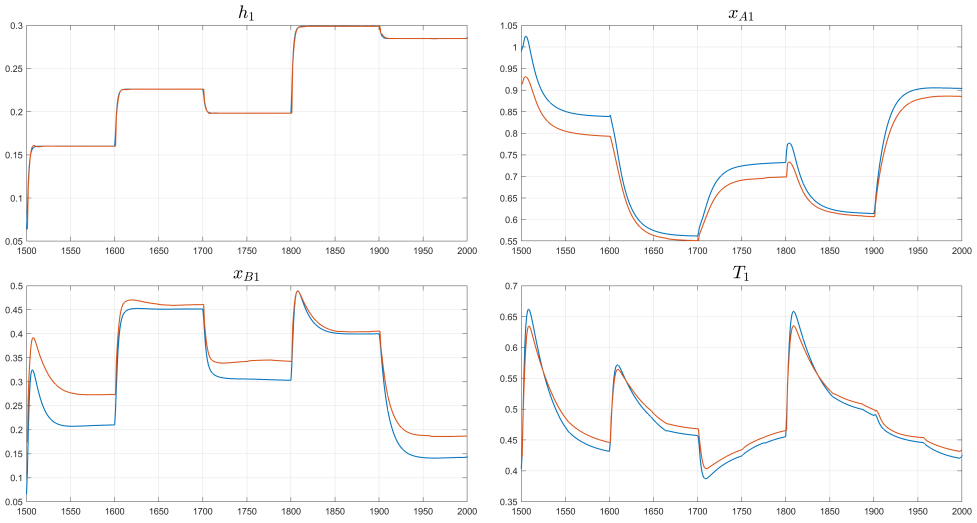
ated and then kept fixed throughout the identification process. Their dimensions are chosen to be consistent with those determined in the previous step, and match the model structure described in Equations (2.1a), (2.1b), and (2.1c).

3. The matrices A_x and $\tilde{A}_x$ are also fixed during the identification, but they are obtained by solving a constrained feasibility problem rather than being drawn directly from a random distribution. In particular, they are generated by enforcing the contractivity and positive-trace constraints described in Chapter 2.2.

4. The identification of the full matrices C, D and D_s is carried out by minimizing the cost function given in Equation (2.8).

The quantitative evaluation of the model performances will be carried out in Section 3.5, showing RMSE and $\text{FIT}_{[\%]}$.

Figures 3.5 and 3.6 represent the centralized REN model outputs, compared to the simulated data of the Chemical Plant, with a focus on both low and high frequency input signals.



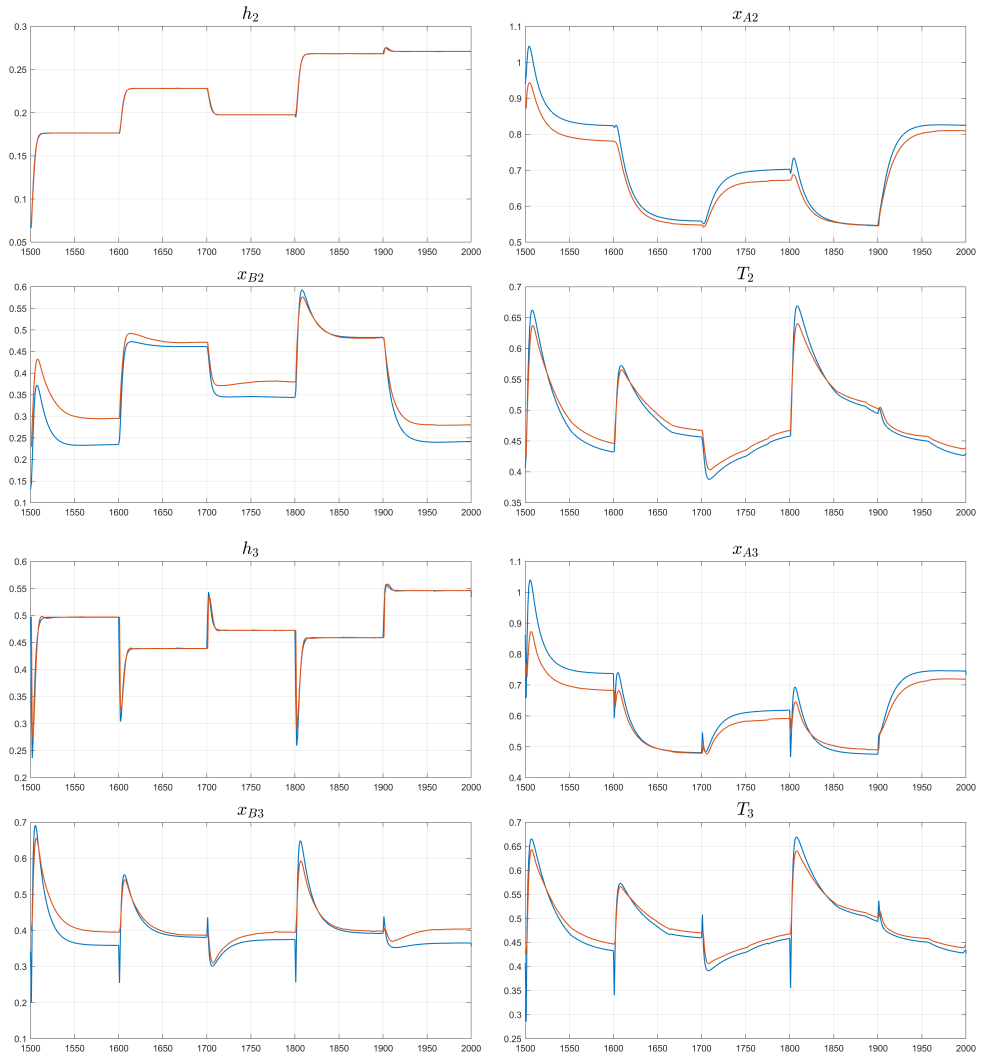


Figure 3.5: Comparison between the simulated output (——) and the REN model output (——), under low-frequency input signals.

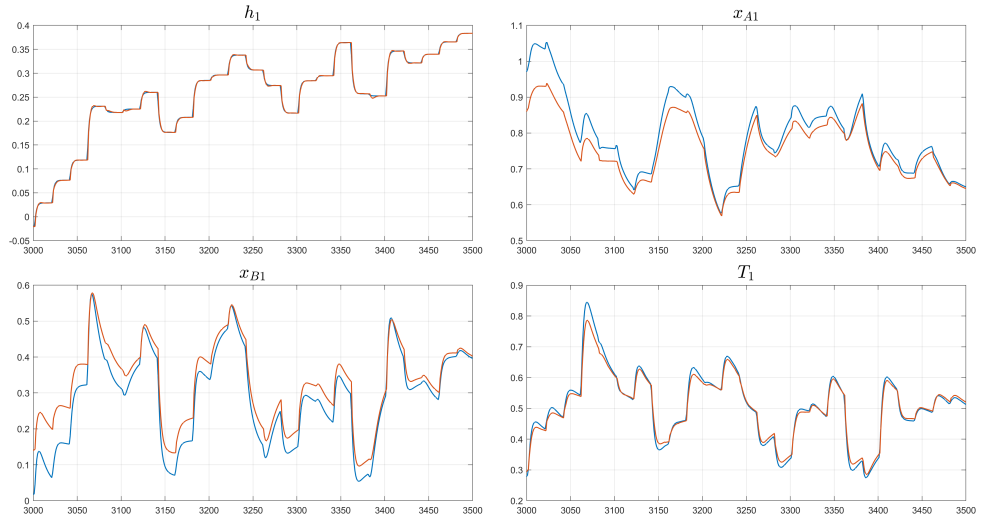


Figure 3.6: Comparison between the simulated output (——) and the REN model output (——), under high-frequency input signals.

The quality of this model will be evaluated in terms of RMSE and $\text{FIT}_{[\%]}$ of each single output and the overall model. It is important to highlight that this model is a MIMO unstructured one, and therefore all inputs, internal states, and non-linearities influence all output profiles. Figures 3.5 and 3.6 show that the model output profiles closely match the simulated ones, showing only slight deviations in states most affected by nonlinearity, such as concentrations, while preserving consistency in response times and steady-state values.

3.4. Structured REN models

In this section we describe the results achieved by applying, to the benchmark data, the two approaches, among the ones presented in Sections 2.4, which lead to a modular structured model with a sparse interconnection graph. In particular, in Section 3.4.1 and 3.4.2 the Output-coupling decomposition and the Structured matrix decomposition, respectively, approaches, are used.

3.4.1. Output-coupling Identification

In this section we describe the results obtained by applying the Output-coupling decomposition identification procedure presented in Section 2.4. As in Section 3.3, the chosen hyperparameters are the ones giving the best matching with the data outputs, while imposed properties reflect the physics of the system and improve the model qualities in terms of RMSE and $\text{FIT}_{[\%]}$. In particular, also in this case, 1000 iterations are performed in order to find the best model among many possible candidates.

For $i = 1, 2, 3$, the following setup is finally obtained, with reference to model (2.12a), (2.12b) and (2.12c).

1. We set $n_{x,1} = n_{x,2} = n_{x,3} = 20$ and $n_{s,1} = 3, n_{s,2} = 1, n_{s,3} = 17$. The selected values give the best model among the candidates, i.e., we tested $n_{x,i} \in [12, 24]$ and $n_{s,i} \in [1, n_x - 3]$, for all $i = 1, ..., 3$.

2. The matrices $B_{u,i}, \tilde{B}_{u,i}, B_{y,i}, \tilde{B}_{y,i}, B_{s,i}$ and $\tilde{B}_{s,i}$ are randomly generated and then kept fixed throughout the identification process. Their dimensions must be consistent with those determined in the previous step and described in Chapter 2.1.

3. The matrices $A_{x,i}$ and $\tilde{A}_{x,i}$ are also fixed during identification, but they are obtained by solving a constrained feasibility problem rather than being directly imposed from a distribution. Indeed, they are generated by a feasibility problem enforcing the contractivity and positive-trace constraints described in Chapter 2.2.

4. The identification of the matrices C_i, D_i and $D_{s,i}$, is carried out by minimizing the cost function given in Equation (2.8).

As specified in Section 2.4, this identification procedure essentially corresponds to three separated identification procedures. The couplings between subsystems are imposed by the input and output vectors. In particular the input vector of the i-th subsystem is composed by its physical inputs u_i, and outputs generated by neighboring subsystems y_j. For the chemical plant:

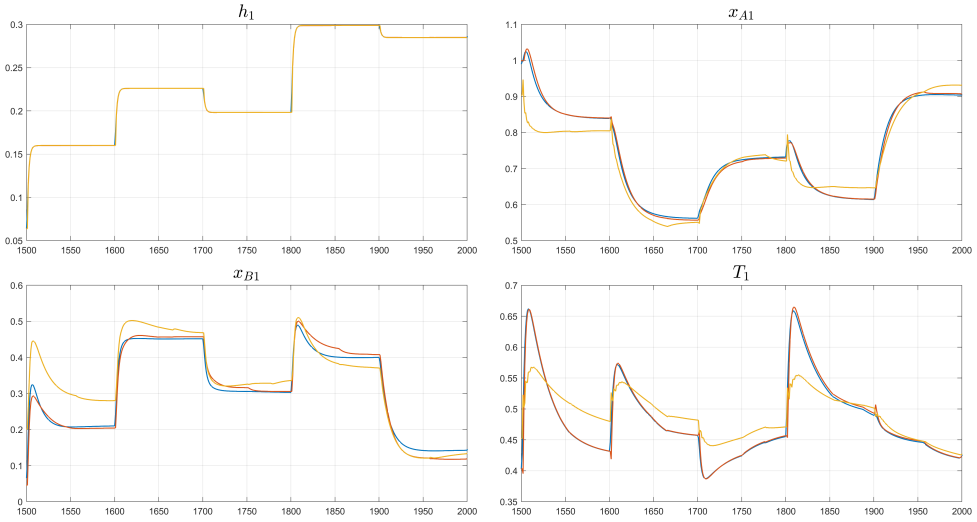
- **Reactor 1:** inputs are $u_1 = [F_{f1}, Q_1]^T$ and $y_3 = [h_3, x_{A3}, x_{B3}, T_3]^T$ and outputs are $y_1 = [h_1, x_{A1}, x_{B1}, T_1]^T$
- **Reactor 2:** inputs are $u_2 = [F_{f2}, Q_2]^T$ and $y_1 = [h_1, x_{A1}, x_{B1}, T_1]^T$ and outputs are $y_2 = [h_2, x_{A2}, x_{B2}, T_2]^T$
- **Separator:** inputs are $u_3 = [F_R, Q_3]^T$ and $y_2 = [h_2, x_{A2}, x_{B2}, T_2]^T$ and outputs are $y_3 = [h_3, x_{A3}, x_{B3}, T_3]^T$

It is important to specify that in each iteration, three sub-models are computed, i.e., one for each subsystem. The overall performance is then evaluated by combining all three sub-models, with the RMSE calculated on the aggregated output rather than separately for each subsystem.

As specified in Section 2.4, two different validation approaches can be employed: "validation from data" and "validation from predictions", leading to different RMSE costs, $\text{RMSE}_{\text{DATA}}$ and $\text{RMSE}_{\text{PREDICTIONS}}$, respectively.

Figures 3.7 and 3.8 represent the REN model outputs computed with two described approaches.

As discussed, the second validation approach relies on full model-generated data, resulting in a more realistic and physically consistent validation procedure. Although the first approach may achieve better RMSE and $\text{FIT}_{[\%]}$ metrics, it may overestimate the model accuracy and does not ensure reliable behavior under real operating conditions.



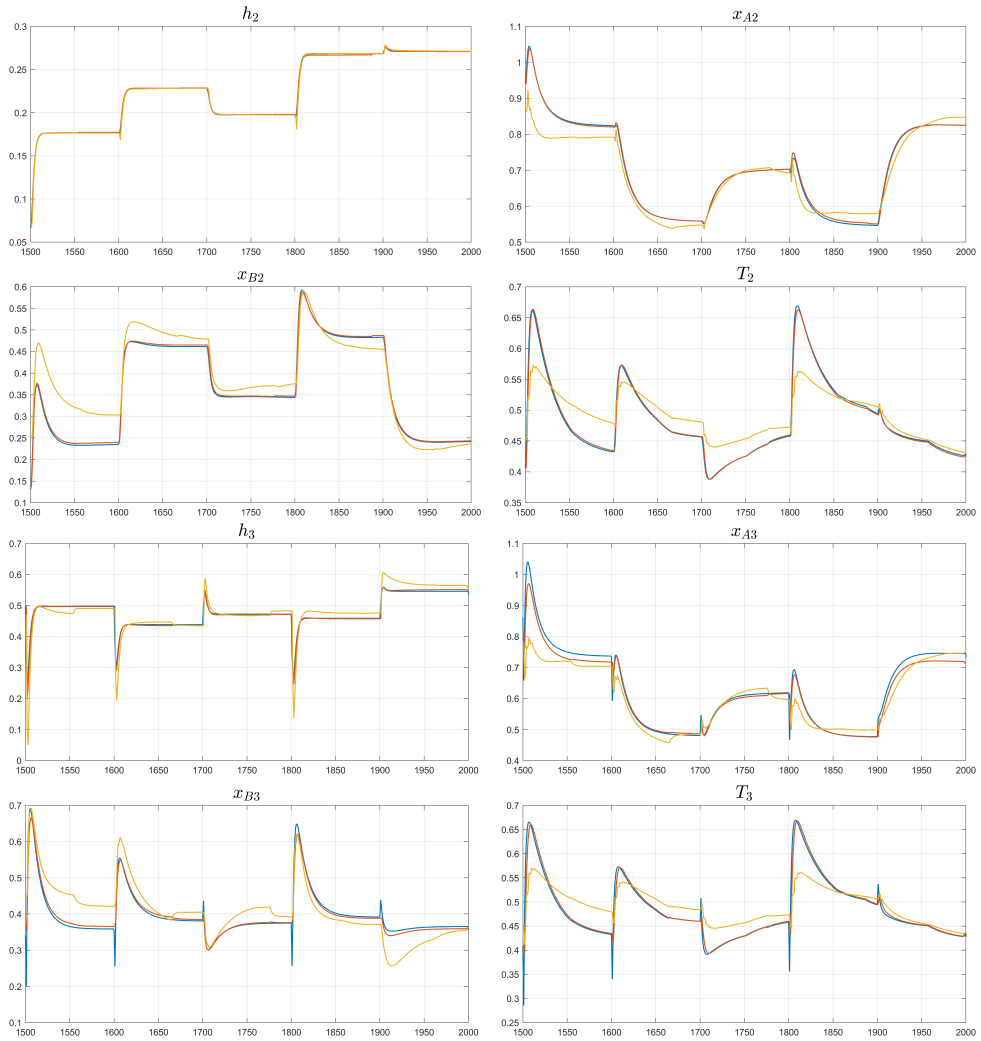


Figure 3.7: Comparison between the simulated output (——), the REN model output (validation from simulated data) (——), and the REN model output (validation from predictions) (——), under low-frequency input signals.

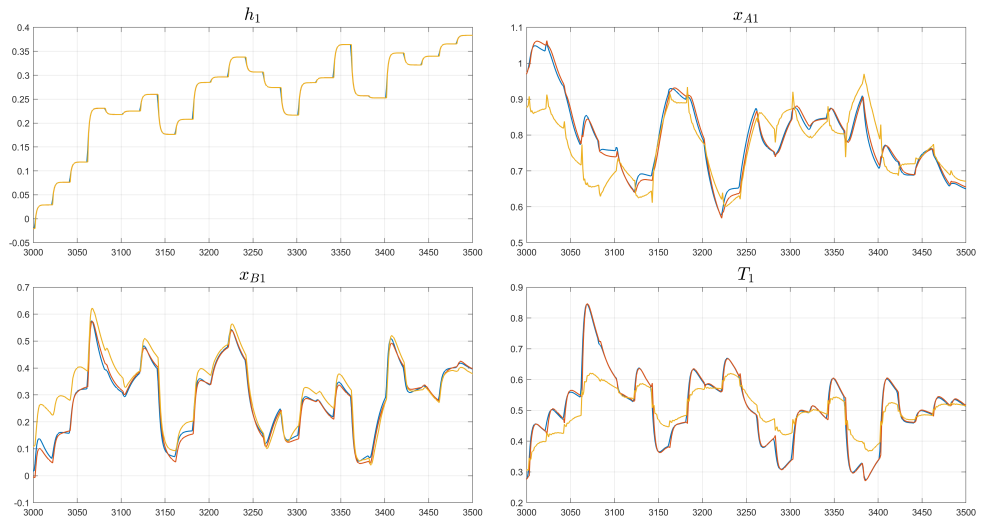
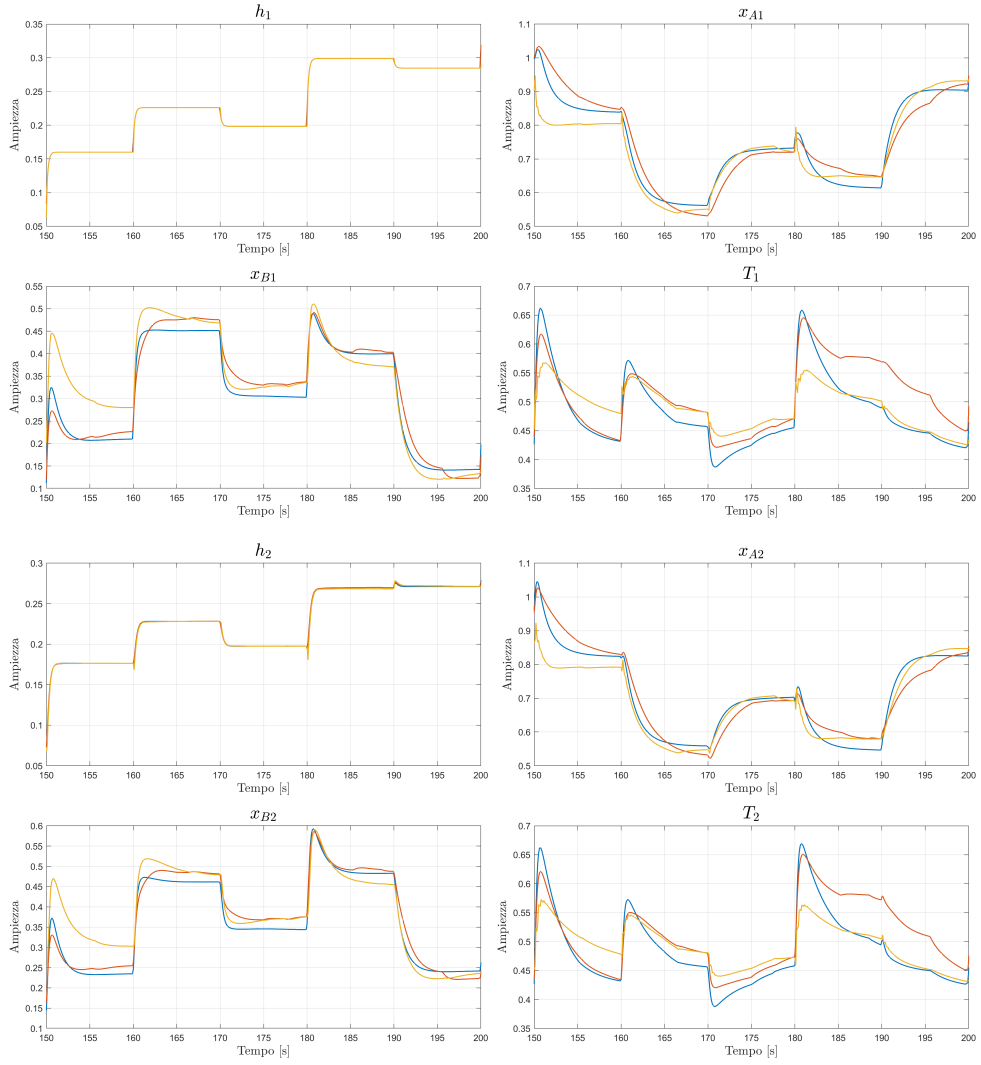


Figure 3.8: Comparison between the simulated output (——), the REN model output (validation from simulated data) (——), and the REN model output (validation from predictions) (——), under high-frequency input signals.

Note that the REN model outputs obtained from simulated data are almost coincident with the data outputs. In contrast, the REN model outputs obtained from predictions provide satisfactory results only when nonlinear effects are marginal. Although the model shows reduced accuracy in tracking the desired trajectories, particularly in nonlinear cases, it still captures the dominant system dynamics, as highlited by the correct steady-state values and settling times. The quantitative evaluation of the model will be carried out in Section 3.5.

As specified in Section 2.4, the quality deterioration of the model obtained from predictions is partially due to the choice of a large sampling time and arises, among other things, as a consequence of the discretization.

Figures 3.9 and 3.10 show the comparison of the same identification procedure, performed with two different datasets. The first model is the same as that used in the previous section (obtained with data sampled at $T_s = 0.1s$), while the second one is obtained using the same identification procedure, but with a sampling time reduced by a factor of 10 ($T_s = 0.01s$). In order to make a fair comparison between the two models, the same system order is imposed. Notably, the dataset sampled with $T_s = 0.01s$ leads to a more accurate model, as clear from Figures 3.9 and 3.10.



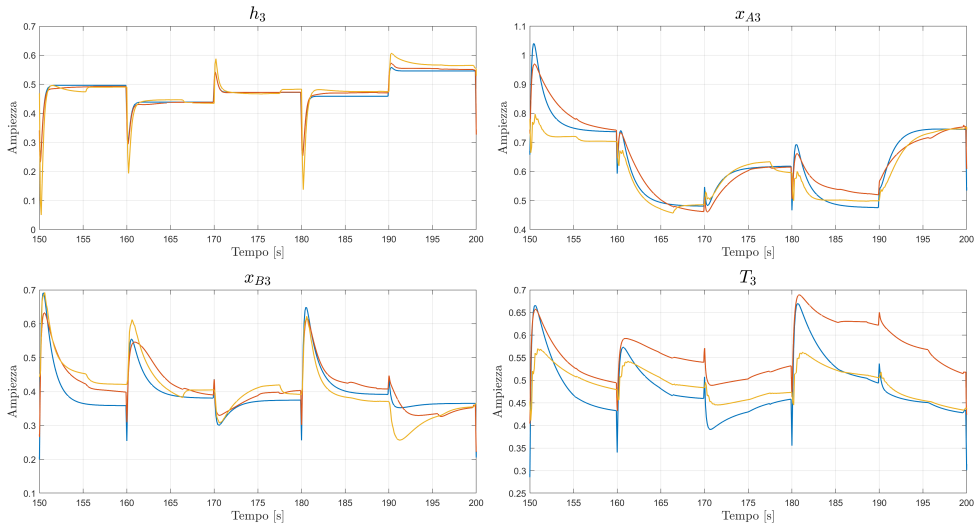


Figure 3.9: Comparison between the simulated output (——), and the REN model output with $T_s = 0.01s$ (——) and with $T_s = 0.1s$ (——), under low-frequency input signals.

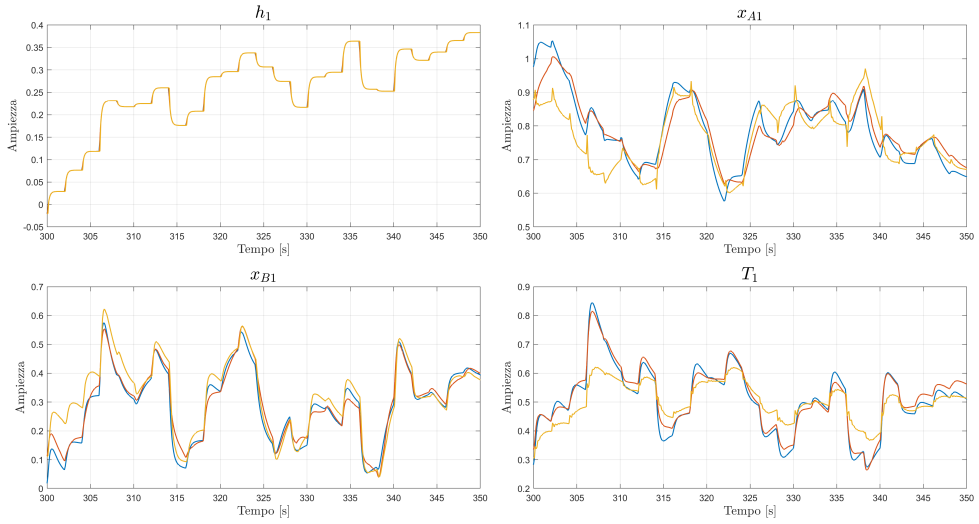


Figure 3.10: Comparison between the simulated output (——), and the REN model output with $T_s = 0.01s$ (——) and with $T_s = 0.1s$ (——), under high-frequency input signals.

From a qualitative analysis of the output profiles, it is possible to observe that a smaller sampling time produces two main improvements. First, steady-state values are reached more accurately and, secondly, the settling times more closely match those of the reference data. Moreover, oscillations arising from the discretization approximation are reduced and, in some cases, become negligible.

These results and observations are confirmed by the quantitative analysis reported in Table 3.3, where the RMSE (2.16), the overall $\text{FIT}_{[\%]}$ (3.4) and the $\text{FIT}_{[\%]}$ (3.5) for each individual output are evaluated. This analysis shows that reducing the sampling time improves the model quality not only from a qualitative perspective, but also in terms of quantitative performance metrics.

Table 3.3: Comparison between models obtained from Individual Subsystem Identification (with validation using predicted outputs) for different sampling times

	Model with $T_s = 0.1$ s	**Model with $T_s = 0.01$ s**
$\text{FIT}_{[\%]}$ h_1	98.46	99.98
$\text{FIT}_{[\%]}$ x_{A1}	70.22	76.04
$\text{FIT}_{[\%]}$ x_{B1}	75.62	85.07
$\text{FIT}_{[\%]}$ T_1	51.65	71.41
$\text{FIT}_{[\%]}$ h_2	98.25	99.69
$\text{FIT}_{[\%]}$ x_{A2}	71.64	77.06
$\text{FIT}_{[\%]}$ x_{B2}	75.75	86.73
$\text{FIT}_{[\%]}$ T_2	53.32	72.31
$\text{FIT}_{[\%]}$ h_3	38.38	92.13
$\text{FIT}_{[\%]}$ x_{A3}	61.65	77.90
$\text{FIT}_{[\%]}$ x_{B3}	53.97	66.12
$\text{FIT}_{[\%]}$ T_3	47.02	71.16
$\text{FIT}_{[\%]}$	**76.51**	**86.41**
RMSE	**0.0455**	**0.0267**

3.4.2. Structured Matrices Identification

In this section we describe the results obtained by applying the procedure for structured identification presented in Section 2.4.

As in Sections 3.3 and 3.4.1, the resulting model is chosen among a set of candidates explored in 1000 iterations. In each iteration, hyperparameters (dimensions and matrices explained in points 1., 2., and 3.) are randomly generated and kept fixed. In each iteration the overall model is computed, validated and tested, then it is selected as the one that corresponds to the smallest RMSE.

1. The dimension of internal state vector is chosen in the set $n_x \in [50, 60]$, resulting in $n_x = 57$, while the dimension of non-linearities vector is chosen in $n_s \in [20, n_x - 1]$, resulting in $n_s = 22$. It is necessary to define also $n_{x,i}$ and $n_{s,i}$, for $i = 1, 2, 3$, in order to define the dimensions of blocks inside model matrices. The resulting block dimensions are $n_{x,1} = 16$, $n_{x,2} = 18$, $n_{x,3} = 23$ and $n_{s,1} = 8$, $n_{s,2} = 7$, $n_{s,3} = 7$.

2. $B_u, \tilde{B}_u, B_y, \tilde{B}_y, B_s, \tilde{B}_s$ are block diagonal matrices, with blocks having dimensions consistent with n_{x_i}, and n_{s_i}, for all $i = 1, 2, 3$ defined in previous point. They are randomly generated and then kept fixed during each single iteration of the identification procedure.

3. Matrices A_x and $\tilde{A}_x$ are defined as block diagonal matrices, having dimensions of blocks consistent with n_{x_i}, and n_{s_i}, for all $i = 1, 2, 3$, defined in step 1. They are the result of a feasibility problem, constrained to contractivity and positive traces, as specified in Chapter 2.2.

4. The output matrices C, D and D_s are the result of the optimization problem that minimizes the cost function (2.8).
   As discussed, the specific structure imposed to C, D and D_s reflect the interconnections represented by the graph $\mathscr{G}^0$. In the selected case study the graph is a cycle and the system interconnections are represented in Figure 3.11.

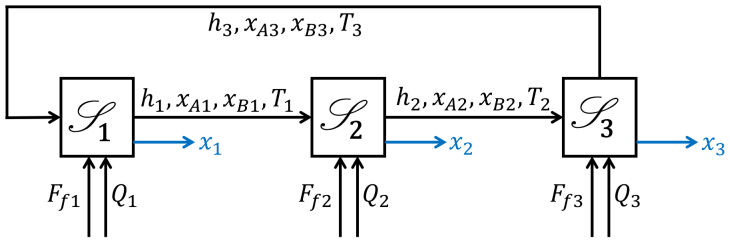


Figure 3.11: Chemical plant interconnection structure

The structure of C, D and D_s is represented in Figure 3.12. Blue squares represent non-zero blocks, while zeros terms represent the absence of interconnections between subsystems i and j. In view of this, the trained model preserves the physical structure of the system, which is reflected in the state-space matrices. Notably, the level of interconnections is not enforced as a hyperparameter, but is instead identified directly by during the training process.

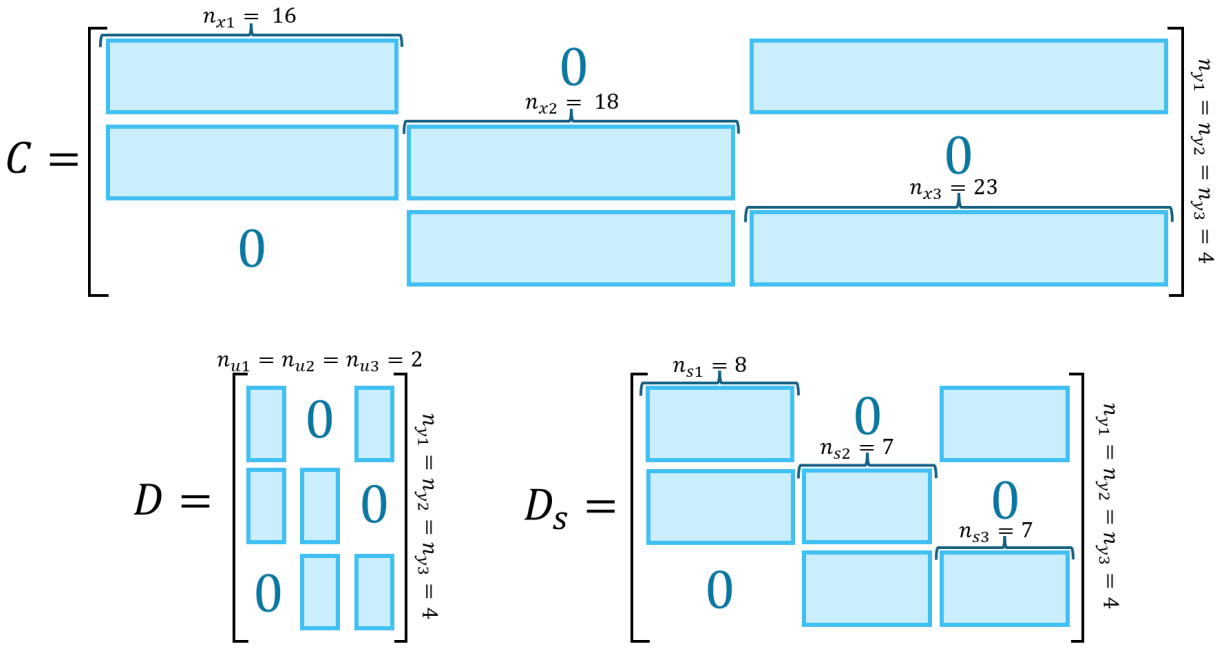
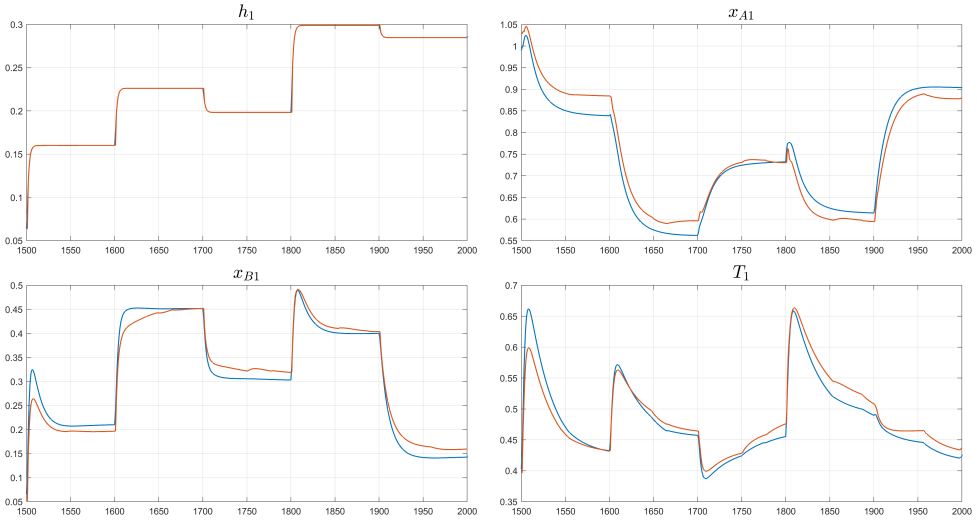


Figure 3.12: Chemical plant output matrices structure

Figures 3.13 and 3.14 represent the REN model outputs computed with structured matrices, compared to simulated data of the Chemical Plant, with a focus on low and high frequency input signals.



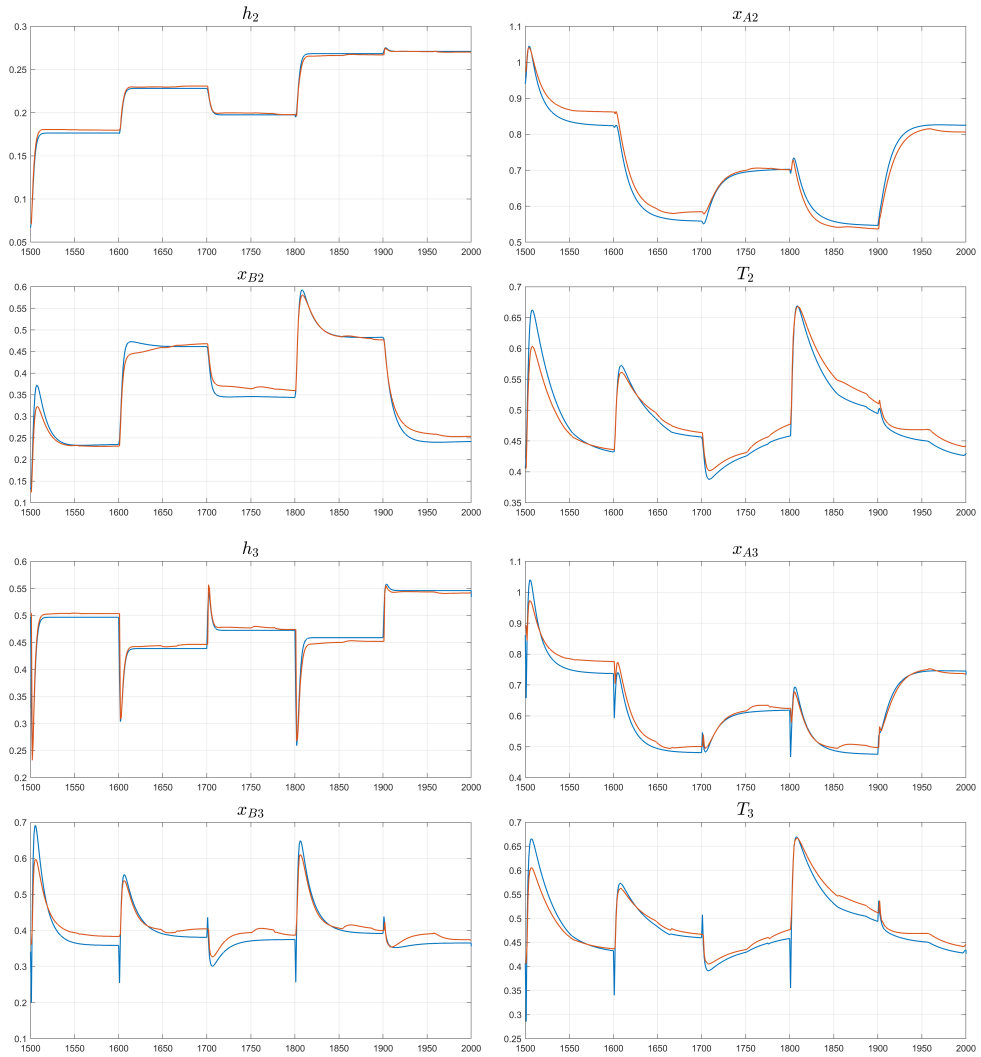
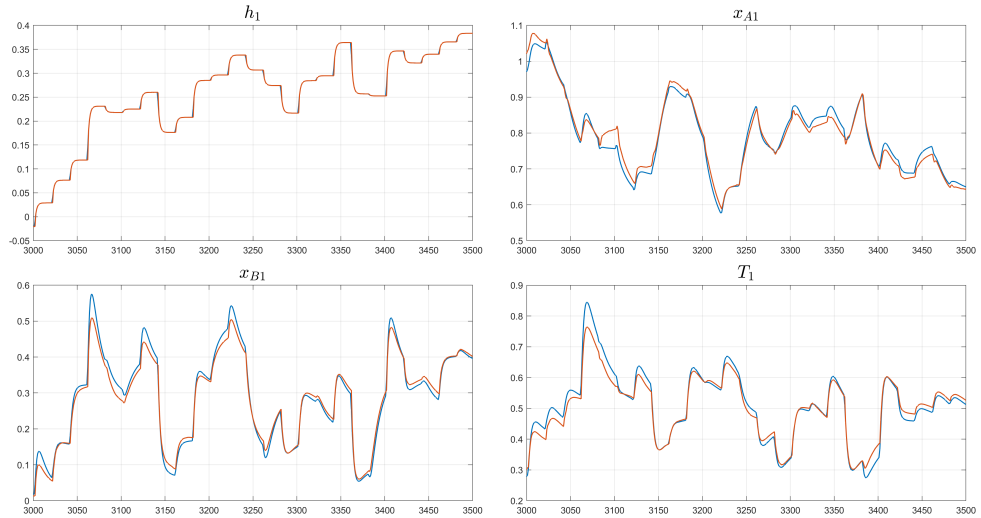


Figure 3.13: Comparison between the simulated output (——), the REN model output (structured matrices) (——), under low-frequency input signals.



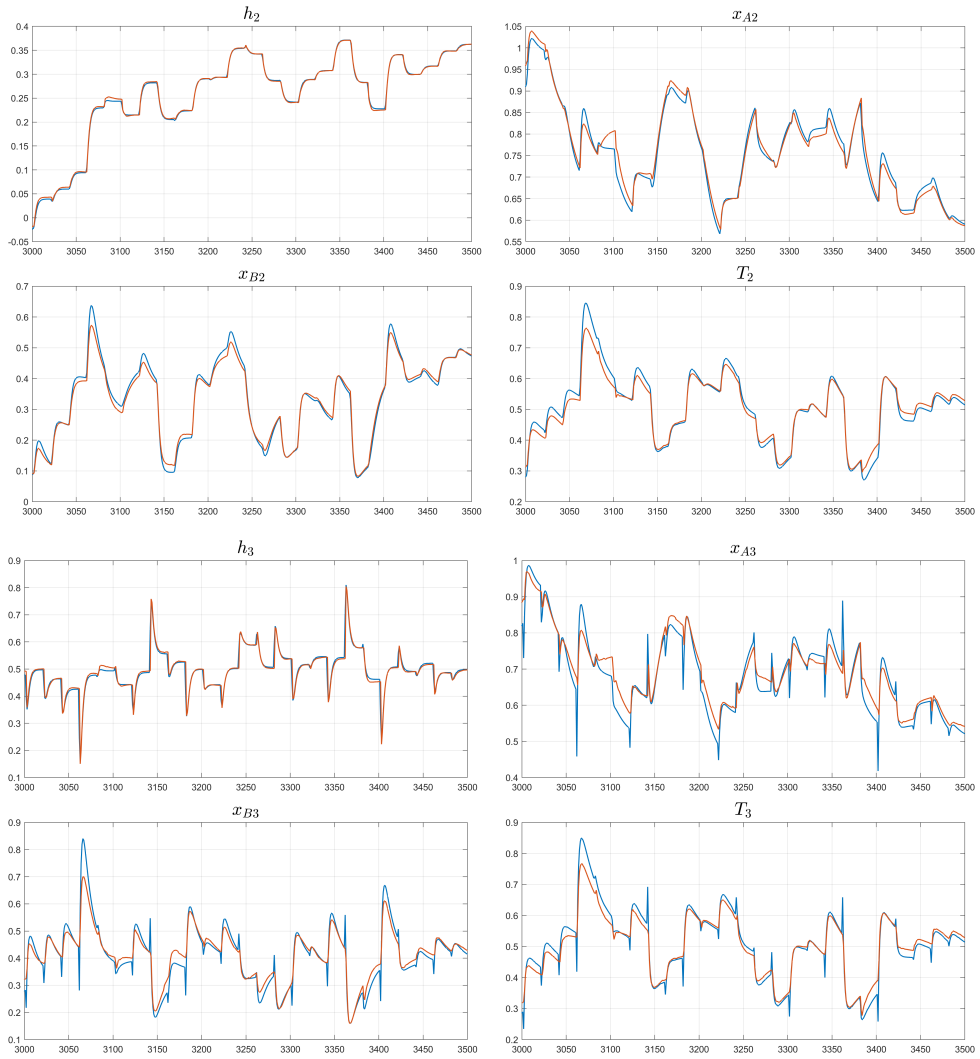


Figure 3.14: Comparison between the simulated output (——), the REN model output (structured matrices) (——), under high-frequency input signals.

The Structured Identification results in a model that faithfully reflects the simulated outputs: steady-state values are reached correctly and the sampling time matches that of the system. This correspondence demonstrates that the dominant dynamics are properly captured, resulting in matching profiles even in cases where non-linearities are strongly involved, such as concentrations outputs. Moreover, as specified in Section 3.4.1, non-relevant oscillations in the output profiles can also be observed in this case, partially due to the effects of the discretization approximation errors.

3.5. Comparative analysis of the models

This section presents a comparison between the identified models considering both quantitative and qualitative aspects. A quantitative analysis is provided in Table 3.4, that shows

- Root Mean Square Error (RMSE), computed as in Equation (2.16)
- FIT [%] of the overall model, i.e.,

$$\text{FIT}_{[\%]} = 100 \left(1 - \frac{\sqrt{\sum_{i=1}^{N} \sum_{t=N_\text{w}}^{N_s} (y_i(t) - \hat{y}_i(t))^2}}{\sqrt{\sum_{i=1}^{N} \sum_{t=N_\text{w}}^{N_s} (y_i(t) - \bar{y}_i)^2}} \right) \tag{3.4}$$

- FIT [%] of the i-th output

$$\text{FIT}_{[\%]} = 100 \left(1 - \frac{\sqrt{\sum_{t=N_\text{w}}^{N_s} (y_i(t) - \hat{y}_i(t))^2}}{\sqrt{\sum_{t=N_\text{w}}^{N_s} (y_i(t) - \bar{y}_i(t))^2}} \right) \tag{3.5}$$

where:

- $y_i(t)$ represents the i-th measured output at time t
- $\hat{y}_i$ represents the i-th model output at time t
- $\bar{y}_i = \dfrac{1}{N_s - \text{w} + 1} \displaystyle\sum_{t=\text{w}}^{N_s} y_i$ is the mean value of the measured data
- N_s is the total number of samples
- $N_\text{w} = 100$ is the washout period. It is applied to discard the transient response and assess the model accuracy only after the system reached steady-state behavior.

From Table 3.4, the results provided by the validation from data related to the Output-coupling identified models (Section 3.4.1) correspond to the largest total $\text{FIT}_{[\%]}$ and the smallest RMSE. However, as discussed, the most significant validation approach is the one that relies on Predictions (denoted as ValSim in Table 3.4); the latter shows that the

results obtained with the Output-coupling approach are not optimal. On the other hand, the best fitting and RMSE values are observed with the Structured Matrix Identification. Smaller values of $\text{FIT}_{[\%]}$ and worse responses are generally observed in the third subsystem and in $\text{concentration}_{[\%]}$ outputs x_A and x_B. This behavior can be explained by the higher degree of nonlinearity associated with these outputs, and by the fact that they are strongly influenced by several nonlinear interactions, as in the case of Subsystem 3.

As a conclusion, if the Structured Model represents the most appropriate choice, because:

- it reflects the structural relationships between the subsystems and the level of interconnection is identified, instead of being apriori fixed as hyperparameter,
- it does not require the knowledge of interconnection variables,
- it has modularity and scalability properties.

Table 3.4: Comparison between models: Centralized, Individual (with validation using output data and validation using simulated outputs), and Structured

	Centralized Model	**Output-coupled Model**		**Structured Model**
		Val-Sim	**Val-Data**	
$\text{FIT}_{[\%]}$ h_1	98.36	98.46	98.44	98.46
$\text{FIT}_{[\%]}$ x_{A1}	84.05	70.22	93.56	86.63
$\text{FIT}_{[\%]}$ x_{B1}	77.01	75.62	92.15	89.94
$\text{FIT}_{[\%]}$ T_1	89.03	51.65	98.85	85.03
$\text{FIT}_{[\%]}$ h_2	99.26	98.25	98.82	98.69
$\text{FIT}_{[\%]}$ x_{A2}	84.78	71.64	96.27	87.64
$\text{FIT}_{[\%]}$ x_{B2}	78.27	75.75	96.79	91.37
$\text{FIT}_{[\%]}$ T_2	89.00	53.32	93.66	85.72
$\text{FIT}_{[\%]}$ h_3	69.35	38.38	68.49	69.80
$\text{FIT}_{[\%]}$ x_{A3}	75.53	61.65	83.86	80.38
$\text{FIT}_{[\%]}$ x_{B3}	68.95	53.97	81.00	73.10
$\text{FIT}_{[\%]}$ T_3	84.03	47.02	80.79	81.84
$\text{FIT}_{[\%]}$	**85.52**	**76.51**	**92.86**	**90.75**
RMSE	**0.0242**	**0.0455**	**0.0133**	**0.0160**

A centralized model is generally expected to achieve better $\text{FIT}_{[\%]}$ and RMSE due to its higher modeling flexibility. However, the results obtained in this work are consistent and can be explained by the considerations discussed in the following points.

- The structure imposed on the system matrices allows indirect correlations between inputs and outputs to be reduced, resulting in a more accurate learning of the dominant dynamics. In other words, the structured model is less flexible but more consistent with the physical behavior of the system. Contrarily, the centralized model is more flexible, but it may also capture non-physical interactions.
- In the centralized model, the level of interconnection among subsystems is partially implicitly fixed by the learning algorithm through the choice of hyperparameters. In the structured model, instead, the interconnections among subsystems—reflected in the trained system matrices—are entirely defined by the prescribed structure of the matrices C, D and D_s, and are therefore identified. The only imposed constraints correspond to the zero entries in the output matrices, which arise from the physics-inspired formulation of the model and are consistent with the system structure.
- The optimization of the cost function differs between the considered cases. For the optimization problems described in Sections 3.3 and 3.4, a fixed number of iterations is required to select hyperparameters that satisfy the LMIs. Using the same number of iterations, the centralized model is characterized by a larger parameter space, resulting in in a higher number of possible local minima and making the search for suitable hyperparameters more challenging. In contrast, the reduced number of degrees of freedom in the structured model allows to identify more easily a suitable combination of hyperparameters, leading to more reliable convergence.
- The total order of the systems differs between the centralized and structured cases. This may lead to improved performance in terms of $\text{FIT}_{[\%]}$ and RMSE.

Part II

Decentralized control design for Structured Recurrent Equilibrium Network models

4 | Centralized Robust Nonlinear Model Predictive Control

In the second part of this thesis a non-cooperative decentralized predictive control (DPC) algorithm, inspired by [13, 30], is devised, leveraging the modular model structure identified based on the guidelines provided in Part I. This control algorithm, consistently with the linear counterparts proposed in [13, 30], is based on a robust model predictive control formulation.

Robust NMPC is an extension of nonlinear MPC (i.e., NMPC) able to handle model uncertainties. Unlike traditional MPC, the so-called tube-based approach [20] ensures robustness with respect to bounded disturbances and uncertain dynamics, using the constraint tightening concept. The main source of uncertainties commonly include external disturbances and modeling errors, and they lead to sub-optimal performance or, in some cases, even instability.

The DPC algorithm described in this part relies on tube-based robust MPC, first proposed in [23] and later extended to the nonlinear context, e.g., in [35]. In this framework, N local controllers are designed in order to be robust with respect to coupling term related to neighboring systems.

In this chapter a short overview on tube-based robust MPC applied to a general nonlinear system is given, in order to anticipate some definitions, concepts and requirements used in the subsequent chapters. First, in Section 4.1, the problem is stated. Then, it is necessary to introduce the concept of incremental input-to-state stability, a major property to be imposed through a suitable controller, in Section 4.2. Finally, in the subsequent sections, the ingredients of the tube-based controller design approach will be described.

4.1. Statement of the problem

A discrete-time system affected by disturbance w can be expressed as:

$$\begin{cases} x(t+1) = f(x(t), u(t), w(t)) \\ y(t) = g(x(t), u(t)) \end{cases} \tag{4.1}$$

where:

- $x(t) \in \mathbb{R}^{n_x}$ is the state vector
- $u(t) \in \mathbb{R}^{n_u}$ is the input vector
- $y(t) \in \mathbb{R}^{n_y}$ is the output vector
- $w(t) \in \mathbb{R}^{n_x}$ is the disturbance
- $f(\cdot) = [f_1 \ ... \ f_{n_x}]^T$ is a vector of nonlinear functions

Before describing the controller design, we introduce a set of common assumptions used in this framework.

Assumption 4.1.1. $w(t) \in \mathbb{W} \ \forall t \geq 0$, i.e., the unknown disturbance $w(t)$ is bounded within a predefined set $\mathbb{W}$.

In order to guarantee that the disturbance remains inside within known limits, ensure the well-posedness of the problem, and to simplify the analysis of stability, control and robustness of the system (4.1), the set $\mathbb{W}$ must be:

- **Convex**: $\forall w_1(t), w_2(t) \in \mathbb{W}, \ \forall \lambda \in [0,1] \Rightarrow \lambda w_1(t) + (1-\lambda) w_2(t) \in \mathbb{W}, \ \forall t \geq 0$, i.e., any possible convex combinations of two disturbances in $\mathbb{W}$ also lies in $\mathbb{W}$.
- **Compact**: the set $\mathbb{W}$ is bounded and closed.

Assumption 4.1.2. The state, input, output vectors are constrained, i.e., $x(t) \in \mathbb{X}$ and $u(t) \in \mathbb{U}$. Sets $\mathbb{X}$ and $\mathbb{U}$ are compact and convex.

Sets $\mathbb{X}$ and $\mathbb{U}$ define admissible regions for states and inputs, respectively.
The control objective is to track a piecewise output constant reference $\bar{y} \in \mathbb{R}^{n_y}$, while mantaining trajectories consistent with the constraints specified in Assumption 4.1.2.

4.2. Incremental input-to-state stability

In many applications, guaranteeing weak notions of stability (i.e., stability of some equilibria) for non linear systems is often insufficient when the objective is to achieve robustness and to ensure that the system trajectories remain bounded even in presence of initial conditions variations and external disturbances.

On the other hand, the definition of δISS introduces a strong and robust notion of stability guaranteeing, among other things, boundedness of the distance between any two solutions, as a function of initial conditions distance and disturbance distance. Also, δISS guarantees that any trajectory of the system converge toward each other, at least when the system is not perturbed.
Consider the following definitions that allow us to formalize the δISS property.

Definition 4.2.1. [4] A continuous function $\gamma : [0, a) \to [0, \infty)$, with $a > 0$, is said to belong to class $\mathcal{K}$ if it is strictly increasing and $\gamma(0) = 0$.

■

Definition 4.2.2. [4] A continuous function $\gamma : [0, a) \to [0, \infty)$ is said to belong to class $\mathcal{K}_\infty$ if it is a class $\mathcal{K}$ function and if $a = \infty$ and $\gamma(r) \to \infty$ as $r \to \infty$.

■

Definition 4.2.3. [4] A continuous function $\beta : [0, a) \times [0, \infty) \to [0, \infty)$ is said to belong to class $\mathcal{KL}$ if, for each fixed s, the mapping $\beta(r, s)$ belongs to class $\mathcal{K}$ with respect to r, and, for each r, the mapping $\beta(r, s)$ is decreasing with respect to s, and $\beta(s, r) \to 0$, as $s \to 0$.

■

Definition 4.2.4. [11] An autonomous system

$$x(t + 1) = f_A(x(t), w(t)) \tag{4.2}$$

perturbed with a disturbance $w(t)$, is said to be δISS with respect to $w(t)$ if there exists a function $\beta \in \mathcal{KL}$ and a function $\gamma \in \mathcal{K}_\infty$ such that, for any time instant $t \geq 0$, for each pair of initial states $x_1(0), x_2(0)$ and each pair of disturbance sequences $\vec{w}_1, \vec{w}_2$, (where, for $i = 1, 2$ $\vec{w}_i = \{w_i(0), w_i(1), ...\}$) it holds that

$$||x(t, x_1(0), \vec{w}_1) - x(t, x_2(0), \vec{w}_2)|| \leq \beta(||x_1(0) - x_2(0)||, t) + \gamma(||\vec{w}_1 - \vec{w}_2||_\infty) \tag{4.3}$$

where:

- $x(t, x_i(0), \vec{w}_i)$ represents the state of the system at time t, with initial state $x_i(0)$ and disturbance sequence $\vec{w}_i$
- $||x_1(0) - x_2(0)||$ represents the norm of the distance between the two initial states
- $||\vec{w}_1 - \vec{w}_2||_\infty$ represents the infinite norm of the distance between the two disturbance sequence, i.e., $||\vec{w}_1 - \vec{w}_2||_\infty = \max_{t \geq 0} ||w_1(t) - w_2(t)||$

■

Definition 4.2.5. Consider a generic system with two states $x_1(t), x_2(t)$, affected by disturbances $w_1(t), w_2(t)$, respectively. $V(\cdot, \cdot)$ is a δISS Lyapunov function if $\exists$ a $\alpha \in \mathcal{K}_\infty$ and $\sigma \in \mathcal{K}$

$$V(x_1(t+1), x_2(t+1)) \leq V(x_1(t), x_2(t)) + \sigma(||w_1(t) - w_2(t)||) - \alpha(||x_1(t) - x_2(t)||) \quad (4.4)$$

■

In Definition 4.2.5, the term $\alpha(\|x_1(t) - x_2(t)\|)$ enforces a decrease of the Lyapunov function when the two state trajectories are different, while $\sigma(\|d_1(t) - d_2(t)\|)$ accounts for the possible increase due to a mismatch between the disturbances.
As in [4], the following theorem provides a sufficient condition for the δISS property of the discrete-time autonomous system (4.2).

Theorem 4.1. *If system (4.2) admits a time-invariant δISS Lyapunov function, then it is δISS with respect to $w(t)$.*

■

If the δISS property holds with respect to $w(t)$, then the system (4.2) is incrementally robust with respect to the disturbances. This property is necessary to define the concept of the incrementally Robust Positive Invariant (δRPI) Set, a fundamental concept in the robust control method presented in this chapter and better defined below.

Definition 4.2.6. [4] A set $\mathbb{E} \subset \mathbb{R}^{n_x} \times \mathbb{R}^{n_x}$ is said to be δRPI under the dynamics of system (4.2) if, for any pair of initial states $(x_1(0), x_2(0)) \in \mathbb{E}$, the state trajectories $(x(t, x_1(0), \vec{w}_1), x(t, x_2(0), \vec{w}_2)) \in \mathbb{E}, \forall t \geq 0, \forall w_1, w_2 \in \mathbb{W}$.

■

In the robust control framework, to ensure that the effect of disturbances remains bounded over time, it is necessary that a system has well-defined δRPI set.

4.3. The nominal model

In robust NMPC context, a crucial role is played by the nominal model, which represents the expected system dynamics in case of absence of disturbances and provides the basis for handling uncertainties.

The nominal model used for computing the state prediction on the robust NMPC optimization problem is:

$$\hat{x}(t+1) = f(\hat{x}(t), \hat{u}(t), 0) \tag{4.5}$$

where $\hat{x}(t) \in \mathbb{R}^{n_x}$ and $\hat{u}(t) \in \mathbb{R}^{n_u}$ are the nominal state and input vectors, respectively. Nominal variables are deterministic and do not account for uncertainties. The nominal system represents an ideal reference system, used to calculate optimal trajectories and control sequences.

In the simplest case, the relationship between the nominal input $\hat{u}(t)$ and the real input $u(t)$ is of the following type

$$u(t) = \hat{u}(t) + K(x(t) - \hat{x}(t)) \tag{4.6}$$

where the gain K must be computed, if possible, in order to guarantee δISS stability properties to the system.

The idea behind the tube-based approach consists in designing a controller such that the real state trajectory remains in a "tube" with respect to the nominal one. For this purpose we can define the error between the state of the real system and the nominal state as:

$$\delta x(t) = x(t) - \hat{x}(t) \tag{4.7}$$

Under the δISS property, it is possible to define explicitly the δISS Lyapunov function $V(x(t), \hat{x}(t)) = V(\delta x(t))$ such that $\exists\ \alpha(\cdot) \in \mathcal{K}_\infty, \sigma(\cdot) \in \mathcal{K}$ such that

$$V(\delta x(t+1)) \leq V(\delta x(t)) + \sigma(||w(t)||) - \alpha(||\delta x(t)||) \tag{4.8}$$

for all $\hat{u}(t) \in \mathbb{U}$. In this way, we can in turn define the set

$$\delta\mathbb{X} := \{\delta x \in \mathbb{R}^{n_x} | V(\delta x) \leq b\} \tag{4.9}$$

such that, if b is properly tuned, $\delta\mathbb{X}$ is positively invariant, i.e., if $\delta x(t) \in \delta\mathbb{X}$, then $\delta x(t+1) \in \delta\mathbb{X}$, for all $w(t) \in \mathbb{W}$.

4.4. Constraint tightening

In the tube-based framework, the constraints $u(t) \in \mathbb{U}$ and $x(t) \in \mathbb{X}$ can be enforced by guaranteeing that $\hat{u}(t) \in \hat{\mathbb{U}}$ and $\hat{x}(t) \in \hat{\mathbb{X}}$, where $\hat{\mathbb{U}}$ and $\hat{\mathbb{X}}$ are suitable tightened sets with respect to $\mathbb{U}$ and $\mathbb{X}$, respectively. This concept is known as constraint tightening and is detailed in this section. The main idea of constraint tightening is to impose stringent constraints on the nominal system state and input, in order to ensure that the real system variables remain within a predefined set.
This concept relies on some set operation defined below:

- Minkowski set sum [16]: $\mathbb{C} = \mathbb{A} \oplus \mathbb{B} := \{\, c = a + b \mid a \in \mathbb{A},\ b \in \mathbb{B} \,\}$
- Minkowski (or Pontryagin) set difference [10]: $\mathbb{C} = \mathbb{A} \ominus \mathbb{B} := \{\, c \mid c \oplus \mathbb{B} \subseteq \mathbb{A} \,\}$

The idea is to enforce the nominal state $\hat{x}(t)$ to belong to a tightened set, obtained by reducing the original constraint set, in such a way that:

$$\hat{x}(t) \in \hat{\mathbb{X}} \Rightarrow x(t) \in \mathbb{X} \tag{4.10}$$

This implication can be obtained provided that $\hat{\mathbb{X}} \subseteq \mathbb{X} \ominus \delta\mathbb{X}$.
In fact

$$x(t) = \hat{x}(t) + \delta x(t) \in \hat{\mathbb{X}} \oplus \delta\mathbb{X} \subseteq \mathbb{X} \ominus \delta\mathbb{X} \oplus \delta\mathbb{X} \subseteq \mathbb{X} \tag{4.11}$$

This ensures that the satisfaction of original constraints $\mathbb{X}$, even for the worst-case disturbance $w(t)$.

Following the same procedure, the tightened input set is defined such that $\hat{\mathbb{U}} \subseteq \mathbb{U} \ominus K\delta\mathbb{X}$.
This implies that

$$\hat{u} \in \hat{\mathbb{U}} \ \Rightarrow\ u(t) = \hat{u}(t) + K\delta x(t) \in \mathbb{U} \tag{4.12}$$

Finally, it is important to remark that the reference setpoint is required to be associated with a steady-state pair $(\bar{x}, \bar{u})$ satisfying:

$$\begin{cases} \bar{x} = f(\bar{x}, \bar{u}) \\ \bar{y} = g(\bar{x}, \bar{u}) \end{cases} \tag{4.13}$$

with $\bar{u} \in \mathcal{INT}(\hat{\mathbb{U}})$ and $\bar{x} \in \mathcal{INT}(\hat{\mathbb{X}})$.

4.5. Online optimal control problem

4.5.1. The finite-horizon optimal control problem

At each time instant t the optimization variables are the nominal state $\hat{x}(t)$ and the nominal inputs $\hat{u}(t), \ \dots \ , \hat{u}(t+N_h-1)$, over a given prediction horizon N_h.
The prediction horizon represents the number of steps over which the MPC algorithm predicts the future system behavior using the nominal model (4.5), optimizes the cost function, and accounts for the constraints. The choice of the prediction horizon N_h determines a compromise between control performance and real-time feasibility in terms of computational cost. Specifically, a small prediction horizon leads to a problem which is easier to solve and less computationally demanding; however, it increases the risk of instability and may result in poor prediction accuracy. On the other hand, a high value of N_h improves tracking and constraint handling, providing more accurate predictions, but at the cost of being highly computationally demanding, conservative and more sensible to model errors.
Therefore, the prediction horizon must be selected in order to guarantee a trade-off between good performances, stability and computational feasibility.

Given the steady-state reference triple $\bar{x}$, $\bar{u}$, and $\bar{y}$ the cost function to be minimized is, in the simple cases, commonly quadratic, i.e.,

$$J = \sum_{k=0}^{N_h-1} \left(||\hat{x}(t+k) - \bar{x}||_Q^2 + ||\hat{u}(t+k) - \bar{u}||_R^2 \right) + V_f(\hat{x}(t+N_h) - \bar{x}) \tag{4.14}$$

where the terms Q and R must be carefully selected in order to guarantee a suitable trade-off between tracking accuracy and control moderation, i.e.,

- $Q \in \mathbb{R}^{n_x \times n_x}$ is a semi-positive definite state weighting matrix, which penalizes deviations of the nominal state $\hat{x}(t)$ from the reference state $\bar{x}$. In particular, higher values of the Q term increase the sensitivity of the system to state errors, enforcing fast convergence of the state to the target.
- $R \in \mathbb{R}^{n_u \times n_u}$ is a positive definite input weighting matrix, which penalizes the control effort. In particular, higher values of the R term limit large variations in the control input, using low control effort for correcting deviations from the desired state.

$V_f(\hat{x}(t+N))$ is the terminal cost, commonly defined according to the Lyapunov function of the system (4.5), under suitable stabilizing auxiliary control law, see Section 4.5.2.

The Finite Horizon Optimization Control Problem (FHOCP) is the following:

$$\min_{\hat{x}(t), \hat{u}(t), \ \dots \ , \hat{u}(t+N_h-1)} J \tag{4.15}$$

subject to:

$$x(t) - \hat{x}(t) \in \delta\mathbb{X}, \tag{4.16}$$

$$\forall k = 0, \ldots, N_h - 1 :$$

$$\hat{x}(t+k+1) = f\big(\hat{x}(t+k), \hat{u}(t+k), 0\big), \tag{4.17}$$

$$\hat{u}(t+k) \in \hat{\mathbb{U}}, \tag{4.18}$$

$$\hat{x}(t+k) \in \hat{\mathbb{X}}, \tag{4.19}$$

$$\hat{x}(t+N_h) \in \mathbb{X}_f. \tag{4.20}$$

The solution to the FHOCP at time instant t is represented by $\hat{x}(t|t), \hat{u}([t : N_h - 1]|t)$. This control strategy is implemented at each time instant over time according to receding horizon procedure.

4.5.2. Terminal ingredients

Two fundamental properties, i.e., convergence and recursive feasibility, are guaranteed by the accurate design of the terminal ingredients, i.e., the terminal set $\mathbb{X}_f$, i.e., the set to which the nominal state must belong at the end of the prediction horizon, and the terminal cost $V_f(\hat{x}(t+N) - \bar{x})$.

Constraint (4.20) is crucial to ensure recursive feasibility, that is, the property guaranteeing that if the optimization problem is feasible at a given time step, it remains feasible at all subsequent time steps.
Moreover, the terminal region is designed to be reachable within N_h steps. Consequently, the convergence property is verified: once the system enters $\mathbb{X}_f$, the auxiliary control law is able to steer the state to the reference, while satisfying all constraints.

The definition of terminal ingredients, is based on the auxiliary control law, which is often a simple control law (for tracking control problems) of the type

$$\hat{u}(t) = K_f(\hat{x}(t) - \bar{x}) + \bar{u} \tag{4.21}$$

where K_f is defined such that the equilibrium $\bar{x}$ obtained setting $\hat{u}(t) = \bar{u}$, is asymptotically stable.

The terminal cost and the terminal set must be defined in such a way that, if the system (4.5) is controlled with (4.21), then

$$V_f(\hat{x}(t+1) - \bar{x}) \leq V_f(\hat{x}(t) - \bar{x}) - ||\hat{x}(t) - \bar{x}||_Q^2 - ||\hat{u}(t) - \bar{u}||_R^2 \tag{4.22}$$

and, if $\hat{x}(t) \in \mathbb{X}_f$, then $\hat{x}(t+1) \in \mathbb{X}_f$, meaning that the terminal set is positively invariant.

5 | Decentralized Nonlinear Model Predictive Control of Structured Recurrent Equilibrium Network Models

Decentralized and distributed architectures represent alternatives developed to overcome centralized control limitations: they allow local controller to operate using limited information, while still guaranteeing the control objective satisfaction. In these approaches, the system models needs to be structured into independent subsystems, each governed by a local controller. While decentralized architectures are characterized by the absence of communication between local controllers that operate independently from each other, in distributed control schemes the local controllers interact by exchanging relevant information. The Robust NMPC framework discussed in Chapter 4 can be leveraged to design decentralized and distributed controllers.

In particular, the objective of this chapter is to formulate the decentralized constrained tracking problem. For this purpose, we consider a system modeled by a Recurrent Equilibrium Network, described in Chapter 2 and by equations (2.1).

5.1. Problem statement

To formulate the decentralized NMPC problem, the system model must be decomposed into N interconnected subsystems.
In Section 2.3, we presented three different decomposition approaches, as a basis for model definition, used for control design.

The Structured matrix decomposition, defined in Section 2.4, is the selected approach for the controller design, because of its ease of implementation, its ability to reflect the physical interconnections of the system, and its good performance in terms of RMSE and

$\text{FIT}_{[\%]}$.

First of all, for tracking objectives, we define the set points $\bar{y}_i$, $i = 1, ..., N$ and the corresponding steady-state inputs $\bar{u}_i$, states $\bar{x}_i$, and nonlinearities $\bar{s}_i$, i.e. such that, for $i = 1, ..., N$:

$$\begin{cases} \bar{x}_i = A_{x_i}\bar{x}_i + B_{u_i}\bar{u}_i + B_{s_i}\bar{s}_i + B_{y_i}\bar{y}_i \\ \bar{s}_i = \sigma\big(\tilde{A}_{x_i}\bar{x}_i + \tilde{B}_{u_i}\bar{u}_i + \tilde{B}_{s_i}\bar{s}_i + \tilde{B}_{y_i}\bar{y}_i\big) \\ \bar{y}_i = C_i\bar{x}_i + D_{u_i}\bar{u}_i + D_{s_i}\bar{s}_i + \sum_{j\in\mathcal{N}_i^0}\big(C_{ij}\bar{x}_j + D_{u_{ij}}\bar{u}_j\big) \end{cases} \tag{5.1}$$

The setpoints are assumed to be available to all local controllers, as they represent the desired steady-state operating condition.
We can rewrite (2.18a) and (2.18b) as

$$x_i(t+1) = A_i x_i(t) + B_i u_i(t) + B^0_{s_i} s_i(t) + \sum_{j\in\mathcal{N}_i^0}\big(A_{ij}(x_j(t)-\bar{x}_j) + B_{ij}(u_j(t)-\bar{u}_j)\big) + \\ + \sum_{j\in\mathcal{N}_i^0}\big(A_{ij}\bar{x}_j + B_{ij}\bar{u}_j\big) \tag{5.2a}$$

$$s_i(t) = \sigma\Big(\tilde{A}_i x_i(t) + \tilde{B}_i u_i(t) + \tilde{B}^0_{s_i} s_i(t) + \sum_{j\in\mathcal{N}_i^0}\big(\tilde{A}_{ij}(x_j(t)-\bar{x}_j) + \tilde{B}_{ij}(u_j(t)-\bar{u}_j)\big) + \\ + \sum_{j\in\mathcal{N}_i^0}\big(\tilde{A}_{ij}\bar{x}_j + \tilde{B}_{ij}\bar{u}_j\big)\Big) \tag{5.2b}$$

In this way, the contribution provided by the deviation of the interconnection variables with respect to their steady-state values on the model can be highlighted and regarded as disturbances to be properly attenuated by the dedicated controller.
In fact, we denote

$$w_i(t) = \sum_{j\in\mathcal{N}_i^0}\big(A_{ij}(x_j(t)-\bar{x}_j) + B_{ij}(u_j(t)-\bar{u}_j)\big) \tag{5.3a}$$

$$\tilde{w}_i(t) = \sum_{j\in\mathcal{N}_i^0}\big(\tilde{A}_{ij}(x_j(t)-\bar{x}_j) + \tilde{B}_{ij}(u_j(t)-\bar{u}_j)\big) \tag{5.3b}$$

and (5.2a) and (5.2b) can be rewritten as

$$x_i(t+1) = A_i x_i(t) + B_i u_i(t) + B^0_{s_i} s_i(t) + w_i(t) + \sum_{j\in\mathcal{N}_i^0}\big(A_{ij}\bar{x}_j + B_{ij}\bar{u}_j\big) \tag{5.4a}$$

$$s_i(t) = \sigma\Big(\tilde{A}_i x_i(t) + \tilde{B}_i u_i(t) + \tilde{B}^0_{s_i} s_i(t) + \tilde{w}_i(t) + \sum_{j\in\mathcal{N}_i^0}\big(\tilde{A}_{ij}\bar{x}_j + \tilde{B}_{ij}\bar{u}_j\big)\Big) \tag{5.4b}$$

Note that, in this way, each subsystem can be represented by a local model (5.4a), (5.4b) belonging to the general class (4.1), where the equivalent perturbation (5.3a), (5.3b) is additive. In view of this, the controller design phase can follow the guidelines of robust control sketched in Chapter 4.

Before introducing the nominal model definition and the controller design, it is necessary to specify some assumptions.

Assumption 5.1.1. The state and input vectors are constrained, i.e., $x(t) \in \mathbb{X}$ and $u(t) \in \mathbb{U}$. Sets $\mathbb{X}$ and $\mathbb{U}$ must be compact and convex.
In particular, for all $j = 1, ..., N$, we have the following:

$$x_j(t) \in \mathbb{X}_j := \{x_j \in \mathbb{R}^{n_{x_j}} : (x_j(t) - \bar{x}_j)^T Q^0_{x_j} (x_j(t) - \bar{x}_j) \leq 1\} \tag{5.5}$$

$$u_j(t) \in \mathbb{U}_j := \{u_j \in \mathbb{R}^{n_{u_j}} : (u_j(t) - \bar{u}_j)^T Q^0_{u_j} (u_j(t) - \bar{u}_j) \leq 1 \tag{5.6}$$

and where $Q^0_{x_j}$ and $Q^0_{u_j}$ are symmetric and positive definite matrices of suitable dimensions.

5.2. Nominal model and tube-based definition

The first step for the application of tube-based robust NMPC to (5.4a), (5.4b) is to define the nominal model, corresponding to the original one, but discarding the perturbation terms w_i and $\tilde{w}_i$ defined in (5.3a) and (5.3b), respectively. Note, in passing, that in view of (5.5), (5.6), and (5.3a), (5.3b), it is possible to define, for all $i = 1, ..., N$ bounded sets $\mathbb{W}_i$ and $\tilde{\mathbb{W}}_i$, where the disturbances w_i and $\tilde{w}_i$ are bounded to lie.

In particular, the latter sets need to be defined in such a way that, for all $i = 1, ..., N$

$$\sum_{j \in \mathcal{N}_i^0} \left(A_{ij}(x_j(t) - \bar{x}_j) + B_{ij}(u_j(t) - \bar{u}_j) \right) \in \mathbb{W}_i \tag{5.7a}$$

$$\sum_{j \in \mathcal{N}_i^0} \left(\tilde{A}_{ij}(x_j(t) - \bar{x}_j) + \tilde{B}_{ij}(u_j(t) - \bar{u}_j) \right) \in \tilde{\mathbb{W}}_i \tag{5.7b}$$

for all $u_j \in \mathbb{U}_j$ and $x_j \in \mathbb{X}_j$, $j \in \mathcal{N}_i^0$.
These sets will be defined in the following chapter through suitable linear matrix inequalities.

The nominal model is the following.

$$\hat{x}_i(t+1) = A_i\hat{x}_i(t) + B_i\hat{u}_i(t) + B^0_{s_i}\hat{s}_i(t) + \sum_{j\in\mathcal{N}^0_i}\big(A_{ij}\bar{x}_j + B_{ij}\bar{u}_j\big) \tag{5.8a}$$

$$\hat{s}_i(t) = \sigma\Big(\tilde{A}_i\hat{x}_i(t) + \tilde{B}_i\hat{u}_i(t) + \tilde{B}^0_{s_i}\hat{s}_i(t) + \sum_{j\in\mathcal{N}^0_i}\big(\tilde{A}_{ij}\bar{x}_j + \tilde{B}_{ij}\bar{u}_j\big)\Big) \tag{5.8b}$$

The input u_i is defined as follows:

$$u_i(t) = \hat{u}_i(t) + K^i_x\big(x_i(t) - \hat{x}_i(t)\big) + K^i_s\big(s_i(t) - \hat{s}_i(t)\big) \tag{5.9}$$

Note that the adoption of two control gains K^i_x and K^i_s, with respect to the law (4.6), provides greater flexibility and improved corrective action in the controller design.
From (5.4a, 5.4b), (5.8a, 5.8b), and (5.9),

$$x_i(t+1) - \hat{x}_i(t+1) = (A_i + B_iK^i_x)(x_i(t) - \hat{x}_i(t)) + (B^0_{s_i} + B_iK^i_s)(s_i(t) - \hat{s}_i(t)) + w_i(t) \tag{5.10}$$

As in the centralized robust control formulation, discussed in Chapter 4, through proper choices of K^i_x and K^i_s, and under Assumption 5.1.1, we can define the RPI set $\delta\mathbb{X}_i$ such that the estimation error

$$x_i(t) - \hat{x}_i(t) \in \delta\mathbb{X}_i, \text{ for all } i = 1, ..., N \tag{5.11}$$

In particular, in view of Definition 4.2.6, since $w_i(t)$ is bounded, if $\delta x_i(t) = x_i(t) - \hat{x}_i(t) \in \delta\mathbb{X}_i$, then $\delta x_i(t+1) \in \delta\mathbb{X}_i$.
Similarly, we will define, for all $i = 1, ..., N$ sets $\delta\mathbb{S}_i$ and $\delta\mathbb{U}_i$, such that

$$s_i(t) - \hat{s}_i(t) \in \delta\mathbb{S}_i \tag{5.12}$$

$$u_i(t) - \hat{u}_i(t) \in \delta\mathbb{U}_i, \tag{5.13}$$

Consistently with the approach presented in Chapter 4, at time t, the MPC optimal control problem addresses the nominal system variables. As discussed in Section 4.4, it is possible to define tightened constraints, for all subsystems, in order to ensure the satisfaction of original constraints, even in the worst case effect of bounded disturbances $w(t)$. The tightened constraints, applied to variables $\hat{x}_i$ and $\hat{u}_i$ are

$$\hat{x}_i(t) \in \hat{\mathbb{X}}_i, \quad \hat{u}_i(t) \in \hat{\mathbb{U}}_i, \quad \forall i = 1, ...N, \forall t \geq 0$$

The sets $\hat{\mathbb{X}}_i$ and $\hat{\mathbb{U}}_i$ will be obtained by properly tightening the original sets $\mathbb{X}_i$ and $\mathbb{U}_i$,

using Pontryagin difference, as follows:

$$\hat{\mathbb{X}}_i \subseteq \mathbb{X}_i \ominus \delta\mathbb{X}_i, \quad \hat{\mathbb{U}}_i \subseteq \mathbb{U}_i \ominus \left(K_x^i \delta\mathbb{X}_i \oplus K_s^i \delta\mathbb{S}_i\right)$$

5.3. Online optimal control problem

As discussed, the decentralized MPC algorithm problem reduces to solving N separated robust MPC problems. For each subsystem, a FHOCP is solved at each time step t, to find the optimization variables coinciding with the input sequence $\hat{u}_i(t), ..., \hat{u}_i(t+N_h-1)$ over a prediction horizon of length N_h and nominal state $\hat{x}_i(t)$.
Under the auxiliary control law, the FHOCP for the i-th subsystem is formulated as follows

$$\min_{\hat{x}_i(t), \hat{u}_i(t), \ \dots \ , \hat{u}_i(t+N_h-1)} J_i \tag{5.14}$$

subject to:

$$x_i(t) - \hat{x}_i(t) \in \delta\mathbb{X}_i, \tag{5.15}$$

$$\forall k = 0, \dots, N_h - 1 :$$

$$\hat{x}_i(t+k+1) = (A_i + B_i K_x^i)\hat{x}_i(t+k) + (B_{s_i}^0 + B_i K_s^i)\hat{s}_i(t+k) + \hat{d}_i(t+k), \tag{5.16}$$

$$\hat{u}_i(t+k) \in \hat{\mathbb{U}}_i, \tag{5.17}$$

$$\hat{x}_i(t+k) \in \hat{\mathbb{X}}_i, \tag{5.18}$$

$$\hat{x}_i(t+N_h) \in \mathbb{X}_{f_i}. \tag{5.19}$$

where $\hat{\mathbb{X}}_i$ and $\hat{\mathbb{U}}_i$ are obtained by properly tightening the original sets, as previously specified.

The i-th controller minimizes the following cost function:

$$J_i = \sum_{k=0}^{N_h-1} \left(||\hat{x}_i(t+k) - \bar{x}_i||_{T_i}^2 + ||\hat{u}_i(t+k) - \bar{u}_i||_{R_i}^2 \right) + V_{f_i}(\hat{x}_i(t+N_h) - \bar{x}_i) \tag{5.20}$$

where the terms $T_i \in \mathbb{R}^{n_{x_i} \times n_{x_i}}$ and $R_i \in \mathbb{R}^{n_{u_i} \times n_{u_i}}$ must be properly chosen to ensure an appropriate balance between tracking performance and control effort, and the terminal cost V_{f_i} is commonly defined based on the system's Lyapunov function, under an appropriate stabilizing auxiliary control law.

The solution of the FHOCP is iteratively evaluated, according to receding horizon procedure, and at time t it is represented by $\hat{x}_i(t|t), \hat{u}_i([t : N_h - 1]|t)$.

5.4. Terminal ingredients

As done in Chapter 4, the two fundamental properties of convergence and recursive feasibility are ensured by the design of the terminal ingredients for each subsystem i: the terminal set $\mathbb{X}_{f,i}$ and the terminal cost $V_{f,i}(\hat{x}_i(t+N) - \bar{x}_i)$.

For tracking purposes, the terminal ingredients are defined based on a local auxiliary control law of the type

$$\hat{u}_i(t) = K_x^i\big(\hat{x}_i(t) - \bar{x}_i\big) + K_s^i\big(\hat{s}_i(t) - \bar{s}_i\big) + \bar{u}_i \tag{5.21}$$

where K_x^i and K_s^i guarantee asymptotic stability of the equilibrium $\bar{x}_i$.

The terminal cost and set are chosen such that, under this control law, the decrease condition

$$V_{f_i}\big(\hat{x}_i(t+1) - \bar{x}_i\big) \leq V_{f_i}\big(\hat{x}_i(t) - \bar{x}_i\big) - \|\hat{x}_i(t) - \bar{x}_i\|_{T_i}^2 - \|\hat{u}_i(t) - \bar{u}_i\|_{R_i}^2 \tag{5.22}$$

holds, when the nominal system (5.8a), (5.8b) is controlled using (5.21), and $\mathbb{X}_{f_i}$ is positively invariant.

In the proposed robust decentralized MPC scheme, the terminal ingredients are constructed consistently with the tube-based formulation.
In particular, the terminal cost is defined as

$$V_{f_i} = \|\hat{x}_i(t) - \bar{x}_i(t)\|_{T_{f_i}}^2,$$

where T_{f_i} is obtained as a suitably scaled version of the matrix P_i, where P_i is the solution of the nominal Lyapunov equation associated with the local stabilizing feedback law (5.21). The scaling is introduced to guarantee the decrease condition of the terminal cost in the presence of bounded disturbances and coupling effects, thus ensuring robust stability.

Similarly, the terminal set $\mathbb{X}_{f_i}$ is not chosen as the nominal invariant set, but as a properly selected subset of the set $\delta\mathbb{X}_i$. More precisely, $\mathbb{X}_{f_i}$ is designed so as to be a robust positively invariant set for the closed-loop error dynamics under the local auxiliary controller and is chosen so as to be compatible with the tube-based constraint tightening.
In particular, it satisfies

$$\mathbb{X}_{f_i} \oplus \delta\mathbb{X}_i \subseteq \mathbb{X}_i$$

so that, once the nominal state enters $\mathbb{X}_{f_i}$, the real state and input trajectories satisfy the original constraints for all admissible disturbances.

With this construction, recursive feasibility and convergence of each local closed-loop subsystem are guaranteed once the subsystem state enters $\mathbb{X}_{f_i}$, the auxiliary controller steers it to the reference while satisfying all constraints.

6 | Offline design of Decentralized NMPC for Structured REN

This chapter presents the offline design procedure required to define all the ingredients needed by the decentralized NMPC scheme. Using LMI-based conditions, we design local gains guaranteeing the δISS property, define suitable RPI error sets, and enforce consistency between disturbance bounds and input and state constraints for the constraint tightening. An algorithm to find all necessary matrices for the MPC is developed via grid search.

6.1. Algorithm design

In this section we describe the design procedure that must be carried out to provide the necessary guarantees and ingredients for the decentralized MPC algorithm, described later in the chapter.
In particular, different properties must be enforced through a suitable design, and consistency between the different sets must be ensured.
Before to delve into the details we first provide a sketch of the requirements and properties that will be guaranteed through suitable LMIs.

- First, we need to design gains K_x^i and K_s^i in (5.9), for all $i = 1, ...N$, in such a way that (5.4a, 5.4b) enjoys the δISS property. This, in turn, ensures stable error dynamics (5.10), which in turn guarantees that, if $w_i(t) \in \mathbb{W}_i$ and if $\tilde{w}_i(t) \in \tilde{\mathbb{W}}_i$ (where $\mathbb{W}_i$ and $\tilde{\mathbb{W}}_i$ must be defined such that (5.7a) and (5.7b), respectively, are verified), then there exists the RPI set $\delta\mathbb{X}_i$ described in (5.11).
- Secondly, we need to provide the explicit definition of the RPI sets $\delta\mathbb{X}_i$, for all $i = 1, ..., N$.
- In order to make it possible to define the tightened sets $\hat{\mathbb{X}}_i$ and $\hat{\mathbb{U}}_i$, we need to ensure that

$$\delta\mathbb{X}_i \subset \mathbb{X}_i \tag{6.1}$$

$$\delta\mathbb{U}_i \subset \mathbb{U}_i \tag{6.2}$$

The latter, in view of (5.9), requires to define, not only $\delta\mathbb{X}_i$ where $x_i(t) - \hat{x}_i(t)$ lies, but also $\delta\mathbb{S}_i$ in which the difference $s_i(t) - \hat{s}_i(t)$ is bounded to lie.

6.1.1. Design of K_x^i and K_s^i for δISS

The main result of this section is the following theorem.

Theorem 6.1. *Let assumption 5.1.1 hold. For any $i, ..., N$, if there exist matrices*

- $Q_i \in \mathbb{R}^{n_{x_i} \times n_{x_i}}$ *such that* $Q_i = Q_i^T \succ 0$
- $\tilde{Q}_{x_i} \in \mathbb{R}^{n_{x_i} \times n_{x_i}}$ *such that* $\tilde{Q}_{x_i} = \tilde{Q}_{x_i}^T \succ 0$
- $Q_{w_i} \in \mathbb{R}^{(n_{x_i}+n_{u_i}) \times (n_{x_i}+n_{u_i})}$ *such that* $Q_{w_i} = Q_{w_i}^T \succ 0$
- $U_i \in \mathbb{R}^{n_{s_i} \times n_{s_i}}$, *diagonal matrix, such that* $U_i \succ 0$
- $Z_i \in \mathbb{R}^{n_{u_i} \times n_{x_i}}$, $\tilde{Z}_i \in \mathbb{R}^{n_{u_i} \times n_{s_i}}$

that satisfy the following condition

$$M_{\delta ISS} = \begin{bmatrix} Q_i - \tilde{Q}_{x_i} & (\cdot)^T & 0 & (\cdot)^T \\ -\tilde{A}_i Q_i - \tilde{B}_i Z_i & -E_{\delta ISS} & -\tilde{D}_i & (\cdot)^T \\ 0 & \tilde{D}_i^T & Q_{w_i} & D_i^T \\ A_i Q_i + B_i Z_i & B_{si} U_i + B_i \tilde{Z}_i & D_i & Q_i \end{bmatrix} \geq 0 \tag{6.3}$$

where $E_{\delta ISS} = U_i^T \tilde{B}_{si}^T + \tilde{Z}_i^T \tilde{B}_i^T - 2U_i + \tilde{B}_{si} U_i + \tilde{B}_i \tilde{Z}_i$, then if we set $K_x^i = Q_i^{-1} Z_i$ and $K_s^i = U_i^{-1} \tilde{Z}_i$, (5.4a) and (5.4b) enjoy the δISS property. In particular, this implies that, for all $x_i(t)$ and $\hat{x}_i(t)$, there exists a δISS Lyapunov function

$$V_i(x_i(t) - \hat{x}_i(t)) = ||x_i(t) - \hat{x}_i(t)||_{P_i}^2$$

where $P_i = Q_i^{-1}$ and $Q_{x_i} = P_i \tilde{Q}_{x_i} P_i$, such that

$$||x_i(t+1) - \hat{x}_i(t+1)||_{P_i}^2 \leq ||x_i(t) - \hat{x}_i(t)||_{P_i}^2 - ||x_i(t) - \hat{x}_i(t)||_{Q_{x_i}}^2 + ||w_i(t)||_{Q_{w_i}}^2 \tag{6.4}$$

Proof of the Theorem

Consider the generic i-th subsystem (5.4a), (5.4b), and (5.9).

We can rewrite (5.4a) and (5.4b) as

$$\begin{cases} x_i(t+1) = (A_i + B_i K_x^i)x_i(t) + (B_{s_i}^0 + B_i K_s^i)s_i(t) + d_i(t) \\ s_i(t) = \sigma\big(\tilde{A}_i + \tilde{B}_i K_x^i)x_i(t) + (\tilde{B}_{s_i}^0 + \tilde{B}_i K_s^i)s_i(t) + \tilde{d}_i(t)\big) \end{cases} \tag{6.5}$$

where

$$d_i(t) = \sum_{j \neq i}(A_{ij}\bar{x}_i + B_{ij}\bar{u}_i) + w_i(t) - B_i\big(K_x^i\hat{x}_i(t) + K_s^i\hat{s}_i(t)\big)$$

$$\tilde{d}_i(t) = \sum_{j \neq i}(\tilde{A}_{ij}\bar{x}_i + \tilde{B}_{ij}\bar{u}_i) + \tilde{w}_i(t) - \tilde{B}_i\big(K_x^i\hat{x}_i(t) + K_s^i\hat{s}_i(t)\big)$$

The nominal system equations are

$$\begin{cases} \hat{x}_i(t+1) = (A_i + B_i K_x^i)\hat{x}_i(t) + (B_{s_i}^0 + B_i K_s^i)\hat{s}_i(t) + \hat{d}_i(t) \\ \hat{s}_i(t) = \sigma\big(\tilde{A}_i + \tilde{B}_i K_x^i)\hat{x}_i(t) + (\tilde{B}_{s_i}^0 + \tilde{B}_i K_s^i)\hat{s}_i(t) + \hat{\tilde{d}}_i(t)\big) \end{cases} \tag{6.6}$$

where

$$\hat{d}_i(t) = \sum_{j \neq i}(A_{ij}\bar{x}_i + B_{ij}\bar{u}_i) - B_i\big(K_x^i\hat{x}_i(t) + K_s^i\hat{s}_i(t)\big)$$

$$\hat{\tilde{d}}_i(t) = \sum_{j \neq i}(\tilde{A}_{ij}\bar{x}_i + \tilde{B}_{ij}\bar{u}_i) - \tilde{B}_i\big(K_x^i\hat{x}_i(t) + K_s^i\hat{s}_i(t)\big)$$

From these definitions, note that:

$$d_i(t) - \hat{d}_i(t) = \delta d_i(t) = w_i(t), \quad \tilde{d}_i(t) - \hat{\tilde{d}}_i(t) = \delta\tilde{d}_i(t) = \tilde{w}_i(t) \tag{6.7}$$

For the sake of simplicity and for later use in the subsequent steps, we define the following matrices:

$$A_{Kx_i} = A_i + B_i K_x^i, \quad B_{Ks_i} = B_{s_i}^0 + B_i K_s^i, \quad \tilde{A}_{Kx_i} = \tilde{A}_i + \tilde{B}_i K_x^i, \quad \tilde{B}_{Ks_i} = \tilde{B}_{s_i}^0 + \tilde{B}_i K_s^i \tag{6.8}$$

Moreover, we can define a vector $\mathbf{w}_i = \begin{bmatrix} w_i \\ \tilde{w}_i \end{bmatrix}$ such that

$$\delta d_i = \begin{bmatrix} I & 0 \end{bmatrix}\mathbf{w}_i = D_i\mathbf{w}_i \tag{6.9a}$$

$$\delta\tilde{d}_i = \begin{bmatrix} 0 & I \end{bmatrix}\mathbf{w}_i = \tilde{D}_i\mathbf{w}_i \tag{6.9b}$$

Step 1: Definition of the δISS Lyapunov function

In this proof, we resort to Definition 4.2.5, applied to the evolution of the difference

between $x_1(t) = x_i(t)$ and $x_2(t) = \hat{x}_i(t)$.

$$x_i(t+1) - \hat{x}_i(t+1) = (A_i + B_i K_x^i)(x_i(t) - \hat{x}_i(t)) + (B_{s_i}^0 + B_i K_s^i)(s_i(t) - \hat{s}_i(t)) + \delta d_i(t)$$

$$\Rightarrow \delta x_i(t+1) = A_{Kx_i}\delta x_i(t) + B_{Ks_i}\delta s_i(t) + \delta d_i(t) \tag{6.10}$$

With particular reference to (4.4), we set

$$V(x_i(t) - \hat{x}_i(t)) = ||\delta x_i(t)||_{P_i}^2$$

$$\alpha(||x_1(t) - x_2(t)||) = ||\delta x_i(t)||_{Q_{x_i}}^2$$

$$\sigma(||d_1(t) - d_2(t)||) = ||\delta d_i(t)||_{Q_{w_i}}^2$$

In this way (4.4) can be written as follows

$$||\delta x_i(t+1)||_{P_i}^2 \leq ||\delta x_i(t)||_{P_i}^2 - ||\delta x_i(t)||_{Q_{x_i}}^2 + ||\delta d_i(t)||_{Q_{w_i}}^2 \tag{6.11}$$

Using (6.10) and (6.9a) in (6.11), omitting (from now on) time indexes for simplicity, (6.11) is rewritten as:

$$\begin{aligned}\left(A_{Kx_i}\delta x_i + B_{Ks_i}\delta s_i + D_i\mathbf{w}_i\right)^T P_i\left(A_{Kx_i}\delta x_i + B_{Ks_i}\delta s_i + D_i\mathbf{w}_i\right) - \delta x_i^T P_i \delta x_i \leq \\ - \delta x_i^T Q_{x_i}\delta x_i + (D_i\mathbf{w}_i)^T Q_{w_i} D_i\mathbf{w}_i\end{aligned} \tag{6.12}$$

If we define the vector $\Phi_i^T = [\delta x_i^T \quad \delta s_i^T \quad \mathbf{w}_i^T]$, the final form of Lyapunov Inequality (6.12), expressed as a quadratic constraint, is the following:

$$\Phi_i^T \left(\begin{bmatrix} A_{Kx_i}^T \\ B_{Ks_i}^T \\ D_i^T \end{bmatrix} P_i \begin{bmatrix} A_{Kx_i} & B_{Ks_i} & D_i \end{bmatrix} - \begin{bmatrix} P_i & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} Q_{x_i} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -Q_{w_i} \end{bmatrix} \right) \Phi_i \leq 0 \tag{6.13}$$

Step 2: Characterization of the nonlinearity via the Sector Conditions
According to the Assumption 2.1.1 and Equation (2.3), it is possible to prove [26] that, for all diagonal matrices $W_i > 0$, it holds that:

$$(\delta v_i - \delta s_i)^T W_i \delta s_i \geq 0 \tag{6.14}$$

where $\delta s_i = si - \hat{s}_i$, and $\delta v_i = \tilde{A}_{Kx_i}\delta x_i + \tilde{B}_{Ks_i}\delta s_i + \delta\tilde{d}_i$.

Inequality (6.14) can be rewritten as follows

$$\left(\tilde{A}_{Kx_i}\delta x_i + \tilde{B}_{Ks_i}\delta s_i + \tilde{D}_i \mathbf{w}_i - \delta s_i\right)^T W_i \delta s_i + \delta s_i^T W_i \left(\tilde{A}_{Kx_i}\delta x_i + \tilde{B}_{Ks_i}\delta s_i + \tilde{D}_i \mathbf{w}_i - \delta s_i\right) \geq 0 \tag{6.15}$$

that, in compact form, becomes:

$$\Phi_i^T \left(\begin{bmatrix} \tilde{A}_{Kx_i}^T \\ \tilde{B}_{Ks_i}^T - I \\ \tilde{D}_i \end{bmatrix} W_i \begin{bmatrix} 0 & I & 0 \end{bmatrix} + \begin{bmatrix} 0 \\ I \\ 0 \end{bmatrix} W_i \begin{bmatrix} \tilde{A}_{Kx_i} & \tilde{B}_{Ks_i} - I & \tilde{D}_i \end{bmatrix} \right) \Phi_i \geq 0 \tag{6.16}$$

The final form of the sector condition, expressed as a quadratic constraint, is the following:

$$\Phi_i^T \begin{bmatrix} 0 & \tilde{A}_{Kx_i}^T W_i & 0 \\ W_i \tilde{A}_{Kx_i} & E_{BWB} & W_i \tilde{D}_i \\ 0 & \tilde{D}_i^T W_i & 0 \end{bmatrix} \Phi_i \geq 0 \tag{6.17}$$

where $E_{BWB} = (\tilde{B}_{Ks_i}^T - I)W_i + W_i(\tilde{B}_{Ks_i} - I)$

Step 3: Imposition of Lyapunov Inequality subject to the Sector Condition
As a next step, we impose the Lyapunov Inequality (6.13) under the Sector Condition (6.17). A sufficient condition is obtained by summing elements of Equations (6.13) and (6.17). After some manipulation, this results in:

$$\Phi_i^T \left(\begin{bmatrix} A_{Kx_i}^T \\ B_{Ks_i}^T \\ D_i^T \end{bmatrix} P_i P_i^{-1} P_i \begin{bmatrix} A_{Kx_i} & B_{Ks_i} & D_i \end{bmatrix} + \begin{bmatrix} Q_{x_i} - P_i & \tilde{A}_{Kx_i}^T W_i & 0 \\ W_i \tilde{A}_{Kx_i} & E_{BWB} & W_i \tilde{D}_i \\ 0 & \tilde{D}_i^T W_i & -Q_{w_i} \end{bmatrix} \right) \Phi_i \leq 0 \tag{6.18}$$

By applying the Schur complement, this is obtained if

$$\begin{bmatrix} P_i - Q_{x_i} & -\tilde{A}_{Kx_i}^T W_i & 0 & A_{Kx_i}^T P_i \\ -W_i \tilde{A}_{Kx_i} & -E_{BWB} & -W_i \tilde{D}_i & B_{Ks_i}^T P_i \\ 0 & -\tilde{D}_i^T W_i & Q_{w_i} & D_i^T P_i \\ P_i A_{Kx_i} & P_i B_{Ks_i} & P_i D_i & P_i \end{bmatrix} \geq 0 \tag{6.19}$$

Define $Q_i = P_i^{-1}$, $U_i = W_i^{-1}$ and a block diagonal matrix

$$M_i = \begin{bmatrix} Q_i & 0 & 0 & 0 \\ 0 & U_i & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & Q_i \end{bmatrix}$$

We perform the congruence transformation, consisting in pre- and post-multiplying (6.20) by M_i^T and M_i, respectively, and then we can reformulate (6.20) in

$$\begin{bmatrix} Q_i - Q_i Q_{x_i} Q_i & -Q_i \tilde{A}_{Kx_i}^T & 0 & Q_i A_{Kx_i}^T \\ -\tilde{A}_{Kx_i} Q_i & -U_i E_{BWB} U_i & -\tilde{D}_i & U_i B_{Ks_i}^T \\ 0 & -\tilde{D}_i^T & Q_{w_i} & D_i^T \\ A_{Kx_i} Q_i & B_{Ks_i} U_i & D_i & Q_i \end{bmatrix} \geq 0 \tag{6.20}$$

Define $\tilde{Q}_{x_i} = Q_i Q_{x_i} Q_i$, $Z_i = K_x^i$, and $\tilde{Z}_i = K_s^i U_i$. The final form of the LMI for the δISS is the following:

$$\begin{bmatrix} Q_i - \tilde{Q}_{x_i} & (\cdot)^T & 0 & (\cdot)^T \\ -\tilde{A}_i Q_i - \tilde{B}_i Z_i & -E_{\delta ISS} & -\tilde{D}_i & (\cdot)^T \\ 0 & \tilde{D}_i^T & Q_{w_i} & D_i^T \\ A_i Q_i + B_i Z_i & B_{si} U_i + B_i \tilde{Z}_i & D_i & Q_i \end{bmatrix} \geq 0 \tag{6.21}$$

where $E_{\delta ISS} = U_i E_{BWB} U_i = U_i^T \tilde{B}_{si}^T + \tilde{Z}_i^T \tilde{B}_i^T - 2U_i + \tilde{B}_{si} U_i + \tilde{B}_i \tilde{Z}_i$ ■

6.1.2. Definition of the set of the coupling terms

According to Assumption 5.1.1, we have $u_i(t) \in \mathbb{U}_i$ and $x_i(t) \in \mathbb{X}_i$, where the sets $\mathbb{U}_i$ and $\mathbb{X}_i$ are defined in (5.6) and (5.5). In this section, we show how the set related to the disturbances is defined accordingly.

From (5.3a) and (5.3b) we obtain that

$$\mathbf{w}_i(t) = \sum_{j \in \mathcal{N}_i^0} \begin{bmatrix} A_{ij} & B_{ij} \\ \tilde{A}_{ij} & \tilde{B}_{ij} \end{bmatrix} \begin{bmatrix} x_j(t) - \bar{x_j} \\ u_j(t) - \bar{u_j} \end{bmatrix} \in \mathbb{W}^* \tag{6.22}$$

where we can define matrix Q_{w_i} such that

$$\mathbb{W}^* := \{\mathbf{w}_i(t) \in \mathbb{R}^{n_{x_i} + n_{u_i}} : \mathbf{w}_i(t)^T Q_{w_i} \mathbf{w}_i(t) \leq 1\} \tag{6.23}$$

i.e.,

$$\sum_{j \neq i} \left(\begin{bmatrix} A_{ij} & B_{ij} \\ \tilde{A}_{ij} & \tilde{B}_{ij} \end{bmatrix} \begin{bmatrix} x_j(t) - \bar{x_j} \\ u_j(t) - \bar{u_j} \end{bmatrix} \right)^T Q_{w_i} \sum_{j \neq i} \left(\begin{bmatrix} A_{ij} & B_{ij} \\ \tilde{A}_{ij} & \tilde{B}_{ij} \end{bmatrix} \begin{bmatrix} x_j(t) - \bar{x_j} \\ u_j(t) - \bar{u_j} \end{bmatrix} \right) \leq 1 \tag{6.24}$$

Defining $\begin{bmatrix} A_{ij} & B_{ij} \\ \tilde{A}_{ij} & \tilde{B}_{ij} \end{bmatrix} = A^*_{ij}$ and $\begin{bmatrix} x_j(t) - \bar{x_j} \\ u_j(t) - \bar{u_j} \end{bmatrix} = x_{u_j}$, we can write

$$w_i(t)^T Q_{w_i} w_i(t) = \sum_{l,j \neq i} (A^*_{ij} x_{u_j})^T Q_{w_i} (A^*_{il} x_{u_l}) \leq 1 \tag{6.25}$$

It can be proven that

$$\sum_{l,j \neq i} (A^*_{ij} x_{u_j})^T Q_{w_i} (A^*_{ij} x_{u_j}) \leq \frac{1}{2} \sum_{l,j \neq i} (A^*_{ij} x_{u_j})^T Q_{w_i} (A^*_{ij} x_{u_j}) + \frac{1}{2} \sum_{l,j \neq i} (A^*_{il} x_{u_l})^T Q_{w_i} (A^*_{il} x_{u_l}) \tag{6.26}$$

Also,

$$\sum_{l,j \neq i} (A^*_{ij} x_{u_j})^T Q_{w_i} (A^*_{ij} x_{u_j}) \leq \nu_i \sum_{l \in \mathcal{N}_i^0} (A^*_{il} x_{u_l})^T Q_{w_i} (A^*_{il} x_{u_l}) \tag{6.27}$$

where ν_i is the number of neighbors of the i-th subsystem. Then, combining (6.25) and (6.27), we obtain that the following represents a sufficient condition for $w_i(t)^T Q_{w_i} w_i(t) \leq 1$:

$$\nu_i \sum_{l \in \mathcal{N}_i^0} (A^*_{il} x_{u_l})^T Q_{w_i} (A^*_{il} x_{u_l}) \leq 1, \quad \forall x_{u_l} \in \mathbb{X}_{u_l} \tag{6.28}$$

In turn, (6.28) is guaranteed if, for all $l \in \mathcal{N}_i^0$

$$\nu_i (A^*_{il} x_{u_l})^T Q_{w_i} (A^*_{il} x_{u_l}) \leq \frac{1}{\nu_i}, \quad \text{i.e.,}$$

$$\nu_i^2 (A^*_{il} x_{u_l})^T Q_{w_i} (A^*_{il} x_{u_l}) \leq 1, \tag{6.29}$$

Now, define $Q_{xu_i} = \frac{1}{2} \begin{bmatrix} Q^0_{x_i} & 0 \\ 0 & Q^0_{u_i} \end{bmatrix}$ in such a way that Assumption 5.1.1 can be rewritten as

$$\begin{bmatrix} x_i \\ u_i \end{bmatrix} \in \mathbb{X}_{u_i} := \left\{ \begin{bmatrix} x_i \\ u_i \end{bmatrix} \in \mathbb{R}^{n_{x_i} + n_{u_i}} : \begin{bmatrix} x_i - \bar{x}_i \\ u_i - \bar{u}_i \end{bmatrix}^T Q^0_{xu_i} \begin{bmatrix} x_i - \bar{x}_i \\ u_i - \bar{u}_i \end{bmatrix} \leq 1 \right\} \tag{6.30}$$

Inequality (6.30) implies (6.29), if

$$x_{u_l}^T Q^0_{xu_l} x_{u_l} \geq \nu_i^2 (A^*_{il} x_{u_l})^T Q_{w_i} (A^*_{il} x_{u_l})$$

The latter holds if Q_{w_i} is defined in such a way that

$$\nu_i^2 {A^*_{il}}^T Q_{w_i} A^*_{il} \leq Q^0_{xu_l}, \quad \forall l \in \mathcal{N}_i^0 \tag{6.31}$$

■

6.1.3. Definition of the RPI set $\delta\mathbb{X}_i$

At this point we can define the RPI set $\delta\mathbb{X}_i$, thanks to the following lemma.

Lemma 6.1. *If there exists a scalar $\gamma_i > 0$ such that*

$$P_i \leq \gamma_i Q_{x_i} \tag{6.32}$$

then

$$\delta\mathbb{X}_i := \{\delta x_i \in \mathbb{R}^{n_{x_i}} : \delta x_i^T P_i \delta x_i \leq \gamma_i\}$$

is an RPI set, as defined in (5.11). In particular, for all $w(t)$ such that

$$(w_i(t) - \bar{w}_i)^T Q_{w_i}(w_i(t) - \bar{w}_i) \leq 1$$

and if $x_i(t) - \hat{x}_i(t) \in \delta\mathbb{X}_i$, then $x_i(t+1) - \hat{x}_i(t+1) \in \delta\mathbb{X}_i$.

The proof of Lemma 6.1 is provided in [22].
Note that (6.32) can be rewritten in LMI form. In particular, defining $a_i = \gamma_i^{-\frac{1}{2}}$ and recalling that $Q_i = P_i^{-1}$

$$\begin{bmatrix} \tilde{Q}_{x_i} - \frac{Q_i}{\gamma_i} & a_i I \\ a_i I & Q_i \end{bmatrix} \geq 0 \tag{6.33}$$

6.1.4. Definition of the set $\delta\mathbb{S}_i$

In this section we provide a method to define the set $\delta\mathbb{S}_i$.
The latter is defined as the set where $\delta s_i = s_i - \hat{s}_i$ lies when $\mathbf{w}_i \in \mathbb{W}_i^*$ and $\delta x_i = x_i - \hat{x}_i \in \delta\mathbb{X}_i$, where $\mathbb{W}_i^*$ is defined in Section 6.1.2, whereas $\delta\mathbb{X}_i$ is the RPI set defined in Section 5.2.

In this section we define matrix $S_i > 0$ such that, provided that $\delta x_i \in \delta\mathbb{X}_i$ (i.e., $\delta x_i^T P_i \delta x_i \leq \gamma_i$) and $\mathbf{w}_i \in \mathbb{W}_i^*$ (i.e., $\mathbf{w}_i^T Q_{w_i} \mathbf{w}_i \leq 1$) and under (6.17), then

$$\delta s_i^T S_i \delta s_i \leq 1, \quad \text{i.e., that} \quad \Phi_i^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & S_i & 0 \\ 0 & 0 & 0 \end{bmatrix} \Phi_i - 1 \leq 0 \tag{6.34}$$

We start by considering that a sufficient condition that guarantees that $\delta x_i \in \delta\mathbb{X}_i$ and

$\mathbf{w}_i \in \mathbb{W}_i^*$ is:

$$\Phi_i^T \begin{bmatrix} \frac{P_i}{2\gamma_i} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{Q_{w_i}}{2} \end{bmatrix} \Phi_i - 1 \leq 0 \tag{6.35}$$

In order to define S_i guaranteeing that (6.34) holds under (6.17) and (6.35), we use the S-procedure and we impose (6.34) + (6.17) - (6.35) ≤ 0, i.e.,

$$\Phi_i^T \left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & S_i & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & \tilde{A}_{Kx_i}^T W_i & 0 \\ W_i \tilde{A}_{Kx_i} & E_{BWB} & W_i \tilde{D}_i \\ 0 & \tilde{D}_i^T W_i & 0 \end{bmatrix} - \begin{bmatrix} \frac{P_i}{2\gamma_i} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{Q_{w_i}}{2} \end{bmatrix} \right) \Phi_i \leq 1 - 1 = 0 \tag{6.36}$$

To guarantee (6.36) we impose

$$\begin{bmatrix} \frac{P_i}{2\gamma_i} & -\tilde{A}_{Kx_i}^T W_i & 0 \\ -W_i \tilde{A}_{Kx_i} & -E_{BWB} - S_i & -W_i \tilde{D}_i \\ 0 & -\tilde{D}_i^T W_i & \frac{Q_{w_i}}{2} \end{bmatrix} \geq 0 \tag{6.37}$$

To make (6.37) an LMI, we can multiply both sides of the latter inequality by $\begin{bmatrix} Q_i & 0 & 0 \\ 0 & U_i & 0 \\ 0 & 0 & I \end{bmatrix}$, recalling that $Q_i = P_i^{-1}$ and $U_i = W_i^{-1}$, obtaining

$$\begin{bmatrix} \frac{Q_i}{2\gamma_i} & -(\tilde{A}_i Q_i + \tilde{B}_i Z_i)^T & 0 \\ -\tilde{A}_i Q_i - \tilde{B}_i Z_i & -\tilde{S}_i - \tilde{B}_{s_i}^0 U_i - \tilde{B}_i \tilde{Z}_i + 2U_i - U_i \tilde{B}_{s_i}^{0^T} - \tilde{Z}_i{}^T \tilde{B}_i^T & -\tilde{D}_i \\ 0 & -\tilde{D}_i^T & \frac{Q_{w_i}}{2} \end{bmatrix} \geq 0 \tag{6.38}$$

where $\tilde{S}_i = U_i S_i U_i$, $Z = K_x^i Q_i$ and $\tilde{Z}_i = K_s^i Q_i$. ■

6.1.5. Consistency of the input and output constraints

As specified in Section 5.1,

$$\delta u_i(t) = u_i - \hat{u}_i = K_x^i (x_i(t) - \hat{x}_i) + K_s^i (s_i(t) - \hat{s}_i) = \tag{6.39}$$

$$= \begin{bmatrix} K_x^i & K_s^i \end{bmatrix} \begin{bmatrix} \delta x_i \\ \delta s_i \end{bmatrix} \tag{6.40}$$

This allows us to implicitly define the set $\delta\mathbb{U}_i$ where δu_i lies, when $\delta x_i \in \delta\mathbb{X}_i$ and $\delta s_i \in \delta\mathbb{S}_i$. For simplicity we write the latter, into a unique (although conservative) constraint

$$\begin{bmatrix} x_i - \hat{x}_i \\ s_i - \hat{s}_i \end{bmatrix} \in \Delta_{xs_i} := \left\{ \begin{bmatrix} x_i - \hat{x}_i \\ s_i - \hat{s}_i \end{bmatrix} \in \mathbb{R}^{n_{x_i} \times n_{s_i}} : \begin{bmatrix} x_i - \hat{x}_i \\ s_i - \hat{s}_i \end{bmatrix}^T \begin{bmatrix} \frac{P_i}{2\gamma_i} & 0 \\ 0 & \frac{S_i}{2} \end{bmatrix} \begin{bmatrix} x_i - \hat{x}_i \\ s_i - \hat{s}_i \end{bmatrix} \leq 1 \right\} \tag{6.41}$$

To guarantee full consistency of the so-obtained control design, we need to ensure that, if $\begin{bmatrix} \delta x_i \\ \delta u_i \end{bmatrix} \in \delta\mathbb{X}_i \times \delta\mathbb{U}_i$, then $\begin{bmatrix} \delta x_i \\ \delta u_i \end{bmatrix} \in \delta\mathbb{X}_{u_i}$, in such a way we can define $\hat{\mathbb{X}}_i$ and $\hat{\mathbb{U}}_i$ as tightened counterparts of $\mathbb{X}_i$ and $\mathbb{U}_i$, respectively.

First, we write $x_i(t) - \hat{x}_i(t) \in \delta\mathbb{X}_i$ and $u_i(t) - \hat{u}_i(t) \in \delta\mathbb{U}_i$, into a unique constraint, i.e.,

$$\begin{bmatrix} x_i - \hat{x}_i \\ u_i - \hat{u}_i \end{bmatrix} \in \delta\mathbb{X}_i \times \delta\mathbb{U}_i := \left\{ \begin{bmatrix} x_i - \hat{x}_i \\ u_i - \hat{u}_i \end{bmatrix} \in \mathbb{R}^{n_{x_i} \times n_{u_i}} : \begin{bmatrix} x_i - \hat{x}_i \\ u_i - \hat{u}_i \end{bmatrix}^T \hat{Q}_{xu_i} \begin{bmatrix} x_i - \hat{x}_i \\ u_i - \hat{u}_i \end{bmatrix} \leq 1 \right\} \tag{6.42}$$

To compute $\hat{Q}_{xu_i}$ we recall that

$$\begin{bmatrix} x_i - \hat{x}_i \\ u_i - \hat{u}_i \end{bmatrix}^T \hat{Q}_{xu_i} \begin{bmatrix} x_i - \hat{x}_i \\ u_i - \hat{u}_i \end{bmatrix} = \begin{bmatrix} x_i - \hat{x}_i \\ s_i - \hat{s}_i \end{bmatrix}^T \begin{bmatrix} I & K_x^{iT} \\ 0 & K_s^{iT} \end{bmatrix} \hat{Q}_{xu_i} \begin{bmatrix} I & 0 \\ K_x^i & K_s^i \end{bmatrix} \begin{bmatrix} x_i - \hat{x}_i \\ s_i - \hat{s}_i \end{bmatrix} \tag{6.43}$$

and we impose

$$\begin{bmatrix} x_i - \hat{x}_i \\ u_i - \hat{u}_i \end{bmatrix}^T \hat{Q}_{xu_i} \begin{bmatrix} x_i - \hat{x}_i \\ u_i - \hat{u}_i \end{bmatrix} \leq 1 \tag{6.44}$$

by requiring that

$$\begin{bmatrix} I & K_x^{iT} \\ 0 & K_s^{iT} \end{bmatrix} \hat{Q}_{xu_i} \begin{bmatrix} I & 0 \\ K_x^i & K_s^i \end{bmatrix} \leq \begin{bmatrix} \frac{P_i}{2\gamma_i} & 0 \\ 0 & \frac{S_i}{2} \end{bmatrix} \tag{6.45}$$

Lately, consistency is guaranteed by simply imposing

$$\hat{Q}_{xu_i} > Q^0_{xu_i}, \quad \forall i = 1, ..., N \tag{6.46}$$

■

6.2. Offline design procedure

The overall offline design procedure is described in the following algorithm. In this context, γ_i represents a design parameter that plays a key role in the tuning process. Its value is

determined through a grid-search approach, which starts from the initial choice $\gamma_i = 1$ and then systematically explores the remaining admissible values in order to identify a suitable configuration.

Algorithm 6.1 Offline Design Procedure

for $i = 1, \ldots, N$ **do**

Step 1:

Solve the LMI feasibility problem with optimization variables $\tilde{Q}_{x_i}, Q_i, Q_{w_i}, U_i, Z_i, \tilde{Z}_i, Q^0_{xu_l}$ and $\tilde{S}_i$ under constraints (6.21), (6.31), (6.32), (6.38), $\tilde{Q}_{x_i} > 0, Q_i > 0, Q_{w_i} > 0, U_i > 0, Q^0_{xu_l} > 0$ and $\tilde{S}_i > 0$

if *a solution is found* **then**

Set $Q_{x_i} = Q_i \tilde{Q}_{x_i} Q_i$, $P_i = Q_i^{-1}$, $W_i = U_i^{-1}$, $S_i = W_i \tilde{S}_i W_i$, $K_x^i = Z_i Q_i^{-1}$ and $K_s^i = \tilde{Z}_i U_i^{-1}$

Go to step 2

else

Set $\gamma_{i,\text{new}} = \gamma_i + 1$ and repeat from Step 1

Step 2:

Solve the LMI feasibility problem with optimization variable $\hat{Q}_{xu_i}$, under constraints (6.46), (6.45), and $\hat{Q}_{xu_i} > 0$

if *a solution is found* **then**

Stop

else

Set $\gamma_{i,\text{new}} = \gamma_i + 1$ and repeat from step 1

6.3. Local δISS property

In some cases, the set of LMIs derived in Section 6.1 may not be solvable. This may be potentially due to the fact that the δISS condition derived in Section 6.1.1 is designed to hold globally. In this case, it is possible to use a slightly different, less stringent condition, ensuring local δISS in a bounded set. This property is derived by replacing the global sector condition (6.14) with a local one, which holds only within a prescribed region of the state space. As a consequence, incremental stability is guaranteed locally, while maintaining the convex formulation of the design conditions.

6.3.1. LMI for local δISS

The condition for local δISS is derived according to the following main result.

Theorem 6.2. *Let Assumption 5.1.1 hold and define* $D_i = \begin{bmatrix} I_{n_{x_i}} & 0 \end{bmatrix}, \tilde{D}_i = \begin{bmatrix} 0 & I_{n_{s_i}} \end{bmatrix}$. *For any* $i, ..., N$, *if there exist matrices*

- $Q_i \in \mathbb{R}^{n_{x_i} \times n_{x_i}}$ *such that* $Q_i = Q_i^T \succ 0$
- $\tilde{Q}_{x_i} \in \mathbb{R}^{n_{x_i} \times n_{x_i}}$ *such that* $\tilde{Q}_{x_i} = \tilde{Q}_{x_i}^T \succ 0$
- $Q_{w_i} \in \mathbb{R}^{(n_{x_i}+n_{u_i}) \times (n_{x_i}+n_{u_i})}$ *such that* $Q_{w_i} = Q_{w_i}^T \succ 0$
- $U_i \in \mathbb{R}^{n_{s_i} \times n_{s_i}}$, *diagonal matrix*
- $Z_i \in \mathbb{R}^{n_{u_i} \times n_{x_i}}$, $\tilde{Z}_i \in \mathbb{R}^{n_{u_i} \times n_{s_i}}$
- $\mathbf{L}_i \in \mathbb{R}^{n_{s_i} \times n_{s_i}}$, *diagonal matrix such that* $\mathbf{L}_i \succeq 0$

that satisfy the following condition

$$M_{\delta ISS} \geq -(\boldsymbol{M}_i(\Lambda_i) + \boldsymbol{M}_i(\Lambda_i)^T)/2 \tag{6.47}$$

where $M_{\delta ISS}$ *is defined in (6.3) and*

$$\boldsymbol{M}_i(\Lambda_i) = \begin{bmatrix} \tilde{A}_{Kx_i}^T & \tilde{A}_{Kx_i}^T \\ \tilde{B}_{Ks_i}^T - I & \tilde{B}_{Ks_i}^T \\ \tilde{D}_i^T & \tilde{D}_i^T \end{bmatrix} \begin{bmatrix} \mathbf{L}_i & 0 \\ 0 & \mathbf{L}_i \end{bmatrix} \begin{bmatrix} \tilde{A}_{Kx_i} & \tilde{B}_{Ks_i} & \tilde{D}_i \\ \tilde{A}_{Kx_i} & \tilde{B}_{Ks_i} - I & \tilde{D}_i \end{bmatrix} \tag{6.48}$$

where $\tilde{A}_{Kx_i}, \tilde{B}_{Ks_i}$ *are defined in (6.8),*
then if we set $K_x^i = Q_i^{-1} Z_i$ *and* $K_s^i = U_i^{-1} \tilde{Z}_i$, *(5.4a) and (5.4b) enjoy the* δ*ISS property locally. In particular, this implies that, for all* $x_i(t)$ *and* $\hat{x}_i(t)$ *in a sufficiently small*

neighborhood of the origin, there exists a δISS Lyapunov function

$$V_i(x_i(t) - \hat{x}_i(t)) = ||x_i(t) - \hat{x}_i(t)||^2_{P_i}$$

where $P_i = Q_i^{-1}$ *and* $Q_{x_i} = P_i \tilde{Q}_{x_i} P_i$*, such that*

$$||x_i(t+1) - \hat{x}_i(t+1)||^2_{P_i} \leq ||x_i(t) - \hat{x}_i(t)||^2_{P_i} - ||x_i(t) - \hat{x}_i(t)||^2_{Q_{x_i}} + ||w_i(t)||^2_{Q_{w_i}} \quad (6.49)$$

Proof of the Theorem

Step 1: definition of δISS Lyapunov function

For this step, the same procedure described in Section 6.1.1 is applied and is not reported here for the sake of conciseness.

Step 2: Study of nonlinearity via Local Sector Condition

According to Assumption 5.1.1 and Equation (2.3), for any matrix $\Lambda_i \geq 0$ diagonal with elements $\lambda_{ij} \in [0, 1]$,

$$(\delta v_i - \delta s_i)^T W_i (\delta s_i - \lambda_i \delta v_i) \geq 0, \quad \forall W_i \in \mathbb{D}_+ \quad (6.50)$$

if v_i and $\hat{v}_i$ are sufficiently close to the origin, see [26]. In (6.50) $\mathbb{D}_+$ is the set of definite positive diagonal matrices, $\delta s_i = s_i - \hat{s}_i$, and $\delta v_i = \tilde{A}_{Kx_i} \delta x_i + \tilde{B}_{Ks_i} \delta s_i + \delta \hat{\tilde{d}}_i$.

Inequality (6.50) can be rewritten as:

$$\left(\tilde{A}_{Kx_i} \delta x_i + (\tilde{B}_{Ks_i} - I)\delta s_i + \tilde{D}_i \mathbf{w}_i\right)^T W_i \left((I - \Lambda_i \tilde{B}_{Ks_i})\delta s_i - \Lambda_i \tilde{A}_{Kx_i} \delta x_i - \Lambda_i \tilde{D}_i \mathbf{w}_i\right) \geq 0 \quad (6.51)$$

Recalling the definition of the vector $\Phi_i^T = [\delta x_i^T \quad \delta s_i^T \quad \mathbf{w}_i^T]$, we can rewrite (6.51) as

$$\Phi_i^T \begin{bmatrix} \tilde{A}_{Kx_i}^T \\ \tilde{B}_{Ks_i}^T - I \\ \tilde{D}_i^T \end{bmatrix} W_i \begin{bmatrix} -\Lambda_i \tilde{A}_{Kx_i} & I - \Lambda_i \tilde{B}_{Ks_i} & -\Lambda_i \tilde{D}_i \end{bmatrix} \Phi_i + (\star)^T \geq 0 \quad (6.52)$$

where $(\star)^T$ denotes the symmetric counterpart.

The final form of sector condition, expressed as a quadratic constraint, is the following:

$$\Phi_i^T \begin{bmatrix} -2\tilde{A}_{K_x^i}^T W_i \Lambda_i \tilde{A}_{K_x^i} & E_{21}^T & -2\tilde{A}_{K_x^i}^T \Lambda_i W_i \tilde{D}_i \\ E_{21} & E_{22} & E_{32}^T \\ -2\tilde{D}_i^T W_i \Lambda_i \tilde{A}_{Kx_i} & E_{32} & -\tilde{D}_i^T \Lambda_i \tilde{D}_i \end{bmatrix} \Phi_i \geq 0 \quad (6.53)$$

where

- $E_{21} = W_i\tilde{A}_{Kx_i} + \big(I - \tilde{B}^T_{Ks_i}\big)W_i\Lambda_i\tilde{A}_{Kx_i} - \tilde{B}^T_{Ks_i}\Lambda_i W_i\tilde{A}_{Kx_i}$
- $E_{22} = \big(\tilde{B}^T_{Ks_i} - I\big)W_i + W_i\big(\tilde{B}_{Ks_i} - I\big) + \big(I - \tilde{B}^T_{Ks_i}\big)W_i\Lambda_i\tilde{B}_{Ks_i} + \tilde{B}^T_{Ks_i}\Lambda_i W_i\big(I - \tilde{B}^T_{Ks_i}\big)$
- $E_{32} = \tilde{D}_i^T W_i - \tilde{D}_i^T W_i\Lambda_i\tilde{B}_{Ks_i} + \tilde{D}_i^T W_i\Lambda_i\big(I - \tilde{B}_{Ks_i}\big)$

Equation (6.53) can now be rewritten by isolating the terms that depend on Λ_i from those that do not, obtaining:

$$\Phi_i^T\Big(\mathbf{G} - \mathbf{M}_i(\Lambda_i)\Big)\Phi_i \geq 0 \tag{6.54}$$

where

$$\mathbf{G} = \begin{bmatrix} 0 & \tilde{A}^T_{Kx_i}W_i & 0 \\ W_i\tilde{A}_{Kx_i} & \big(\tilde{B}^T_{Ks_i} - I\big)W_i + W_i\big(\tilde{B}_{Ks_i} - I\big) & W_i\tilde{D}_i \\ 0 & \tilde{D}_i^T & 0 \end{bmatrix} \tag{6.55}$$

$$\mathbf{M}(\Lambda_i) = \begin{bmatrix} \tilde{A}^T_{Kx_i} & \tilde{A}^T_{Kx_i} \\ \tilde{B}^T_{Ks_i} - I & \tilde{B}^T_{Ks_i} \\ \tilde{D}_i^T & \tilde{D}_i^T \end{bmatrix} \begin{bmatrix} W_i\Lambda_i & 0 \\ 0 & W_i\Lambda_i \end{bmatrix} \begin{bmatrix} \tilde{A}_{Kx_i} & \tilde{B}_{Ks_i} & \tilde{D}_i \\ \tilde{A}_{Kx_i} & \tilde{B}_{Ks_i} - I & \tilde{D}_i \end{bmatrix} \tag{6.56}$$

In view of the LMI definition, we can define $\mathbf{L}_i = W_i\Lambda_i$

Step 3: Imposition of Lyapunov Inequality subject to the Local Sector Condition

As in Section 6.1.1, we impose the Lyapunov inequality subject to the local sector condition as follows:

$$||\delta x_i(t+1)||^2_{P_i} - ||\delta x_i(t)||^2_{P_i} + \Phi_i^T\big(\mathbf{G} - \mathbf{M}_i(\Lambda_i)\big)\Phi_i \leq -||\delta x_i(t)||^2_{Q_{x_i}} + ||\delta d_i(t)||^2_{Q_{w_i}} \tag{6.57}$$

where $\delta x_i, \delta d_i(t), P_i, Q_{x_i}$, and Q_{w_i} are defined as in Section 6.1.1.

After applying the same operations detailed in Step 3 of the proof in Section 6.1.1, the resulting LMI coincides with (6.21), up to an additional term that explicitly accounts for the degree of freedom introduced by Λ_i, and thus represents the relaxation associated with the local condition. ■

6.3.2. Modification of the Offline Design Procedure using local δISS

Algorithm 6.2 can be modified using the local δISS property derived in Section 6.3.1 if a solution cannot be found according to the procedure sketched in Section 6.2.

The first step of the procedure consists in the relaxation of the δISS condition, i.e., we need to replace (6.3) with (6.47). To do so, we define a key design parameter in the tuning process, called $\beta_i > 0$. The relaxation is performed on constraint (6.21) as follows:

$$M_{\delta ISS} \geq -\beta_i \tag{6.58}$$

Algorithm 6.2 Offline Design Procedure

for $i = 1, \ldots, N$ **do**

Step 1:

Solve the LMI feasibility problem with optimization variables $\tilde{Q}_{x_i}, Q_i, Q_{w_i}, U_i, Z_i, \tilde{Z}_i, Q^0_{xu_l}$ and $\tilde{S}_i$ under constraints (6.58), (6.31), (6.32), (6.38), $\tilde{Q}_{x_i} > 0, Q_i > 0, Q_{w_i} > 0, U_i > 0, Q^0_{xu_l} > 0$ and $\tilde{S}_i > 0$

if *a solution is found* **then**

Set $Q_{x_i} = Q_i \tilde{Q}_{x_i} Q_i$, $P_i = Q_i^{-1}$, $W_i = U_i^{-1}$, $S_i = W_i \tilde{S}_i W_i$, $K_x^i = Z_i Q_i^{-1}$ and $K_s^i = \tilde{Z}_i U_i^{-1}$

Go to step 2

else

Set $\beta_{i,\text{new}} = \beta_i + 10^{-2}$ and repeat from Step 1

Step 2:

Solve the LMI feasibility problem with optimization variables $\tilde{Q}_{x_i}, Q_i, Q_{w_i}, Q^0_{xu_l}, \tilde{S}_i$ and $\mathbf{L}_i$ under constraints (6.47), (6.31), (6.32), (6.38), $\tilde{Q}_{x_i} > 0, Q_i > 0, Q_{w_i} > 0, Q^0_{xu_l} > 0, \tilde{S}_i > 0$ and $\mathbf{L}_i \geq 0$

if *a solution is found* **then**

Set $Q_{x_i} = Q_i \tilde{Q}_{x_i} Q_i$, $P_i = Q_i^{-1}$ and $S_i = W_i \tilde{S}_i W_i$

Go to step 3

else

Set $\gamma_{i,\text{new}} = \gamma_i + 1$ and repeat from step 1

Step 3:

Solve the LMI feasibility problem with optimization variable $\hat{Q}_{xu_i}$, under constraints (6.46), (6.45), and $\hat{Q}_{xu_i} > 0$

if *a solution is found* **then**

Stop

else

Set $\gamma_{i,\text{new}} = \gamma_i + 1$ and repeat from step 1

7 | Conclusions

This thesis has addressed the problem of physics-inspired scalable modeling and control design for large-scale nonlinear systems: these two components are presented in an unified framework as two complementary parts of the same problem.

In fact, in decentralized and distributed control architectures, scalability and modularity can be achieved only if the adopted model explicitly reflects the interconnection structure of the plant. In this work, a physics-inspired and control-oriented identification strategy has been developed, considering Recurrent Equilibrium Network models. The proposed identification procedure imposes structural information and physical properties, directly on the learning phase.
Different approaches, based on several decomposition techniques, have been investigated and compared in terms of trade-off between accuracy, sparsity of the interconnection, and computational complexity. The effectiveness of the proposed methodology has been assessed on a case study consisting in a nonlinear chemical plant.
The resulting REN models are able to capture the dominant dynamical couplings while significantly reducing the complexity of the problem, and at the same time they highlight the crucial role of the sampling time. In fact, the choice of the discretization step and the sampling time value directly affect the capability of neural models to faithfully reproduce the system behavior.

From the control design point of view, a robustness-based nonlinear Model Predictive Control framework has been developed for the class of structured REN models and in particular a decentralized architecture has been devised. A crucial step in the decentralized controller design is the definition of suitable LMIs, to guarantee consistency among subsystems, preservation of the δISS property, recursive feasibility, and robust constraint satisfaction.

The main outcome of this work is therefore the demonstration that physics-inspired structured neural models can be effectively used as control-oriented representations for large-scale systems, enabling the design of decentralized robust MPC schemes with guaranteed stability and constraint satisfaction. This points out the importance of integrating model-

ing and control objectives within a single framework, rather than treating them as separate and sequential steps.

Some aspects of the proposed approach could be further investigated in future research. In this work, the LMI-based design procedure has been used to derive the local stabilizing feedback gains and to guarantee the required incremental stability and consistency properties. The next step is therefore the integration of these results into a full receding-horizon optimization framework, in which the decentralized optimal control problem is solved in real time. This next implementation would assess the closed-loop performance of the proposed architecture and evaluate the computational advantages provided by the structured REN models.
Another relevant direction concerns the validation of the proposed methodology on experimental plants and the integration of online learning strategies for the REN models, so as to handle slowly varying dynamics and modeling uncertainties.

A | Appendix - Structured Modelling

A.1. Model Decomposition

In case of large-scale, multi-agent and complex systems, the use of traditional centralized control methods may be prevented in view of several challenges:

- Such systems are characterized by high dimensionality, (i.e., large number of subsystems, variables and outputs) which requires managing, storing, and processing large amount of data.
- System's components may be geographically distributed, and communication among them may introduce serious transmission delays or failures.
- Model structural variations, like plugging in or removing a subsystem, often require the complete re-design of both the model and the controller.

In order to increase flexibility and modularity in large-scale systems, it is often necessary to decompose these models into smaller, independent subproblems. The decomposition of available dynamic large-scale models is traditionally a preliminary step in the design of decentralized and distributed control methods, as presented in [14].

It is instructive to analyze the model decomposition problem in the linear case, e.g., consider a system $\mathscr{S}$ described by the following available linear discrete-time model:

$$x(t+1) = \mathbf{A}x(t) + \mathbf{B}u(t) \tag{A.1a}$$

$$y(t) = \mathbf{C}x(t) \tag{A.1b}$$

where $x \in \mathbb{R}^{n_x}$, $u \in \mathbb{R}^{n_u}$, and $y \in \mathbb{R}^{n_y}$. The main idea behind model decomposition is to partition input and output vectors in N sub-vectors, to obtain local inputs and local outputs, denoted as $u_i(t)$ and $y_i(t)$. For any adopted decomposition approach, the aim of the partition phase consists of defining, for each subsystem $\mathscr{S}_i$, the corresponding

state-space model, having the general form shown below.

$$x_i(t+1) = A_{ii}x_i(t) + B_{ii}u_i(t) + \sum_{j\neq i}\big(A_{ij}x_j(t) + B_{ij}u_j(t)\big) \tag{A.2a}$$

$$y_i(t) = C_{ii}x_i(t) + \sum_{i\neq j} C_{ij}x_j(t) \tag{A.2b}$$

Based on the system's physics and the nature of interconnection between variables three different types of models can be distinguished:

- **Non-overlapping decomposition:** in this case, the global state vector x is partitioned into N non-overlapping sub-vectors x_i, with $\sum_{i=1}^{N} n_{xi} = n_x$. Then the original system can be rewritten as:

$$\begin{bmatrix} x_1(t+1) \\ \vdots \\ x_N(t+1) \end{bmatrix} = \begin{bmatrix} A_{11} & \cdots & A_{1N} \\ \vdots & \ddots & \vdots \\ A_{N1} & \cdots & A_{NN} \end{bmatrix} \begin{bmatrix} x_1(t) \\ \vdots \\ x_N(t) \end{bmatrix} + \begin{bmatrix} B_{11} & \cdots & B_{1N} \\ \vdots & \ddots & \vdots \\ B_{N1} & \cdots & B_{NN} \end{bmatrix} \begin{bmatrix} u_1(t) \\ \vdots \\ u_N(t) \end{bmatrix} \tag{A.3a}$$

$$\begin{bmatrix} y_1(t) \\ \vdots \\ y_N(t) \end{bmatrix} = \begin{bmatrix} C_{11} & \cdots & C_{1N} \\ \vdots & \ddots & \vdots \\ C_{N1} & \cdots & C_{NN} \end{bmatrix} \begin{bmatrix} x_1(t) \\ \vdots \\ x_N(t) \end{bmatrix} \tag{A.3b}$$

  In this configuration, couplings among subsystems are null or almost negligible (i.e., for $i \neq j$, A_{ij}, B_{ij}, C_{ij} should be zero or the as small as possible). Although the order of each subsystem $\mathscr{S}_i$ is minimal and requires a lower computational cost, this representation does not capture all coupling dynamics between subsystems. Hence, this decomposition may have some problems in representing connections among subsystems, but is locally robust and effective, since it reduces the probability of network-induced issues.

- **Completely overlapping decomposition:** in this case the coupling strength between subsystems is relevant, and —unlike the non-overlapping approach—the connections between subsystems are not neglected.
  This decomposition can be obtained from Equations (A.1a) and (A.4b). First, matrix $\mathbf{B}$ and input vector $u(t)$ can be decomposed into N blocks, resulting in $\mathbf{B} = [B_1 \mid B_2 \mid ... \mid B_N]$, and $u(t) = [u_1(t) \quad u_2(t) \quad ... \quad u_N(t)]^T$. Similarly, for outputs, the decomposition is $\mathbf{C} = [C_1 \mid C_2 \mid ... \mid C_N]^T$.

The decomposed model equations are:

$$x(t+1) = \mathbf{A}x(t) + B_1u_1(t) + \ \ldots \ + B_Nu_N(t) \tag{A.4a}$$

$$y_i(t) = C_ix_i(t) \tag{A.4b}$$

where $u_i(t), y_i(t)$ are the input and the output of the i-th channel, respectively.
This decomposition ensured the highest descriptive capabilities, allowing the representation to be as close as possible to the real system. However, such a non-approximated representation requires a dense network of interconnections and requires the identification of all transfer functions, i.e, all input-output relationships. As a result, the system order increases, leading to a high computational burden and a reduction in structural sparsity and scalability.

- **Partially overlapping decomposition:** this approach represents a middle ground between the previous two. It accounts for shared dynamics between subsystems, neglecting only irrelevant connections, thus improving model accuracy while preventing unnecessary complexity in large-scale settings. It is particularly suitable when subsystem interactions are significant, since it balances the trade-off between modularity and accuracy: the values assigned to matrices A_{ij}, B_{ij}, C_{ij} (for $i \neq j$) ensure that the coupling between subsystems is sufficiently captured to maintain realistic dynamics, while the suppression of unnecessary connections prevents the excessive growth of the state-space dimension.

In practical applications, the decomposition can be implemented by modeling the couplings either through inputs or through states. Input-coupled representations correspond to completely overlapping decompositions, since shared dynamics are introduced through external inputs. Alternatively, achieving a non-overlapping decomposition requires a state-coupled formulation, where each subsystem evolves based only on its own state, and interactions are removed or absorbed into internal dynamics.

In this work, the state-coupled (non-overlapping) approach is adopted to examine the sparsity structure of the system in the case study and to reduce computational load.

Remark that this analysis focuses on the linear models, where decomposition and coupling structures can be explicitly represented through matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$. Nevertheless, the same concepts can be extended to nonlinear systems.
In this work, the decomposition is formulated in terms of RNN structures, where each subsystem is modeled by a local nonlinear function. This generalization preserves the advantages of modularity and scalability while enabling the representation of complex,

nonlinear interactions — a crucial aspect when moving toward Recurrent Neural Networks and physics-based architectures.

A.2. The effect of the Sampling Time and Discretization in structured models

In real applications, physical system models are usually defined in continuous time, but many recent control synthesis techniques are specifically designed for discrete-time systems. For example, for large-scale and sparse systems, the MPC synthesis methods are developed commonly in discrete-time. This makes the choice of appropriate discretization methods crucial [15] to preserve fundamental properties like sparsity and stability of the continuous-time systems. At the same time, however, a fundamental aspect is also the proper selection of the sampling time T_s.

From a learning perspective, a large T_s reduces the informativeness of the input-output sequences, used for training. In a discrete-time domain, the input-output profiles present a piecewise-constant behavior, hence, any variations between two sampling instants is not captured. When training RNNs to reproduce the system behavior, if T_s is too large, the real fast dynamics are not captured: the algorithm may introduce artificial, usually oscillating, modes that don't corresponds to real ones. Conversely, choosing a very small sampling time increases computational cost and the risk of overfitting to redundant data or noise.

From the model structure perspective, the discretization introduces a change in the structure imposed to the model. For clarity and simplicity, consider a continuous-time (^{ct}), autonomous, linear state-space representation of a system $\mathscr{S}_i$ decomposed into N non-overlapping subsystems.

$$\begin{bmatrix} \dot{x}_1(t) \\ \vdots \\ \dot{x}_N(t) \end{bmatrix} = \begin{bmatrix} A_{11}^{ct} & \cdots & A_{1N}^{ct} \\ \vdots & \ddots & \vdots \\ A_{N1}^{ct} & \cdots & A_{NN}^{ct} \end{bmatrix} \begin{bmatrix} x_1(t) \\ \vdots \\ x_N(t) \end{bmatrix} + \begin{bmatrix} B_{11}^{ct} & \cdots & B_{1N}^{ct} \\ \vdots & \ddots & \vdots \\ B_{N1}^{ct} & \cdots & B_{NN}^{ct} \end{bmatrix} \begin{bmatrix} u_1(t) \\ \vdots \\ u_N(t) \end{bmatrix} \tag{A.5a}$$

$$\begin{bmatrix} y_1(t) \\ \vdots \\ y_N(t) \end{bmatrix} = \begin{bmatrix} C_{11}^{ct} & \cdots & C_{1N}^{ct} \\ \vdots & \ddots & \vdots \\ C_{N1}^{ct} & \cdots & C_{NN}^{ct} \end{bmatrix} \begin{bmatrix} x_1(t) \\ \vdots \\ x_N(t) \end{bmatrix} \tag{A.5b}$$

This continuous-time system is discretized, with a sampling time T_s into corresponding

discrete-time model, resulting in:

$$\begin{bmatrix} x_1(t+1) \\ \vdots \\ x_N(t+1) \end{bmatrix} = \begin{bmatrix} A_{11} & \cdots & A_{1N} \\ \vdots & \ddots & \vdots \\ A_{N1} & \cdots & A_{NN} \end{bmatrix} \begin{bmatrix} x_1(t) \\ \vdots \\ x_N(t) \end{bmatrix} + \begin{bmatrix} B_{11} & \cdots & B_{1N} \\ \vdots & \ddots & \vdots \\ B_{N1} & \cdots & B_{NN} \end{bmatrix} \begin{bmatrix} u_1(t) \\ \vdots \\ u_N(t) \end{bmatrix} \tag{A.6a}$$

$$\begin{bmatrix} y_1(t) \\ \vdots \\ y_N(t) \end{bmatrix} = \begin{bmatrix} C_{11} & \cdots & C_{1N} \\ \vdots & \ddots & \vdots \\ C_{N1} & \cdots & C_{NN} \end{bmatrix} \begin{bmatrix} x_1(t) \\ \vdots \\ x_N(t) \end{bmatrix} \tag{A.6b}$$

When the continuous-time system matrices exhibit a structure that reflects the interconnections between subsystems, discretization may lead to a loss of this structure. In particular, the exact i.e., the Zero-Order Hold (ZOH) discretization can introduce non-zero entries in the discrete-time matrices that correspond to zeros in the continuous-time representation. In other words it can happen that $A_{ij}^{ct} = 0$, but $A_{ij} \neq 0$, after the discretization.

The only way to make the discrete-time system to inherit the same structure of the to continuous-time system is to select approximated methods (e.g. the mixed Euler-ZOH method proposed in [15]), whose approximation is however limited only for small values of the sampling time T_s.
In this thesis, we consider the structured modelling issue from aa different perspective, where a structure is directly imposed on the discrete-time matrices before performing the identification. Nevertheless, the choice of the sampling time remains critical, and, as also discussed in the thesis, the sampling time must be properly selected and suficiently small, to reduce the error induced by the model structure of the identified discrete-time model.

Bibliography

[1] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4 (11), 2018.

[2] L. Bakule. Decentralized control: Status and outlook. *Annual Reviews in Control*, 38(1):71–80, 2014.

[3] I. A. Basheer and M. Hajmeer. Artificial neural networks: fundamentals, computing, design, and application. *Journal of microbiological methods*, 43(1):3–31, 2000.

[4] F. Bayer, M. Bürger, and F. Allgöwer. Discrete-time incremental iss: A framework for robust nmpc. In *2013 European control conference (ECC)*, pages 2068–2073. IEEE, 2013.

[5] 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. 2017.

[6] F. Bonassi, M. Farina, and R. Scattolini. Stability of discrete-time feed-forward neural networks in narx configuration. *IFAC-PapersOnLine*, 54(7):547–552, 2021.

[7] 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*, 114:92–104, 2022.

[8] 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)*, pages 799–804. IEEE, 2019.

[9] M. Cilimkovic. Neural networks and back propagation algorithm. *Institute of Technology Blanchardstown, Blanchardstown Road North Dublin*, 15(1):18, 2015.

[10] A. Cotorruelo, I. Kolmanovsky, and E. Garone. A sum-of-squares-based procedure to approximate the pontryagin difference of basic semi-algebraic sets. *Automatica*, 135:109783, 2022.

[11] W. D'Amico, A. La Bella, and M. Farina. An incremental input-to-state stability condition for a class of recurrent neural networks. *IEEE Transactions on Automatic Control*, 69(4):2221–2236, 2023.

[12] W. D'Amico, A. La Bella, and M. Farina. Data-driven control of echo state-based recurrent neural networks with robust stability guarantees. *Systems & Control Letters*, 195:105974, 2025.

[13] M. Farina and R. Scattolini. Distributed predictive control: A non-cooperative algorithm with neighbor-to-neighbor communication for linear systems. *Automatica*, 48 (6):1088–1096, 2012.

[14] M. Farina and R. Scattolini. Distributed mpc for large-scale systems. In *Handbook of Model Predictive Control*, pages 239–258. Springer, 2018.

[15] M. Farina, P. Colaneri, and R. Scattolini. Block-wise discretization accounting for structural constraints. *Automatica*, 49(11):3411–3417, 2013.

[16] E. Fogel, D. Halperin, and R. Wein. Minkowksi sums and offset polygons. In *CGAL Arrangements and Their Applications: A Step-by-Step Guide*, pages 209–240. Springer, 2011.

[17] M. T. Hagan and H. B. Demuth. Neural networks for control. In *Proceedings of the 1999 American control conference (cat. No. 99CH36251)*, volume 3, pages 1642–1656. IEEE, 1999.

[18] M. Kordestani, A. A. Safavi, and M. Saif. Recent survey of large-scale systems: Architectures, controller strategies, and industrial applications. *IEEE Systems Journal*, 15(4):5440–5453, 2021.

[19] F. Liang. *Distributed control of large scale systems.* Acta Automatica Sinica, 1990.

[20] D. Limon, I. Alvarado, T. Alamo, and E. Camacho. On the design of robust tube-based mpc for tracking. *IFAC Proceedings Volumes*, 41(2):15333–15338, 2008.

[21] T. Lin, B. G. Horne, P. Tino, and C. L. Giles. Learning long-term dependencies in narx recurrent neural networks. *IEEE transactions on neural networks*, 7(6):1329–1338, 1996.

[22] M. Maccagni. Robust and decentralized model predictive control for recurrent neu ral network models.

[23] D. Q. Mayne, M. M. Seron, and S. V. Raković. Robust model predictive control of

constrained linear systems with bounded disturbances. *Automatica*, 41(2):219–224, 2005.

[24] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. Pmlr, 2013.

[25] D. Ravasio, M. Farina, and A. Ballarino. Lmi-based design of a robust model predictive controller for a class of recurrent neural networks with guaranteed properties. *IEEE Control Systems Letters*, 8:1126–1131, 2024.

[26] D. Ravasio, M. Farina, A. L. Bella, and A. Ballarino. Recurrent neural network-based robust control systems with closed-loop regional incremental iss and application to mpc design, 2025. URL `https://arxiv.org/abs/2506.20334`.

[27] M. Revay, R. Wang, and I. R. Manchester. A convex parameterization of robust recurrent neural networks. *IEEE Control Systems Letters*, 5(4):1363–1368, 2020.

[28] M. Revay, R. Wang, and I. R. Manchester. Recurrent equilibrium networks: Unconstrained learning of stable and robust dynamical models. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 2282–2287. IEEE, 2021.

[29] M. Revay, R. Wang, and I. R. Manchester. Recurrent equilibrium networks: Flexible dynamic models with guaranteed stability and robustness. *IEEE Transactions on Automatic Control*, 69(5):2855–2870, 2023.

[30] S. Riverso, M. Farina, and G. Ferrari-Trecate. Plug-and-play decentralized model predictive control for linear systems. *IEEE Transactions on Automatic Control*, 58 (10):2608–2614, 2013.

[31] R. M. Schmidt. Recurrent neural networks (rnns): A gentle introduction and overview. *arXiv preprint arXiv:1912.05911*, 2019.

[32] D. D. Siljak. *Decentralized control of complex systems*. Courier Corporation, 2013.

[33] B. T. Stewart, A. N. Venkat, J. B. Rawlings, S. J. Wright, and G. Pannocchia. Cooperative distributed model predictive control. *Systems & Control Letters*, 59(8): 460–469, 2010.

[34] D. Svozil, V. Kvasnicka, and J. Pospichal. Introduction to multi-layer feed-forward neural networks. *Chemometrics and intelligent laboratory systems*, 39(1):43–62, 1997.

[35] S. Yu, C. Maier, H. Chen, and F. Allgöwer. Tube mpc scheme based on robust

control invariant set with application to lipschitz nonlinear systems. *Systems & Control Letters*, 62(2):194–200, 2013.

List of Figures

1 Comparison of control architectures: centralized, decentralized, and distributed. 1

2.1 REN as a feedback interconnection of a linear system G and a nonlinear activation σ 19
2.2 Function $\sigma_i(v_i) = tanh(v_i)$ 21

3.1 Two reactors in series with separator and recycle. 33
3.2 Plots of simulated inputs 37
3.3 Plots of non-scaled simulated outputs 38
3.4 Plots of concentrations' sum for each subsystem 38
3.5 Comparison between the simulated output (——) and the REN model output (——), under low-frequency input signals. 41
3.6 Comparison between the simulated output (——) and the REN model output (——), under high-frequency input signals. 42
3.7 Comparison between the simulated output (——), the REN model output (validation from simulated data) (——), and the REN model output (validation from predictions) (——), under low-frequency input signals. . 45
3.8 Comparison between the simulated output (——), the REN model output (validation from simulated data) (——), and the REN model output (validation from predictions) (——), under high-frequency input signals. . 46
3.9 Comparison between the simulated output (——), and the REN model output with $T_s = 0.01s$ (——) and with $T_s = 0.1s$ (——), under low-frequency input signals. 48
3.10 Comparison between the simulated output (——), and the REN model output with $T_s = 0.01s$ (——) and with $T_s = 0.1s$ (——), under high-frequency input signals. 49
3.11 Chemical plant interconnection structure 51
3.12 Chemical plant output matrices structure 52

3.13 Comparison between the simulated output (——), the REN model output (structured matrices) (——), under low-frequency input signals. 53
3.14 Comparison between the simulated output (——), the REN model output (structured matrices) (——), under high-frequency input signals. 54

List of Tables

3.1 Chemical Plant System's Parameters 35
3.2 Chemical Plant Input constraints 35
3.3 Comparison between models obtained from Individual Subsystem Identification (with validation using predicted outputs) for different sampling times 50
3.4 Comparison between models: Centralized, Individual (with validation using output data and validation using simulated outputs), and Structured . 56