Master's thesis

OPEN- AND CLOSED-LOOP IDENTIFICATION USING ARTIFICIAL NEURAL NETWORKS

Kevin Herman Muraro Gularte

Brasília
2017, December

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

Master's thesis

OPEN- AND CLOSED-LOOP IDENTIFICATION USING ARTIFICIAL NEURAL NETWORKS

Kevin Herman Muraro Gularte

Report submitted to the Department of Mechanical Engineering in partial fulfillment of the requirements for the degree of Master in Mechatronic Systems

FICHA CATALOGRÁFICA

GULARTE, KEVIN HERMAN MURARO
Open- and Closed-loop Identification Using Artificial Neural Networks

[Distrito Federal] 2017.

x, 149p., 210X297 mm (ENM/FT/UnB, Sistemas Mecatrônicos, 2017). Dissertação de Mestrado – Universidade de Brasília. Faculdade de Tecnologia. Departamento de Engenharia Mecânica.

1. Identificação	2. Observação
3. Controle	4. Lyapunov
5. Redes Neurais	6. Transiente
I. ENM/FT/UnB	

REFERÊNCIA BIBLIOGRÁFICA

GULARTE, K. H. M., (2017). *Open- and Closed-loop Identification Using Artificial Neural Networks*. Dissertação de Mestrado em Sistemas Mecatrônicos, Publicação ENM.DM-133/2017, Departamento de Engenharia Mecânica, Faculdade de Tecnologia, Universidade de Brasília, Brasília, DF, 149p.

CESSÃO DE DIREITOS

AUTOR: Kevin Herman Muraro Gularte

TÍTULO: *Open- and Closed-loop Identification Using Artificial Neural Networks*.

GRAU: Mestre ANO: 2017

Kevin Herman Muraro Gularte
Departamento de Eng. Mecânica (ENM) – FT
Universidade de Brasília (UnB)
Campus Darcy Ribeiro
CEP 70919-970 - Brasília - DF – Brasil.

ABSTRACT

This work presents several schemes for online identification, observation, and adaptive control of uncertain nonlinear systems by using artificial neural networks in which the transient and residual errors can be independently adjusted. Based on Lyapunov theory, and using results already available in adaptive control theory, schemes for identification, observation, and control are proposed. However, unlike other works in the literature, the identification model, learning algorithm, and control laws are designed to decouple the transient and residual state performance, which is accomplished through the manipulation of independent design parameters.

Initially, the case of online identification is considered, since black-box systems can be parameterized by neural networks and these parameterizations are needed to tackle more complex problems, such as observation and adaptive control. The proposed identifier has the following peculiarities: 1) Possibility to control the size of residual state error from design matrices; 2) Possibility to adjust the duration of the transient regime from a design parameter regardless of the size of the regime error. The identification of a chaotic system of three states was considered to apply the scheme.

The result is then extended to the case where some states are not available for measurement. To do so, it is only necessary to make some adjustments in the identification scheme. Basically, the state error is placed as a function of the output error, which is available for measurement. The main characteristic of the proposed observer is the preservation of the properties of the proposed identifier. The observation of a Rössler system was implemented in order to exemplify this observer.

In the sequence, the case of control with state feedback is considered. For this, a controller was proposed using the open loop identification case as analogue to it. The main peculiarities of the identifier also occur in the controller. Finally, in order to emphasize the applicability and relevance of the proposed algorithms, the identification and control of a welding system were performed.

IDENTIFICAÇÃO DE SISTEMAS NÃO LINEARES EM MALHA ABERTA E FECHADA USANDO REDES NEURAIS ARTIFICIAIS

RESUMO ESTENDIDO

Este trabalho apresenta vários esquemas para identificação, observação e controle adaptativos em tempo real de sistemas não lineares incertos usando redes neurais artificiais. Com base na teoria de estabilidade de Lyapunov, e usando resultados já disponíveis na teoria de controle adaptativo, são propostos esquemas para identificação, observação e controle nos quais os erros de identificação, observação e rastreamento estão relacionados com parâmetros de projeto que podem ser ajustados diretamente pelo usuário. Entretanto, ao contrário das propostas usuais na literatura, este trabalho propõe algoritmos nos quais o desempenho transiente e em regime podem ser desacoplados e ajustados independentemente através de parâmetros de projeto independentes.

Inicialmente, o caso de identificação em tempo real é considerado, uma vez que cada vez mais a resolução de sistemas caixa preta tem sido demandados. O identificador proposto apresenta as seguintes peculiaridades: 1) Possibilidade de controlar o tamanho do erro residual de estado a partir de matrizes de projeto; 2) Possibilidade de ajustar a duração do regime transiente a partir de um parâmetro de projeto que é independente do tamanho do erro em regime. A identificação de um sistema caótico de três estados foi considerada para validar o esquema.

A seguir, o resultado é estendido para o caso nos quais alguns estados não estão disponíveis para medida. Para tanto, é necessário apenas fazer alguns ajustes no esquema de identificação proposto, para permitir agora estimar um ou mais estados não disponíveis para medição, a partir das entradas e saídas ao sistema. O observador proposto apresenta as mesmas peculiaridades do identificador. A observação de um sistema de Rössler foi implementada de forma a exemplificar este observador.

Na sequência, considera-se o caso de controle com realimentação do estado. Para tanto, se propôs o projeto do controlador empregando como base o caso de identificação de malha aberta. As principais peculiaridades do identificador ocorrem também no controlador. Finalmente, de modo a ressaltar a aplicabilidade e relevância dos algoritmos propostos, a identificação e controle de um sistema de soldagem foram realizados.

A dissertação está organizada da seguinte forma. O capítulo 1 apresenta a introdução, motivação, objetivo, possíveis contribuições e estrutura do trabalho proposto. No capítulo 2 é apresentada uma revisão do estado da arte dos métodos de identificação, observação e controle baseados em redes neurais artificiais.

No capítulo 3, usando a teoria de estabilidade de Lyapunov, propõe-se um esquema de identificação neural adaptativo em tempo real para uma classe de sistemas não lineares na presença de distúrbios limitados. É importante ressaltar que nenhum conhecimento prévio sobre a dinâmica do erro de aproximação, pesos ideais ou perturbações externas é necessário. Mostra-se que o algoritmo de aprendizado baseado na teoria de estabilidade de Lyapunov leva o estado estimado a convergir

assintoticamente para o estado de sistemas não lineares. O algoritmo proposto permite: 1) reduzir o erro residual de estimação de estado para valores pequenos por meio de matrizes de projeto; 2) controlar o tempo de transiente de maneira arbitraria a partir de um parâmetro de projeto. Foram feitas simulações para um sistema caótico de 3 estados e para um sistema hipercaótico de 4 estados para demonstrar a eficácia e a eficiência do algoritmo de aprendizado proposto. Nessas simulações foram feitas análises do tamanho do erro residual de estado e da escolha do tempo de transiente. Finaliza-se o capítulo com uma aplicação: a identificação neural de um sistema caótico de soldagem na qual se analisou o ajuste do tamanho do erro residual de estado.

Posteriormente, no capítulo 4, os resultados obtidos no capítulo anterior são estendidos para um sistema de observação neural. O caso de observação ocorre quando nem todos os estados estão disponíveis e um ou mais estados precisam ser estimados. A metodologia de projeto do algoritmo de aprendizado é semelhante ao caso do capítulo 3, sendo necessário fazer algumas adaptações próprias para um esquema de observação. Mais espepecificamente, a ideia principal consiste em expressar o erro de estimação de estado, que não é mais disponível para medida, em função do erro de estimação da saída. Para tanto, faz-se necessária a imposição de uma hipótese de detectabilidade e de uma outra condição matricial que devem ser satisfeitas simultaneamente para que o esquema de observação apresente características de estabilidade e convergência semelhantes ao esquema de identificação proposto. Realiza-se no final do capítulo a observação de um sistema caótico de Rössler sob a presença de distúrbios externos com controle do tempo de transiente.

No capítulo 5, os resultados do capítulo 3 são estendidos para controlar sistemas não lineares afins no controle. O caso de controle ocorre quando se realiza uma identificação em malha fechada, ou seja, há uma realimentação no sistema. Mais exatamente, através da realimentação objetiva-se cancelar as não lineares desconhecidas no sistema que podem ser parametrizadas por uma rede neural artificial. Dessa maneira, a equação de erro de restreamente pode ser reescrita com uma estrutura similar à equação de erro de estimação do caso de identificação. Com a finalidade de ressaltar a aplicabilidade do esquema de controle proposto, para situações de interesse industrial, realiza-se a simulação do controle de um sistema de soldagem com tranferência globular-spray em um processo GMAW (Soldagem por arco elétrico com gás de proteção). Finalmente, no capítulo 6 resume-se as contribuições da pesquisa, os resultados obtidos, e sugestões para pesquisas futuras são discutidas.

A fundamentação teórica das redes neurais artificiais (incluindo suas propriedades), dos algoritmos de aprendizado e da teoria de estabilidade de Lyapunov são descritas no apêndice 1, assim como outras informações importantes que embasam os capítulos do trabalho. O apêndice 2 contém os códigos utilizados para implementacçaão do identificador, observador e controlador propostos nesta dissertação.

Table of Contents

1 Introduction **1**
- 1.1 Motivation of the Thesis 1
- 1.2 Thesis Objectives 2
- 1.2.1 General objective 2
- 1.2.2 Specific objectives 2
- 1.3 Thesis Contributions 3
- 1.3.1 Main Contribution 3
- 1.3.2 Other Contributions 3
- 1.4 Thesis Overview 3

2 Literature Review **5**

3 Online Neuro Identification of Uncertain Systems With Control of Residual Error and Transient Time **11**
- 3.1 Problem Formulation 11
- 3.2 Identification Model and State Estimate Error Equation 12
- 3.3 Adaptive Laws and Stability Analysis 14
- 3.4 Simulation 21
- 3.4.1 Chaotic System 21
- 3.4.2 Hyperchaotic Finance System 27
- 3.4.3 Hyperchaotic Welding System 33
- 3.5 Summary 40

4 Adaptive Observer Design of Uncertain Systems With Control of Residual Error **41**
- 4.1 Problem Formulation 41
- 4.2 Adaptive Observer 42
- 4.3 Adaptive Laws and Stability Analysis 44
- 4.4 Simulation 51
- 4.4.1 Rösller oscillator system 51
- 4.5 Summary 57

5 Neural Controller in Closed Loop with Control of Residual Error and Transient Time: An application in Welding **59**

5.1 Problem Formulation 60
5.2 Controller Model and Tracking Equation 60
5.3 Adaptive Laws and Stability Analysis 63
5.4 Simulation 69
5.4.1 Welding System 69
5.5 Summary 78

6 Conclusions 79

References 81

Appendix 90

I Technical Background 91
I.1 Motivation 91
I.2 Mathematical Preliminaries 91
I.2.1 Vector Norms 91
I.2.2 Matrix Norms 91
I.2.3 Lyapunov Stability Theory 92
I.3 Artificial Neural Networks 95
I.3.1 Biological Neural Networks 95
I.3.2 Artificial Neural Models 96
I.3.3 Linearly Parameterized Neural Networks 97
I.3.4 Neural Network Structures 98
I.3.5 Sigmoidal functions 99
I.4 Systems Identification 99
I.4.1 Theoretical and Experimental Modeling 99
I.4.2 Offline and Online Identification 101
I.5 Transient State, Steady-State, and Unsteady-State Response 102

II Codes 103
II.1 Simulink plant used for simulations corresponding to Figures 3.1 - 3.18 103
II.2 Code for plant model corresponding to Figures 3.1-3.8 103
II.3 Code for identifier corresponding to Figures 3.1, 3.3, 3.5 and 3.7 104
II.4 Code for identifier corresponding to Figures 3.2, 3.4, 3.6 and 3.8 106
II.5 Code to plot the Figures 3.1, 3.3, 3.5 and 3.7 107
II.6 Code to plot the Figures 3.2, 3.4, 3.6 and 3.8 110
II.7 Code for plant model corresponding to Figures 3.9 - 3.18 112
II.8 Code for identifier corresponding to Figures 3.9, 3.11, 3.13, 3.15 and 3.17 114
II.9 Code for identifier corresponding to Figures 3.10, 3.12, 3.14, 3.16 and 3.18 115

II.10 Code to plot the Figures 3.9, 3.11, 3.13, 3.15 and 3.17 117
II.11 Code to plot the Figures 3.10, 3.12, 3.14, 3.16 and 3.18 120
II.12 Simulink plant used for simulations corresponding to Figures 3.19 - 3.29 123
II.13 Code for the welding system plant corresponding to Figures 3.19-3.29 123
II.14 Code for identifier corresponding to Figures 3.19 - 3.29 125
II.15 Code to plot the Figures 3.19 - 3.29 126
II.16 Simulink plant used for simulations corresponding to Figures 4.1 - 4.8 130
II.17 Code for plant model corresponding to Figures 4.1 - 4.8 130
II.18 Code for observer corresponding to Figures 4.1, 4.3, 4.5 and 4.7 132
II.19 Code for observer corresponding to Figures 4.1, 4.3, 4.5 and 4.7 133
II.20 Code to plot the Figures 4.1, 4.3, 4.5 and 4.7 135
II.21 Code to plot the Figures 4.2, 4.4, 4.6 and 4.8 138
II.22 Simulink plant used for simulations corresponding to Figures 5.2 - 5.14 141
II.23 Code for the welding system plant corresponding to Figures 5.2-5.14 141
II.24 Code for controller corresponding to Figures 5.2 - 5.14 142
II.25 Code for derivative 144
II.26 Code to plot the Figures 5.2 - 5.14 145

LIST OF FIGURES

3.1 Performance in the estimation of x_1 when $\gamma_W = 0.5$ 22
3.2 Performance in the estimation of x_1 when $\gamma_W = 15$ 23
3.3 Performance in the estimation of x_2 when $\gamma_W = 0.5$ 23
3.4 Performance in the estimation of x_2 when $\gamma_W = 15$ 24
3.5 Performance in the estimation of x_3 when $\gamma_W = 0.5$ 24
3.6 Performance in the estimation of x_3 when $\gamma_W = 15$ 25
3.7 Frobenius norm of the estimated weight matrix W when $\gamma_W = 0.5$ 25
3.8 Frobenius norm of the estimated weight matrix W when $\gamma_W = 15$ 26
3.9 Performance in the estimation of x_1 when $\gamma_W = 0.5$ 28
3.10 Performance in the estimation of x_1 when $\gamma_W = 20$ 28
3.11 Performance in the estimation of x_2 when $\gamma_W = 0.5$ 29
3.12 Performance in the estimation of x_2 when $\gamma_W = 20$ 29
3.13 Performance in the estimation of x_3 when $\gamma_W = 0.5$ 30
3.14 Performance in the estimation of x_3 when $\gamma_W = 20$ 30
3.15 Performance in the estimation of x_4 when $\gamma_W = 0.5$ 31
3.16 Performance in the estimation of x_4 when $\gamma_W = 20$ 31
3.17 Frobenius norm of the estimated weight matrix W when $\gamma_W = 0.5$ 32
3.18 Frobenius norm of the estimated weight matrix W when $\gamma_W = 20$ 32
3.19 Input 1 35
3.20 Input 2 35
3.21 Performance in the estimation of x_1 36
3.22 Performance in the estimation of x_2 36
3.23 Performance in the estimation of x_3 37
3.24 Performance in the estimation of x_4 37
3.25 Performance in the estimation of x_1 - Monolog graph 38
3.26 Performance in the estimation of x_2 - Monolog graph 38
3.27 Performance in the estimation of x_3 - Monolog graph 39
3.28 Performance in the estimation of x_4 - Monolog graph 39
3.29 Frobenius norm of the estimated weight matrix W when $\gamma_W = 0.5$ 40

4.1 Performance in the estimation of x_1 when $\gamma_W = 0.5$ 53
4.2 Performance in the estimation of x_1 when $\gamma_W = 8$ 54
4.3 Performance in the estimation of x_2 when $\gamma_W = 0.5$ 54
4.4 Performance in the estimation of x_2 when $\gamma_W = 8$ 55

4.5 Performance in the estimation of x_3 when $\gamma_W = 0.5$ 55
4.6 Performance in the estimation of x_3 when $\gamma_W = 8$ 56
4.7 Frobenius norm of the estimated weight matrix W when $\gamma_W = 0.5$ 56
4.8 Frobenius norm of the estimated weight matrix W when $\gamma_W = 8$ 57

5.1 Filter 71
5.2 Filtered Reference 1 71
5.3 Filtered Reference 2 72
5.4 Filtered Reference 3 72
5.5 Performance in the estimation of x_1 73
5.6 Performance in the estimation of x_2 73
5.7 Performance in the estimation of x_3 74
5.8 Performance in the estimation of x_1 - Monolog graph 74
5.9 Performance in the estimation of x_2 - Monolog graph 75
5.10 Performance in the estimation of x_3 - Monolog graph 75
5.11 Control Input 1 76
5.12 Control Input 1 - time varying between [0.5,8] 76
5.13 Control Input 2 77
5.14 Control Input 3 77

I.1 Biological Neuron Scheme [1] 96
I.2 Nonlinear model of a neuron [2] 97
I.3 Multilayer Perceptron [1] 99
I.4 Different kinds of mathematical models [3] 100
I.5 Different setups for the data processing as part of the identification [3] 102

LIST OF TABLES

I.1 Common Sigmoidal Activation Functions for MLP Networks 99

LIST OF SYMBOLS

V	: Lyapunov function candidate
W^*	: matrix of optimal weights
$\hat{W}$	: estimation of ideal weight matrix W^*
$\tilde{W}$	: estimation error of ideal weight matrix W^*
x	: n-dimensional state vector
$\hat{x}$	: estimation of the n-dimensional state vector
$\tilde{x}$	: estimation error of the n-dimensional state vector
u	: m-dimensional admissible input vector
σ	: regressor vector
$s(.)$	: sigmoidal function
tr	: trace
sup	: supreme
max	: maximum
min	: minimum
λ_{min}	: smallest of the eigenvalues
I	: identity matrix
$\Re$	: set of real numbers

Acronyms

NNs	: Neural Networks
MLP	: Multilayer Perceptron
RBFs	: Radial Basis Function Neural Networks
LPNNs	: Linearly Parameterized Neural Networks
HONNs	: Higher-Order Neural Networks
GMAW	: Gas Metal Arc Welding

Chapter 1

Introduction

1.1 Motivation of the Thesis

Many works in the area of control of uncertain dynamic systems have been made in terms of linear models. However, this subject has already been extensively studied, and many of these models are unsuitable for generalized applications since non-linearity is much more common in nature. Furthermore, most linear models are simplifications or approximations that may have applicability limitations. For this reason, non-linear models have been increasingly demanded. Hence, the first motivation of this work is to study nonlinear models.

The models can be subdivided into white box, gray box, and black box. The white box models are those in which the parameters of a model are known. In the gray box models, we do not know all the parameters of the model. In the black box, we do not know any parameter. Historically, the study of white box models has been extensively accomplished, but also, in a growing way, the gray and black box models have been considered. The reason is that we do not normally have all the information about a system, and then, the estimation of their parameters has been frequently used. Hence, the area of systems identification has had great relevance and, in particular, the identification based on neural networks as well. A case similar to the identification problem is the observation problem, in which not all states are available for measurement, but can be estimated by using estimation techniques. Another interesting case to study is the case of control, where closed-loop identification is performed and, therefore, there is a feedback. All these cases have interesting applications that needed to be investigated. So, the second motivation of this work lies in the design of identification, observation, and control schemes.

NNs are used to approximate unknown nonlinearities in a system, since they satisfy the universal approximation condition on a compact domain, which allows unknown maps to be approximated with an arbitrary degree of precision, if a suitable structure is provided for the neural model. Neural networks have the advantage of relatively fast implementation and auto-learning, that is, the ability to rely on historical samples to learn. In addition, they are adaptive, since neural networks can be used in online applications without needing to have their architecture changed with each update. The problem in these applications lies in that the residual state error frequently depends on the structure of the network and this can present a problem. However, Lyapunov's stability theory can be used to overcome this drawback, since weight adjustment laws based on Lyapunov's direct method allow to ensure the boundedness of the estimates. In addition, a suitable choice for identification, observation, and control models, based on the Lyapunov analysis, ensures the convergence of the residual state errors to an arbitrary neighborhood of the origin. The advantage of this framework lies in that, irrespective of how the model is constructed, it is possible to have small residual approximation errors, even in the presence of bounded disturbances. Which may arise, for example, as a consequence of changes in dynamics due to faults or aging of equipment. Therefore, the motivation for the use of artificial neural networks is this work is the fact that they allow us to approximate unknown nonlinearities and via Lyapunov theory, and it is possible to ensure that the errors are bounded, even in the presence of unknowns.

In the estimation process, there is a moment when the learning process of the neural network occurs, called the transient regime, and a moment when the neural network has stabilized and there are no significant changes in the residual state error, called the permanent regime. Controlling the duration of the transient regime may be useful for practical purposes since certain controllers have a very short transient regime and may have problems of chattering, other controllers have a very long transient regime and may have bad performance. In literature, there have been few studies on this subject, therefore there is much room for new academic studies. Hence, the last motivation of this work lies in the manipulation of the transient duration, which, until now, is little explored in literature.

1.2 Thesis Objectives

1.2.1 General objective

Based on Lyapunov stability theory, develop theoretical models of online adaptive identification, observation, and control of non-linear systems using artificial neural networks, with the possibility of adjusting the residual state error and the regime transient duration from design parameters.

1.2.2 Specific objectives

- Propose and validate a neural identification algorithm and its respective adaptive learning law using Lyapunov's stability theory. This algorithm has three properties desired: 1) the residual state error is related to arbitrary design matrices, in order to allow arbitrary reduction of the residual state error; 2) the duration of the transient regime is related to a design parameter that is not related to the residual state error; 3) the model is applicable for complex non-linear cases, such as chaotic systems, and even in the presence of bounded disturbances.

- Extend the identification algorithm to the observer case, maintaining the same properties of the neural identification model.

- Extend the identification algorithm to the controller case, maintaining the same properties of the neural identification model.

- Perform several simulations to validate the theoretical schemes. It is intended to make: 1) the adaptive open-loop identification of a chaotic and a hyperchaotic systems; 2) the adaptive observation of a Rössler chaotic system; and 3) the open and closed loop identification of a welding system.

1.3 Thesis Contributions

1.3.1 Main Contribution

The main contribution of this work is the proposal of online identification, observation, and control schemes that uses artificial neural networks that allows an adjustment of the duration of the transient regime from a design parameter that is not related to the size of the residual state error.

1.3.2 Other Contributions

- To perform an application of the proposed schemes in a welding system using these online adaptive identification and control schemes that allow to reduce the size of the residual state error based on Lyapunov stability theory.
- The application of the Lyapunov stability theory to find a neural identifier, observer, and controller in which the residual state error relates to some design matrices, so as to allow its convergence to a neighborhood of the origin, even if in presence of bounded disturbances. Note that the scheme used allows the adjustment of the size of the residual state error and the duration of the transient, since the neural scheme is the same.
- It is shown from simulations that it is possible to validate the estimation of states even for chaotic and hyperchaotic systems, demonstrating the robustness of the proposed methods.

1.4 Thesis Overview

The Master's thesis is organized as follows. This chapter presents the introduction, motivation, objective, possible contributions and structure of the proposed work.In chapter 2 a review of the state of the art of identification, observation, and control methods based on artificial neural networks is presented.

In chapter 3, by using Lyapunov's stability theory, an online adaptive neural identification scheme is proposed for a class of non-linear systems in the presence of bounded disturbances. The proposed algorithm allows: 1) to reduce the residual error of state estimation to small values by means of design matrices; 2) to control the transient time arbitrarily from a design parameter. Simulations were done to demonstrate the effectiveness and efficiency of the proposed learning algorithm. Simulations were performed for a chaotic and a hyperchaotic systems to demonstrate the effectiveness and efficiency of the proposed learning algorithm. In these simulations, the size of the residual state error and the choice of the transient time were analyzed. The chapter ends with an application: the neural identification of a welding system with chaotic behavior where the size of the residual state error was analyzed.

Subsequently, in chapter 4, the results obtained in the chapter 3 are extended to a neural observation system. The observation of a chaotic Rössler system in the presence of external disturbances with control of the transient time is realized at the end of the chapter. In Chapter 5, the results of Chapter 3 are extended to a neural controller. A simulation of a chaotic welding system is carried out as an application. Chapter 6 summarizes the theoretical contributions of the research, the results obtained and suggestions for future research are also discussed.

The appendix 1 describes the theoretical basis of the artificial neural networks, learning algorithms and Lyapunov stability theory, as well as other technical backgrounds that support the chapters of this work. The appendix 2 contain the used codes for the simulations in MATLAB/SIMULINK.

Chapter 2

Literature Review

Modeling techniques are usually classified into two sets: the first is based on modeling by process physics and the second is based on identification from data, also called system identification methods [4]. The first set corresponds to a white-box modeling and the second to a black-box identification. There is still gray-box identification, which are especially interesting because they do not require the user to have a prior deep knowledge of the process, but allow the use of prior knowledge. What differentiates these types of modeling is the amount of information we have of the systems. If it is a white-box identification we can use any of the techniques, if it is a black-box modeling or gray-box it uses the technique based on the system identification method [5]. That is, in cases the parameters of a dynamic system are unknown, adaptive identifiers are designed to estimate the parameters [6].

Since most of the time in the real world we do not have the exact information of the parameters, the systems identification methods have an advantage over the traditionally used methods. The gray-box identification has received great attention from the scientific community in recent decades because of usually resulting in obtaining better models, since they use some prior knowledge, unlike the black-box identification [7]. As the quality of the model generally determines the quality of a problem resolution, thus allowing modeling a greater importance in the development of a design, more sophisticated modeling techniques for problem solving have been demanded [8].

Identification of systems is the construction of mathematical models from the input and output measurement of dynamic systems. This tool has been of great interest to several areas such as engineering, economics, physics, and chemistry [9, 10]. Neural identification schemes are important to predict the behavior of dynamic systems as well as to provide a parameterization when the model is uncertain. Key applications include state estimation and control systems with nonlinearities. In 1956, the term system identification was first employed by Zadeh for the problem of identifying a black-box model by its input-output relationship [11]. In sequence, many researches have shown interest in system identification.

The study of Artificial Neural Networks (ANNs) belongs to the area of Artificial Intelligence and was inspired by biological nerve cells [12]. The first studies on ANNs happened in 1943 with the publication of [13]. The ANNs have allowed important advances in the intelligent systems development, being used to solve a set of problems such as prediction and pattern recognition [14]. Some characteristics of artificial neural networks have been shown to be desirable for nonlinear systems models since they are structures adaptable to systems with great complexity, with good learning capacity and generalization power [15, 16].

In nature, systems are normally nonlinear. Usually we use control methods that approximated nonlinear systems to linear systems. In some cases linear approximations are sufficient for practical applications. However, it has been noted that there is a disadvantage in using this method since linearized systems can not fully represent some real nonlinear systems [17]. Identification systems using dynamic neural networks were first introduced in [18], which is considered a mark in terms of systems identification. Dynamic Neural Networks have the same structure as the plant, but contains neural networks with weight adjustments [18].

This area has had an increased interest due to the ability of neural networks to learn complex input-output mappings, since they are universal approximations, and by the inevitable presence of uncertainties in modeling problems, due to the simplifications imposed by mathematical modeling, unexpected failures, changes in operating conditions, aging of the material, and so on. In addition, neural identification schemes are not only important to predict system behavior, but also to provide an attractive parameterization system that can be used in the synthesis of control algorithms.

Thus, from [18], the use of neural networks as a powerful tool for identification of uncertain nonlinear systems has led to several heuristic and theoretical studies as can be seen, for example, in [19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30] and their references. The scope of the identification of nonlinear systems using artificial neural networks is broad, and can be in continuous time [30, 31] or in discrete time [32, 33]. Parameter identification can be done both offline and online. Online identification has the great advantage of being able to be used for maps that have parameters that vary in time, whereas in the offline identification it is only suitable for maps with parameters that are invariant in time [10].

Over the years the discussion about the systems stability has become more important and several concepts of stability, such as exponential stability, asymptotic stability, and global stability have been defined [34]. Usually the study of systems stability theories lead to the second Lyapunov method or one of its variants [35]. The Lyapunov Stability Theory emerged with his doctoral thesis in 1892 [36].

In dynamic neural network models, their weights are adjusted using mostly backpropagation and gradient algorithms or robust modifications of them [19, 20, 18, 21, 22, 23, 24, 37, 38, 39, 40]. In the last years, feedforward ANNs are most used for nonlinear system control and identification [41, 42]. One of most popular feedforward ANNs is the multilayer perceptron (MLP), which is utilized to identify the dynamic characteristics of a nonlinear system

The MLP network is a type of feedforward neural network, since each neuron of a layer receives as input only the outputs of neurons from the previous layer. MLP networks are multilayered and generally have nonlinear activation functions on the output. However, in identification problems, these networks have usually been used with only three layers: an initial node layer, an intermediate layer (hidden layer), and the output layer [2].

The Multilayer Perceptron networks are one of the neural network architectures that stands out for its fault tolerance and its capacity for generalization, adaptability and to approximate any continuous function to almost any arbitrary value [2, 43]. In addition, another advantage is its fast convergence speed, making it a great choice for problems involving nonlinear systems [44]. Several works have made use of MLP networks in identification problems. In [45], for example, an MLP network is employed to model a nonlinear discrete system in states space. Due to its great use it has been very common in the literature to find structures derived from the MLP model.

In 1992, [30] did another approach developing a direct adaptive tracking control architecture with Gaussian Radial Basis Function (RBF) networks to compensate for the model nonlinearities. This process makes the weights of the neural networks bounded.

The most used robust modifications in neural identification present in the literature are σ, switching-σ, ε_1, parameter projection, and dead zone [19, 23, 21, 22, 20, 24], which avoid parameter drift, common phenomenon in gradient-based adaptation laws. However, most learning algorithms for neural identification now only ensure that the residual state error is proportional to the upper bounds for approximation errors, ideal weights, and disturbances.

In [20], dynamic neural networks with a gradient algorithm for weight adjustment were used to identify a general class of uncertain nonlinear systems. In this work it was assumed that: the unknown system can be exactly modeled by a neural network model, that is, the approximation errors and disturbances are identically null. This may not be verifiable because the uncertain system and the neural model are not generally correlated, which would limit the generalization capability of the algorithm. In [20], the identification of a general class of uncertain continuous dynamic systems was proposed, and a σ-modification adaptive law was used to adjust the weights of high-order recurrent neural networks in order to ensure that the state error converge to a neighborhood of origin.

Others relevant works, such as [21, 46, 47, 48] show that dead-zone adaptation, δ-rule, σ-modification, ε_1-modification and other robust modifications can be used to make the entire identification process stable in the presence of approximation error and disturbances. Although the mentioned works have had their relevance, the dependence between the design parameters and the residual error of the state, in general, is not direct. This may end up preventing arbitrary small residual state errors. In [49], the neural identification of an uncertain nonlinear system class is proposed, it being demonstrated that the residual state error and the error of weight estimation are bounded. The problem is that the upper limit of the residual state error norm and the design parameters in the identification scheme are not related. Therefore, there is the possibility of not obtaining small residual state errors.

In open loop identification, there are works like [50], in which it is possible to prove from the Lyapunov Stability Theory that there is an estimator able to relate design parameters with the residual error of state, even in the presence of disturbances. Differently from [46], the residual state error can be arbitrarily reduced, being achieved through the scaling of unknown nonlinearities, prior to the neural approximation, and selecting a neural identification model with feedback, which aims to correlate the residual state error with estimated weights.

However, the method used in [50] may have difficulties in the sense that certain parameters are related to characteristics such as transient time. In this way it is not possible to decouple transient and permanent regime performance. Thus, it would be desirable to find an identification scheme having a parameter capable of controlling the duration of the transient regime independently of the size of the residual state error, even in the presence of disturbances.

In all the previous mentioned works, we did not have the possibility to control the transient time. For example, in [51] a proposal of an identifier practically immune to disturbances was made, but the transient time was very short with no possibility of reducing this time.

On other hand, it is not always possible to measure all the states of a real system, since in real systems it depends on sensors and possibly there will be physical limitations of the sensors or high cost. Generally, it is desirable to have information of all states of a system in control applications. It is known that an observable state can be estimated through an observer using system inputs and outputs [52]. In most cases a reliable estimation of state variables that can not be measured directly is required. For this kind of problem, state observers are usually employed [53].

The first work on the observers design in linear systems was introduced in [54]. For the good functioning of the observer it is important to have a good accuracy of the states estimation. It has been generally used as parameter to show the accuracy and reliability of the observer the estimated state error [54, 55]. Observers can be used in different applications, such as [52, 56].

Luenberger Observer and Kalman Filter are the most popular linear observers [57]. An example of Kalman Filter is [58]. However, there are many applications for nonlinear systems as extended Kalman Filter [59, 60], Extended Luenberger [61], and sliding mode observers [60, 62, 63]. Most of these works have the disadvantage of requiring prior knowledge of the system nonlinearities.

The literature for linear observers may be on a saturated level, but research on observers of nonlinear systems is far from complete. One of the difficult properties to achieve in nonlinear systems is the arbitrary increase in the attraction region of the observer's stability [64]. Designing observers for nonlinear systems is considered a difficult problem, since there is no single method that works for all classes of nonlinear systems [52, 65]. Examples of observers using neural networks in the literature are extensive [66, 67, 68, 69, 70].

Adaptive observers are computational tools that allow the simultaneous estimation of the state and parameters of a dynamic system using its inputs and outputs. Its main applications include fault detection [21, 71, 72], control of dynamic systems [73, 74, 75], and secure telecommunication [76, 77, 78]. In this way, they have become an object of extensive research in the last decade. See, for example [19, 79, 80, 81, 82, 83], and their references.

The design of adaptive observers is motivated by the knowledge of the dominant system dynamics. In this sense, there are at least two approaches. In the first approach, it is considered that the model structure is known and only its parameters are unknown [84, 85, 86]. In the second, most of the system model structure is assumed unknown and its associated state vector is bounded in norm. Typical examples of the second approach are observers based on neural networks [48, 87, 88, 89] and fuzzy systems [66, 90, 91]. It is worth noting that the last approach extends the application as it relaxes the need to accurately know the system model, which corresponds typically to cases of practical application. For example, in [48, 87] Adaptive observers discontinuous were proposed based on linearly parameterized neural networks and with activation functions defined respectively by wavelets and sigmoid. Although the observers in [48, 87] ensure the existence of an upper bound for the mean square error of observation, a preliminary stage of experimentation is required to obtain an approximation of the nominal weight which are used in the algorithms, a fact that complicates the implementation when prior experimental data of the system are not available.

Aiming to ensure residual state errors asymptotically null, in [82] and [89] discontinuous adaptive observers were proposed. In [82] was considered a class of mechanical systems and proposed an observer commuting between an adaptive neural mode for large values of error and a nonadaptive sliding mode for small errors. However, the sliding mode operation requires prior knowledge of an upper bound for the neural approximation error which is usually unknown in practice. Similarly, in [89] was assured the asymptotic convergence of the estimated state for actual using a discontinuous observer based on sliding mode and estimation of a bound for the approximation errors. However, it is important to note that observers in [48, 82, 87, 89] show chattering because of delays and imperfections of switching devices. The chattering due to the use of these observed may result in poor control precision, high heat losses in electric power circuits, high wear of mechanical moving parts and high frequency unmodeled dynamics excitation which degrades the system performance and can cause the system instability [34].

Unknown input observers which decouple the residual signal from the unknown disturbances were introduced by the pioneering work of Wuennenberg and Frank in [42] and then considerable contribution was made in [7, 8, 30].

One way to prevent the chattering is using a continuous approximation of the discontinuity. In this regard, in [83] a continuous observer was proposed for uncertain nonlinear systems based on artificial neural networks which ensure the convergence of the observation residual error to zero even in the presence of rounding errors, disturbances and time varying parameters. However, the proposed observer assumes that bounds for the approximation error, disturbances, and nominal weights are known in advance.

It should be noted that in [48, 82, 83, 87, 89] the dependence between the various design parameters with the observer performance is not simple because the implementation requires, among other hypotheses, that several linear matrix inequalities be satisfied.

In [92] an adaptive neural observer was designed, which does not present a chattering, does a scaling of unknown nonlinearities, and is able to relate design parameters to the size of the residual state error, even in the presence of disturbances. However, in this case it is not possible to control the transient time duration without changing the design matrices so as not to modify the size of the residual state error.

In the context of identification, it is important to say that sometimes it is desired to use a feedback in the system, which is the case of control. Control cases occur when a closed-loop identification is performed and has been extensively studied in the literature. Several areas of knowledge explore the use of controllers and one case that is worth emphasizing is the case of synchronization of master and slave systems. Cases such as the chaotic synchronization, proposed in 1990, emerged with the purpose of increasing the reliability of the area of communication with security [93].

Many works in nonlinear control systems have been found in the literature [94, 95, 96]. Other examples can be found using neural networks as [97, 98]. It is relevant to say that the control systems using dynamic neural networks were first introduced in [18].

Some relevant works were found, for example [99, 100]. In [100] an adaptive controller design is applied for nonlinear systems with parameter uncertainties and control constraints. In [99] the author proposed to achieve stabilization and synchronization of a Chen–Lee control system. In the area of nonlinear control many applications have been found, for example in welding case [101, 102, 103, 104, 105] and in the financial case [106].

The works we found in control area focuses on the boundedness of the tracking error. However, none of these works focused on transient duration, in order to relate this duration to a design parameter.

Thus, in the previous works, manipulation of the duration of transient without changing the size of residual state error was not considered. Motivated by this information, in the next chapters, we propose an online identification, observation, and control scheme that uses artificial neural networks that allows an adjustment of the duration of the transient regime from a design parameter that is not related to the size of the residual state error.

Chapter 3

Online Neuro Identification of Uncertain Systems With Control of Residual Error and Transient Time

This chapter focuses on the online identification problem of uncertain systems. By using a neural identification model with feedback, scaling, and a weight law based on Lyapunov theory, an online identification algorithm is proposed to make ultimately bounded the residual state error and related to two design matrix. In addition, it is shown that the transient can be controlled by a constant which is not related to the residual state error. In this way, it is shown that it is possible to decouple the transient performance of the steady state error. In order to validate the theoretical results, the identification of two chaotic systems was accomplished by comparing its performance achieved while changing the design parameters. In this chapter, a neural identification of a welding system with chaotic behavior was also done with a focus on the reduction of the residual state error.

3.1 Problem Formulation

Consider the following nonlinear differential equation

$$\dot{x} = F\left(x, u, v, t\right), \quad x\left(0\right) = x_0 \tag{3.1}$$

where $x \in X \subset \Re^n$ is the n-dimensional state vector, $u \in U \subset \Re^m$ is a m-dimensional admissible input vector, $v \in V \subset \Re^p$ is p-dimensional a vector of time varying uncertain variables, t is the time, and $F : X \times U \times V \times [0, \infty) \mapsto \Re^n$ is a continuous map. In order to have a well-posed problem, we assume that X, U, V are compact sets and F is locally Lipschitzian with respect to x in $X \times U \times V \times [0, \infty)$, such that (3.1) has a unique solution through x_0.

We assume that the following can be established:

Assumption 1. *On a region* $X \times U \times V \times [0, \infty)$

$$\|h(x,u,v,t)\| < h_0 \tag{3.2}$$

where

$$h\left(x,u,v,t\right) = F\left(x,u,v,t\right) - f\left(x,u\right) \tag{3.3}$$

f is an unknown map, h are internal or external disturbances, and $\bar{h}_0$, such that $\bar{h}_0 > h_0 \geq 0$, is a unknown constant. Note that (3.2) is verified when x and u evolve on compact sets and the temporal disturbances are bounded.

Thus, except for the Assumption 1, we say that $F(x,u,v,t)$ is an unknown map and our aim is to design a identifier based on neural networks for (3.1) to ensure the state error convergence, which will be accomplished despite the presence of approximation error and disturbances.

3.2 Identification Model and State Estimate Error Equation

The following lines presented follow a pattern adopted in [50]

We start by presenting the identification model and the definition of the relevant errors associated with the problem.

Let $\bar{f}$ be the best known approximation of f, $P \in \Re^{n \times n}$ a scaling matrix defined as $P = P^T > 0$, $\bar{g} = P^{-1}g$, and $g(x,u) = f(x,u) - \bar{f}(x,u)$. Then, by adding and subtracting $\bar{f}(x,u)$, the system (3.1) becomes

$$\dot{x} = \bar{f}\left(x,u\right) + P\bar{g}\left(x,u\right) + h\left(x,u,v,t\right) \tag{3.4}$$

Remark 1. It should be noted that if the designer has no previous knowledge of f, so $\bar{f}$ is simply assumed as being the zero vector. From (3.4), by using LPNNs, the nonlinear mapping $\bar{g}(x,u)$ can be replaced by the neural parametrization $W^*\sigma(x,u)$ plus an approximation error term $\varepsilon(x,u)$. More exactly, (3.4) can be rewritten as

$$\dot{x} = \bar{f}\left(x,u\right) + PW^*\sigma\left(x,u\right) + P\varepsilon\left(x,u\right) + h\left(x,u,v,t\right) \tag{3.5}$$

where $\sigma\left(x,u\right)$ is a nonlinear vector function whose arguments are preprocessed by a scalar sigmoidal function $s(\cdot)$ and $W^* \in \Re^{n \times L}$ is the "optimal" or ideal matrix, only required for analytical purposes, which can be defined as

$$W^* := \underset{(\hat{W} \in \Gamma)}{\arg\min} \left\{ \left\| \bar{g}\left(x,u\right) - \hat{W}\sigma\left(x,u\right) \right\|_{\infty} \right\} \tag{3.6}$$

where $x \in X$, $u \in U$, $\Gamma = \left\{\hat{W} \in \Re^{n\times L} : ||\,\hat{W}\,||_F < \alpha_{\hat{w}}\right\}$, $\alpha_{\hat{w}}$ is a strictly positive constant, $\hat{W}$ is an estimate of W^*, and $\varepsilon(x,u)$ is an approximation error term, corresponding to W^*, which can be defined as

$$\varepsilon\left(x,u\right) := \bar{g}\left(x,u\right) - W^*\sigma\left(x,u\right) \tag{3.7}$$

The approximation, reconstruction, or modeling error ε in (3.7) is a quantity that exists due to the incapacity of LPNNs to match the unknown map $\bar{g}(x,u)$. Since X, U are compact sets and from (I.17), the following can be established.

Assumption 2. *On a region $X \times U$, the approximation error is upper bounded by*

$$\|\varepsilon(x,u)\| < \varepsilon_0 \tag{3.8}$$

where $\bar{\varepsilon_0}$, such that $\bar{\varepsilon_0} > \varepsilon_0 \geq 0$, is an unknown constant.

Remark 2. The assumption 1 is usual in identification literature. The assumption 2 is quite natural since $\bar{g}$ is continuous and their arguments evolve on compact sets and σ satisfies (I.17).

Remark 3. Note that any σ_0, h_0, and ε_0 are the smallest constants such that (I.17), (3.2), and (3.8) are satisfied.

Remark 4. It should be noted that W^* and $\varepsilon(x,u)$ might be nonunique. However, the uniqueness of $||\varepsilon(x,u)||$ is ensured by (3.6).

Remark 5. It should be noted that W^* was defined as being the value of $\hat{W}$ that minimizes the L_∞-norm difference between $\bar{g}(x,u)$ and $\hat{W}\sigma(x,u)$. The scaling matrix P from (3.4) is introduced to manipulate the magnitude of uncertainties and hence the magnitude of the approximation error. This procedure improves the performance of the identification process.

Remark 6. Notice that the proposed neuro-identification scheme is a black-box methodology, hence the external disturbances and approximation error are related. Based on the system input and state measurements, the uncertain system (including the disturbances) is parametrized by a neural network model plus an approximation error term. However, the parametrization (3.5) is motivated by the fact that neural networks are not adequate for approximating external disturbances, since the basis function depends on the input and states, whereas the disturbances depend on the time and external variables. The aim for presenting the uncertain system in the form (3.5), where the disturbance h is explicitly considered, is also to highlight that the proposed scheme is in addition valid in the presence of unexpected changes in the systems dynamics that can emerge, for instance, due to environment change, aging of equipment or faults. We propose an identification model of the form

$$\dot{\hat{x}} = -L\left(\hat{x} - x\right) - \gamma_W\gamma_0\left(\hat{x} - x\right) + P\hat{W}\sigma\left(x,u\right) \tag{3.9}$$

where $\hat{x}$ is the estimated state, $\gamma_W > 0$, $\gamma_0 > 0$, and $L \in \Re^{n\times n}$ is a positive definite feedback gain matrix introduced to attenuate the effect of the uncertainties and disturbances. It will be demonstrated that the identification model (3.9) used in conjunction with a convenient adjustment law for $\hat{W}$, to be proposed in the next section, ensures the convergence of the state error to a neighborhood of the origin, even in the presence of the approximation error and disturbances, whose radius depends on the design parameters.

Remark 7. Note that the identification model requires states are available to measure. However, the main relevance of the method is to provide a parametrization for the uncertain system (3.1) that can be later used to project adaptive control and observation schemes.

Remark 8. It should be noted that in our formulation, the LPNNs is only required to approximate $P^{-1}[f(x,u)-\bar{f}(x,u)]$ (whose magnitude is often small) instead of the entire function $P^{-1}[f(x,u)]$. Hence, standard identification methods (to obtain some previous $\bar{f}$]) can be used together with the proposed algorithm to improve performance. By defining the state estimation error as $\tilde{x}=\hat{x}-x$, from (3.5) and (3.9), we obtain the state estimation error equation

$$\dot{\tilde{x}} = -L\tilde{x} - \gamma_W\gamma_0\tilde{x} + P\tilde{W}\sigma\left(x,u\right) - P\varepsilon\left(x,u\right) - h\left(x,u,v,t\right) \tag{3.10}$$

where $\tilde{W}=\hat{W}-W^*$.

3.3 Adaptive Laws and Stability Analysis

Before presenting the main theorem, We state some facts, which will be used in the stability analysis.

Fact 1. In our problem, the following equation is valid:

$$tr\left(\tilde{W}^T\tilde{x}\sigma^T\right) = \tilde{x}^T\tilde{W}\sigma \tag{3.11}$$

Fact 2. Let $W^*, W_0, \hat{W}, \tilde{W} \in \Re^{n\times L}$. Then, with the definition of $\tilde{W}=\hat{W}-W^*$, the following equations are true:

$$2tr\left[\tilde{W}^T\left(\hat{W}-W_0\right)\right] = \left\|\tilde{W}\right\|_F^2 + \left\|\hat{W}-W_0\right\|_F^2 - \left\|W^*-W_0\right\|_F^2 \tag{3.12}$$

Fact 3. Let $A\in\Re^{c\times d}$, $b\in\Re^c$, where $c>0$ and $d>0$ are whole numbers. Then, the following expressions are true:

$$tr\left(A^T+A\right) = tr\left(2A\right) = 2tr\left(A\right) \tag{3.13}$$

$$-b^TAb \le -b^T\lambda_{min}\left(A\right)b \tag{3.14}$$

where $\lambda(A)$ is its eigenvalues

Fact 4. Whereas that a, b and $c\in\Re^+$, so

$$a\left\|\tilde{x}\right\|^2 - b\left\|\tilde{x}\right\| - c > 0 \tag{3.15}$$

$$\|\tilde{x}\|^2 - \frac{b}{a}\|\tilde{x}\| > \frac{c}{a} \tag{3.16}$$

$$\|\tilde{x}\|^2 - \frac{b}{a}\|\tilde{x}\| + \frac{b^2}{4a^2} > \frac{c}{a} + \frac{b^2}{4a^2} \tag{3.17}$$

$$\left(\|\tilde{x}\| - \frac{b}{2a}\right)^2 > \frac{4ac + b^2}{4a^2} \tag{3.18}$$

$$\|\tilde{x}\| - \frac{b}{2a} > \frac{\pm\sqrt{4ac + b^2}}{2a} \tag{3.19}$$

$$\|\tilde{x}\| > \frac{b \pm \sqrt{4ac + b^2}}{2a} \tag{3.20}$$

As $b - \sqrt{4ac + b^2} < 0$ and $\|\tilde{x}\| \geq 0$, this is an invalid solution, so

$$\|\tilde{x}\| > \frac{b + \sqrt{4ac + b^2}}{2a} \tag{3.21}$$

$$\|\tilde{x}\| > \frac{\frac{b}{2} + \sqrt{ac + \left(\frac{b}{2}\right)^2}}{a} \tag{3.22}$$

Fact 5. Whereas that a, b, and $c \in \Re^+$, so

$$m(\tilde{x}) = -a\|\tilde{x}\|^2 + b\|\tilde{x}\| + c \tag{3.23}$$

The derivative of equation (3.23) is equal to

$$\dot{m} = -2a\|\tilde{x}\| + b \tag{3.24}$$

The maximum value of (3.23) occurs when $\dot{m} = 0$

$$\|\tilde{x}\| = \frac{b}{2a} \tag{3.25}$$

Replacing this value in (3.23)

$$m = -a\left(\frac{b}{2a}\right)^2 + b\left(\frac{b}{2a}\right) + c \tag{3.26}$$

$$m = -\frac{b^2}{4a} + \frac{2b^2}{4a} + c \tag{3.27}$$

Thus, the maximum value of (3.23) is equal to

$$m(\tilde{x}) = \frac{4ac + b^2}{4a} \tag{3.28}$$

We now state and prove the main theorem of this chapter.

Theorem 3.3.1. *Consider the class of general nonlinear systems described by (3.1) which satisfies Assumptions 1-2, the identification model (3.9). Let the weight law be given by*

$$\dot{\hat{W}} = -2\gamma_W \left[\gamma_0 \left(\hat{W} - W_0\right) + \tilde{x}\sigma^T\right] \tag{3.29}$$

where $\dot{\hat{W}} = \dot{\tilde{W}}$, W_0 is a constant matrix and P is arbitrary, since $P = P^T > 0$, then the following is valid

$$L^T P^{-1} + P^{-1} L = Q \tag{3.30}$$

where $L > 0$ and $Q > 0$. So the errors $\tilde{x}$, $\tilde{W}$ are bounded and $\tilde{x}$ is uniformly ultimately bounded with ultimate bound ρ_2, where $\rho_2 = \frac{\frac{b}{2}+\sqrt{\lambda_{min}(Q)c+(\frac{b}{2})^2}}{\lambda_{min}(Q)}$, $b = 2\bar{\varepsilon}_0 + 2\left\|P^{-1}\right\|_F \bar{h}_0$, and $c = \gamma_0 \left\|W^* - W_0\right\|_F^2$.

Proof. Consider the Lyapunov function candidate

$$V = \tilde{x}^T P^{-1}\tilde{x} + \frac{tr\left(\tilde{W}^T \gamma_W^{-1} \tilde{W}\right)}{2} \tag{3.31}$$

By Deriving (3.31), we obtain

$$\dot{V} = \dot{\tilde{x}}^T P^{-1}\tilde{x} + \tilde{x}^T P^{-1}\dot{\tilde{x}} + \frac{\gamma_W^{-1} tr\left(\tilde{W}^T \dot{\tilde{W}} + \dot{\tilde{W}}^T \tilde{W}\right)}{2} \tag{3.32}$$

$$\dot{V} = \tilde{x}^T P^{-1}\dot{\tilde{x}} + \left(\tilde{x}^T P^{-1}\dot{\tilde{x}}\right)^T + \frac{\gamma_W^{-1} tr\left[\tilde{W}^T \dot{\tilde{W}} + \left(\tilde{W}^T \dot{\tilde{W}}\right)^T\right]}{2} \tag{3.33}$$

Using equation (3.13), this results

$$\dot{V} = \tilde{x}^T P^{-1}\dot{\tilde{x}} + (\tilde{x}^T P^{-1}\dot{\tilde{x}})^T + \gamma_W^{-1} tr\left(\tilde{W}^T \dot{\tilde{W}}\right) \tag{3.34}$$

Replacing equations (3.10) and (3.29)

$$\begin{aligned}\dot{V} &= \tilde{x}^T P^{-1}\left(-L\tilde{x} - \gamma_W\gamma_0\tilde{x} + P\tilde{W}\sigma - P\varepsilon - h\right)\\ &+ \left[\tilde{x}^T P^{-1}\left(-L\tilde{x} - \gamma_W\gamma_0\tilde{x} + P\tilde{W}\sigma - P\varepsilon - h\right)\right]^T\\ &+ \gamma_W^{-1} tr\left\{\tilde{W}^T\left[-2\gamma_W\left(\gamma_0\left(\hat{W} - W_0\right) + \tilde{x}\sigma^T\right)\right]\right\}\end{aligned} \tag{3.35}$$

$$\begin{aligned}\dot{V} &= -\tilde{x}^T\left(P^{-1}L + L^T P^{-1}\right)\tilde{x} - \gamma_W\gamma_0\left[\tilde{x}^T P^{-1}\tilde{x} + \left(\tilde{x}^T P^{-1}\tilde{x}\right)^T\right]\\ &+ \tilde{x}^T\left(\tilde{W}\sigma - \varepsilon - P^{-1}h\right) + \left(\tilde{x}^T\left(\tilde{W}\sigma - \varepsilon - P^{-1}h\right)\right)^T\\ &- 2\gamma_0 tr\left[\tilde{W}^T\left(\hat{W} - W_0\right)\right] - 2tr\left(\tilde{W}^T\tilde{x}\sigma^T\right)\end{aligned} \tag{3.36}$$

Since $\tilde{x}^T\left(\tilde{W}\sigma - \varepsilon - P^{-1}h\right)$ is a scalar number, so the transpose of this number is itself and employing facts 1 and 2 and equation (3.30), it results

$$\begin{aligned}\dot{V} &= -\tilde{x}^T Q\tilde{x} - 2\gamma_W\gamma_0\tilde{x}^T P^{-1}\tilde{x} + 2\tilde{x}^T\left(\tilde{W}\sigma - \varepsilon - P^{-1}h\right)\\ &- \gamma_0\left(\left\|\tilde{W}\right\|_F^2 + \left\|\hat{W} - W_0\right\|_F^2 - \left\|W^* - W_0\right\|_F^2\right) - 2\tilde{x}^T\tilde{W}\sigma\end{aligned} \tag{3.37}$$

Turning to an inequality and taking into account (3.14)

$$\begin{aligned}\dot{V} &\le -\lambda_{min}(Q)\left\|\tilde{x}\right\|^2 - 2\gamma_W\gamma_0\tilde{x}^T P^{-1}\tilde{x}\\ &+ 2\left\|\tilde{x}\right\|\left(\left\|\varepsilon\right\| + \left\|P^{-1}\right\|_F\left\|h\right\|\right) - \gamma_0\left(\left\|\tilde{W}\right\|_F^2 + \left\|\hat{W} - W_0\right\|_F^2 - \left\|W^* - W_0\right\|_F^2\right)\end{aligned} \tag{3.38}$$

Considering that $\|\varepsilon\| < \bar{\varepsilon}_0$, $\|h\| < \bar{h}_0$, and rearranging (3.38) implies

$$\begin{aligned}\dot{V} &\le -\left\|\tilde{x}\right\|^2\left[\lambda_{min}(Q)\right] + \left\|\tilde{x}\right\|\left(2\bar{\varepsilon}_0 + 2\left\|P^{-1}\right\|_F\bar{h}_0\right) + \gamma_0\left\|W^* - W_0\right\|_F^2\\ &- \gamma_0\left\|\tilde{W}\right\|_F^2 - \gamma_0\left\|\hat{W} - W_0\right\|_F^2 - 2\gamma_W\gamma_0\tilde{x}^T P^{-1}\tilde{x}\end{aligned} \tag{3.39}$$

Considering that $a = \lambda_{min}(Q)$, $b = 2\bar{\varepsilon}_0 + 2\left\|P^{-1}\right\|_F\bar{h}_0$, $c = \gamma_0\left\|W^* - W_0\right\|_F^2$,where $a \ge 0$, $b \ge 0$, and $c \ge 0$, then

$$\begin{aligned}\dot{V} &\le -a\left\|\tilde{x}\right\|^2 + b\left\|\tilde{x}\right\| + c\\ &- \gamma_0\left\|\tilde{W}\right\|_F^2 - \gamma_0\left\|\hat{W} - W_0\right\|_F^2 - 2\gamma_W\gamma_0\tilde{x}^T P^{-1}\tilde{x}\end{aligned} \tag{3.40}$$

Case 1. For analysis of the limitation of $\tilde{W}$, resuming (3.40) and disregarding some negative terms

$$\dot{V} \le -a\left\|\tilde{x}\right\|^2 + b\left\|\tilde{x}\right\| + c - \gamma_0\left\|\tilde{W}\right\|_F^2 \tag{3.41}$$

Using fact 5, we have

$$\dot{V} \leq \frac{4ac + b^2}{4a} - \gamma_0 \left\| \tilde{W} \right\|_F^2 \tag{3.42}$$

Hence, $\dot{V} < 0$ as long as

$$\gamma_0 \left\| \tilde{W} \right\|_F^2 > \frac{4ac + b^2}{4a} \tag{3.43}$$

$$\left\| \tilde{W} \right\|_F > \pm \sqrt{\frac{4ac + b^2}{4a\gamma_0}} \tag{3.44}$$

As $-\sqrt{\frac{4ac+b^2}{4a\gamma_0}} < 0$ and $\left\| \tilde{W} \right\|_F \geq 0$, this is an invalid solution, so

$$\left\| \tilde{W} \right\|_F > \sqrt{\frac{4ac + b^2}{4a\gamma_0}} \tag{3.45}$$

$$\left\| \tilde{W} \right\|_F > \sqrt{\frac{\frac{ac}{\gamma_0} + \frac{1}{\gamma_0} \left(\frac{b}{2}\right)^2}{a}} \tag{3.46}$$

Replacing a,b, and c

$$\left\| \tilde{W} \right\|_F > \sqrt{\frac{\lambda_{min}(Q) \left\| W^* - W_0 \right\|_F^2 + \frac{1}{\gamma_0} \left(\bar{\varepsilon}_0 + \left\| P^{-1} \right\|_F \bar{h}_0 \right)^2}{\lambda_{min}(Q)}} = \rho_1 \tag{3.47}$$

Thus, since ρ_1 is a constant, by using Lyapunov arguments [34], we concluded that $\tilde{W}$ is uniformly ultimately bounded, with ultimate bound equal to ρ_1 . Note that if, by any reason, $\left\| \tilde{W} \right\|_F$ escapes of the residual set Ω_1, where $\Omega_1 = \left\{ \tilde{W} : \left\| \tilde{W} \right\|_F \leq \rho_1 \right\}$, $\dot{V}$ becomes negative definite again, and it forces the convergence of the weight error to the residual set Ω_1.

Case 2. For analysis of the limitation of $\tilde{x}$, resuming (3.40) and disregarding some negative terms

$$\dot{V} \leq -a \left\| \tilde{x} \right\|^2 + b \left\| \tilde{x} \right\| + c \tag{3.48}$$

Hence, $\dot{V} < 0$ as long as

$$a \left\| \tilde{x} \right\|^2 - b \left\| \tilde{x} \right\| - c > 0 \tag{3.49}$$

Using fact 4, we have

$$\left\| \tilde{x} \right\| > \frac{\frac{b}{2} + \sqrt{ac + (\frac{b}{2})^2}}{a} \tag{3.50}$$

Replacing a,b, and c

$$\|\tilde{x}\| > \frac{\bar{\varepsilon}_0 + \left\|P^{-1}\right\|_F \bar{h}_0 + \sqrt{\lambda_{min}(Q)\,\gamma_0 \left\|W^* - W_0\right\|_F^2 + \left(\bar{\varepsilon}_0 + \left\|P^{-1}\right\|_F \bar{h}_0\right)^2}}{\lambda_{min}(Q)} = \rho_2 \tag{3.51}$$

Thus, since ρ_2 is a constant, by using Lyapunov arguments [34], we concluded that $\tilde{x}$ is uniformly ultimately bounded, with ultimate bound equal to ρ_2 . Note that if, by any reason, $\|\tilde{x}\|$ escapes of the residual set Ω_2, where $\Omega_2 = \{\tilde{x} : \|\tilde{x}\| \leq \rho_2\}$, $\dot{V}$ becomes negative definite again, and it forces the convergence of the state error to the residual set Ω_2.

Case 3. For analysis of the transient time, resuming (3.40) and doing some manipulations

$$\begin{aligned} \dot{V} \leq & -\alpha V + \alpha V - a\|\tilde{x}\|^2 + b\|\tilde{x}\| + c \\ & - \gamma_0 \left\|\tilde{W}\right\|_F^2 - \gamma_0 \left\|\hat{W} - W_0\right\|_F^2 - 2\gamma_W \gamma_0 \tilde{x}^T P^{-1} \tilde{x} \end{aligned} \tag{3.52}$$

We analyze when $-a\|\tilde{x}\|^2 + b\|\tilde{x}\| + c < 0$. This consideration is necessary, since it is in this period that the transient is occurring

$$\begin{aligned} \dot{V} \leq & -\alpha V + \alpha \left(\tilde{x}^T P^{-1} \tilde{x} + \frac{\gamma_W^{-1}}{2} \left\|\tilde{W}\right\|_F^2\right) - \gamma_0 \left\|\hat{W} - W_0\right\|_F^2 \\ & - \gamma_0 \left\|\tilde{W}\right\|_F^2 - 2\gamma_W \gamma_0 \tilde{x}^T P^{-1} \tilde{x} \end{aligned} \tag{3.53}$$

$$\begin{aligned} \dot{V} \leq & -\alpha V - \gamma_0 \left\|\hat{W} - W_0\right\|_F^2 \\ & + \left\|\tilde{W}\right\|_F^2 \left(\frac{\alpha}{2\gamma_W} - \gamma_0\right) + \tilde{x}^T P^{-1} \tilde{x} \left(\alpha - 2\gamma_W \gamma_0\right) \end{aligned} \tag{3.54}$$

Considering that $\alpha = 2\gamma_W \gamma_0$, then

$$\dot{V} \leq -\alpha V - \gamma_0 \left\|\hat{W} - W_0\right\|_F^2 \tag{3.55}$$

$$\dot{V} \leq -\alpha V \tag{3.56}$$

Using Lemma 3.2.4 [84]. It can be stated that:

$$V(t) \leq e^{-\alpha(t-t_0)} V(t_0), \qquad \forall t \geq t_0 \geq 0 \tag{3.57}$$

Assuming that $t_0 = 0$

$$V(t) \leq V(0) e^{-\alpha t} \tag{3.58}$$

$$ln\left[V(t)\right] \leq -\alpha t + ln\left[V(0)\right] \tag{3.59}$$

$$ln\left[\frac{V(t)}{V(0)}\right] \leq -\alpha t \tag{3.60}$$

$$ln\left[\frac{V(0)}{V(t)}\right] \geq \alpha t \tag{3.61}$$

$$t \leq \frac{ln\left[\frac{V(0)}{V(t)}\right]}{\alpha} \tag{3.62}$$

$\square$

Remark 9. Note that the scaling of unknown nonlinearities has a positive impact on the performance of the identification [50]. The scaling matrix P is introduced to attenuate the effect of approximation errors and disturbances, as can be seen in (3.51).

Remark 10. It is perceived from (3.51), that the size of the residual state error is inversely proportional to $\lambda_{min}(Q)$, where the eigenvalues of Q can be arbitrarily manipulated while changing the values of matrices L and P. Thus, it is possible from these arbitrary design matrices to control the residual state error size.

Remark 11. Note that the time t in (3.62)refers to the transient regime duration in relation to the residual state error determined in (3.51). Thus, we can not say anything about the transient duration for a residual state error smaller than that.

Remark 12. It is possible to verify in (3.62) that the maximum transient duration is inversely proportional to the value of α. Since α is related to γ_0 and γ_W, it is concluded that it is possible to increase or decrease the transient time by changing these design parameters.

Remark 13. Note that the choice of different values of γ_W does not imply a new calculation of γ_0 or the matrices P and L to maintain the desired allocation of the eigenvalues of Q. In addition, it is verified in (3.51) that γ_W does not influence the size of the residual state error norm. Thus, it can be stated that from γ_W it is possible to decouple the transient performance of the steady-state error.

Remark 14. In the equation (3.51) in the numerator there is the term $\left\|P^{-1}\right\|_F \bar{h}_0$, thus the external disturbances are present in that part. In this way, it can be stated that by changing the eigenvalues of the matrix Q, the size of the residual state error is also adjusted, even in the presence of limited disturbances.

Remark 15. In previous works, for example in [49, 50, 51], in spite of the residual state error is bounded, they can not increase or reduce the transient duration independently of residual state error size. To allow this, it is necessary to have a parameter related to the transient duration that does change the residual state error. In this work this was possible because the identifier model and the learning law were chosen in order to allow an independent control of the transient duration.

3.4 Simulation

This section presents three examples to validate the theoretical results and to show the performance in the presence of disturbances. In all simulations, Solver ode45(Dormand-Pince) of Matlab/Simulink®, with variable-step and a relative tolerance of 1e-10 was used to obtain the numerical solutions. First, the identification of a chaotic three-dimensional system under disturbances has been proposed, second it is considered a hyperchaotic finance system in the presence of disturbances. In both cases an analysis is made of the duration of the transient regime in relation to the steady-error. Finally, the identification of a welding system with chaotic behavior is performed.

3.4.1 Chaotic System

To illustrate the application of the proposed scheme, we consider the following example. Consider the chaotic system described in [107]

$$\begin{aligned} \dot{x_1} &= a(x_1 - x_2) + d_{x_1} \\ \dot{x_2} &= -4ax_2 + x_1x_3 + mx_1^3 + d_{x_2} \\ \dot{x_3} &= -adx_3 + x_1^3x_2 + bx_3^2 + d_{x_3} \end{aligned} \tag{3.63}$$

where x_1, x_2, and x_3 are state variables, d_{x_1}, d_{x_2}, and d_{x_3} are unknown disturbances, and a, b, d, and d are parameters, being chosen as $a = 1.8$, $b = -0.07$, $d = 1.5$, and $m = 0.12$. Note that system (3.63) satisfies the Assumption 1, since the state variables evolve into compact sets.

To identify the uncertain system (3.63), the proposed identification model (3.9) and the adaptive law (3.29) were implemented. The initial conditions for the chaotic system and for the identification model were $x_1(0) = 2$, $x_2(0) = 1$, $x_3(0) = 2$, $\hat{x}_1(0) = 0$, $\hat{x}_2(0) = 0$, $\hat{x}_3(0) = 0$, and $\hat{W}(0) = 0$ in order to evaluate the performance of the proposed algorithm under adverse initial conditions. The design matrices were chosen as $P = 5I$ and $L = 2I$, where I is the identity matrix. The nonlinear vector function σ is equal to $\sigma = \begin{bmatrix} s(x_1) & s(x_2) & s(x_3) & s(x_1)^2 & s(x_2)^2 & s(x_3)^2 \end{bmatrix}^T$ and the sigmoidal function used is a logistic function and is equal to $s(.) = \frac{5}{1+e^{-0.5(.)}}$. The design parameter γ_0 was chosen as $\gamma_0 = 1$ and W_0 was chosen as

$$W_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{3.64}$$

To verify the robustness of proposed method, it is considered the presence of the following disturbances: $d_{x_1} = cos(2t)$, $d_{x_2} = 2sin(t)$, and $d_{x_3} = 2sin(2t)$. To better verify each part, the disturbance was introduced from $t = 5s$.

From equation (3.62), it can be seen that the parameter α affects the transient time. We choose the parameter γ_W to control the value of α because γ_W is not related to the size of the residual state error, as can be seen in (3.51). In all states we analyzed for two different values of γ_W. In the first situation, a small γ_W was intentionally chosen to detect the transient and in the second case a high value was chosen with the intention of reducing this transient time.

In this example, the two values chosen for γ_W are $\gamma_W = 0.5$ and $\gamma_W = 15$. Figures 3.1, 3.3, and 3.5 show the performances obtained in the estimation of the three states when $\gamma_W = 0.5$. Figures 3.2, 3.4, and 3.6 show the performances obtained in the estimation of the three states when $\gamma_W = 15$. The Frobenius norm associated with the estimated weights matrix is shown in Figure 3.7 when $\gamma_W = 0.5$ and is shown in Figure 3.8 when $\gamma_W = 15$.

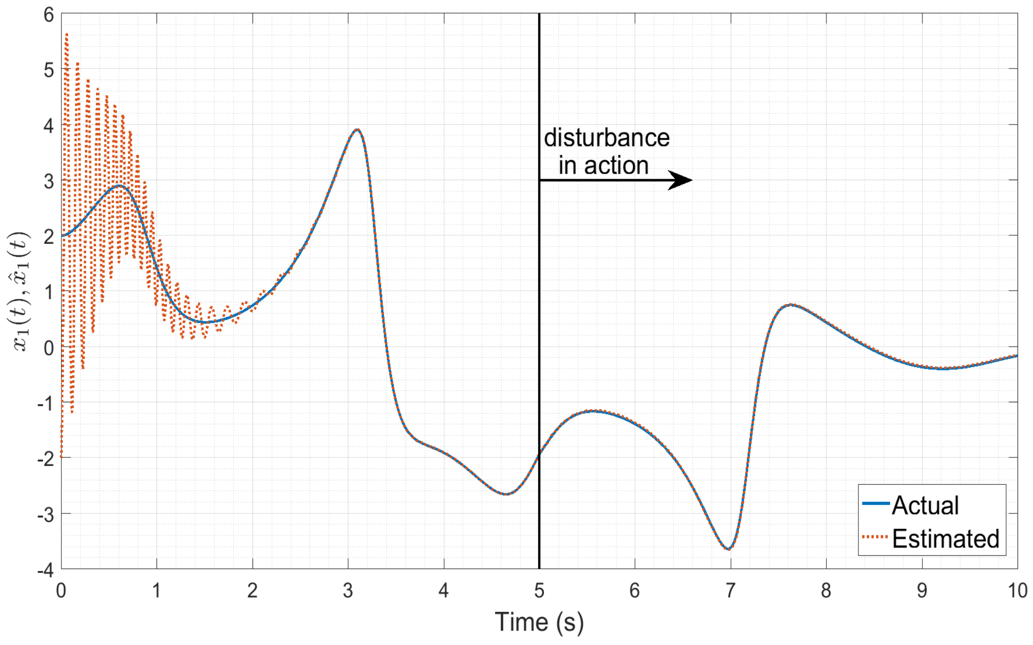


Figure 3.1: Performance in the estimation of x_1 when $\gamma_W = 0.5$

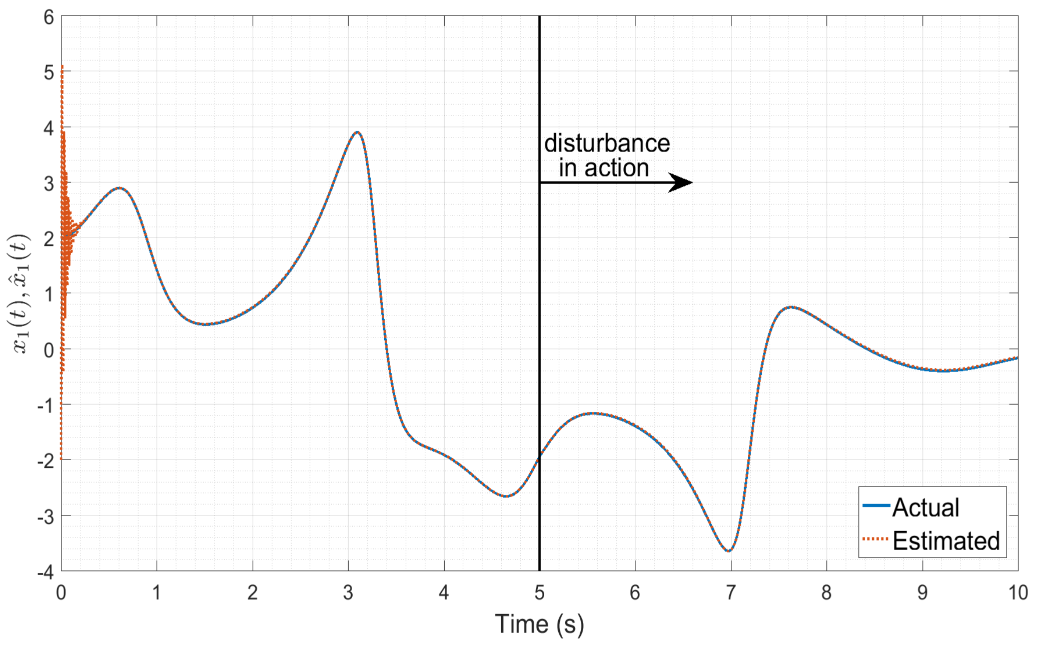


Figure 3.2: Performance in the estimation of x_1 when $\gamma_W = 15$

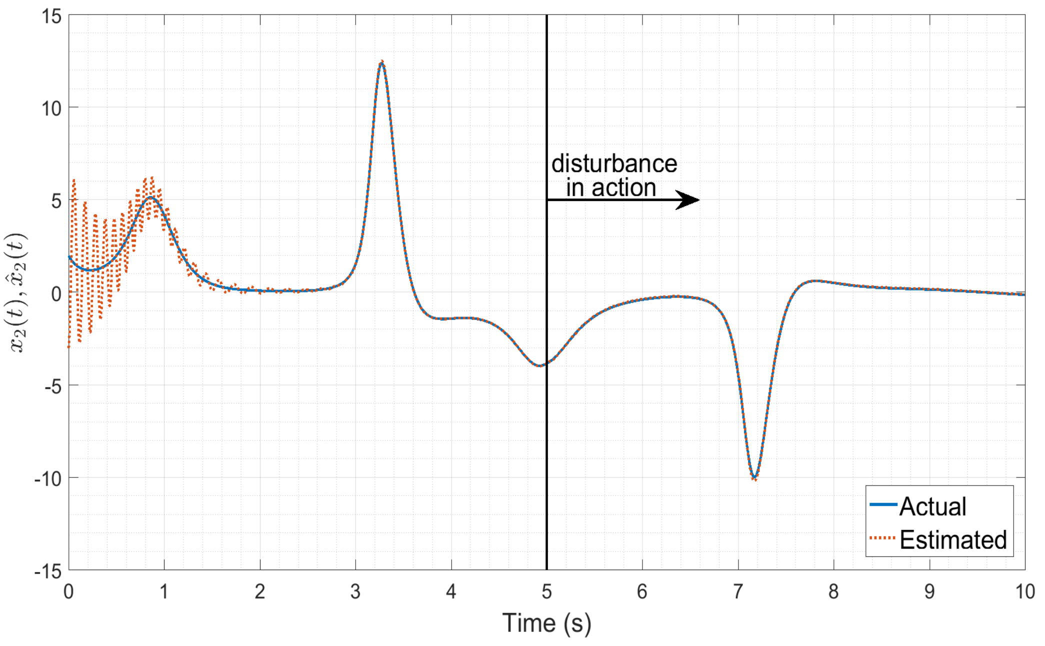


Figure 3.3: Performance in the estimation of x_2 when $\gamma_W = 0.5$

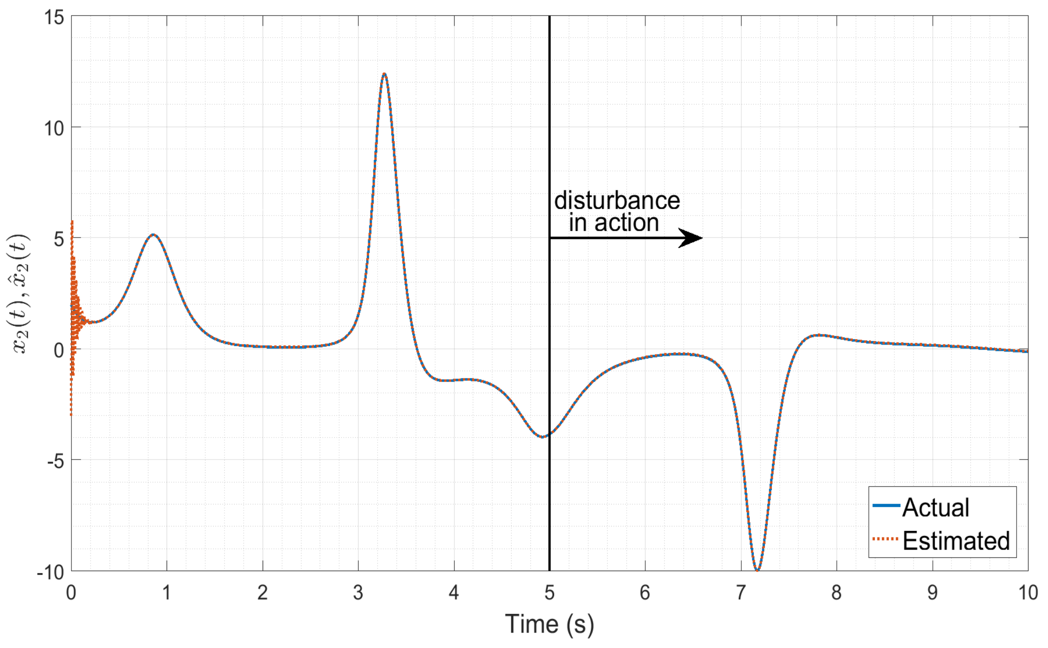


Figure 3.4: Performance in the estimation of x_2 when $\gamma_W = 15$

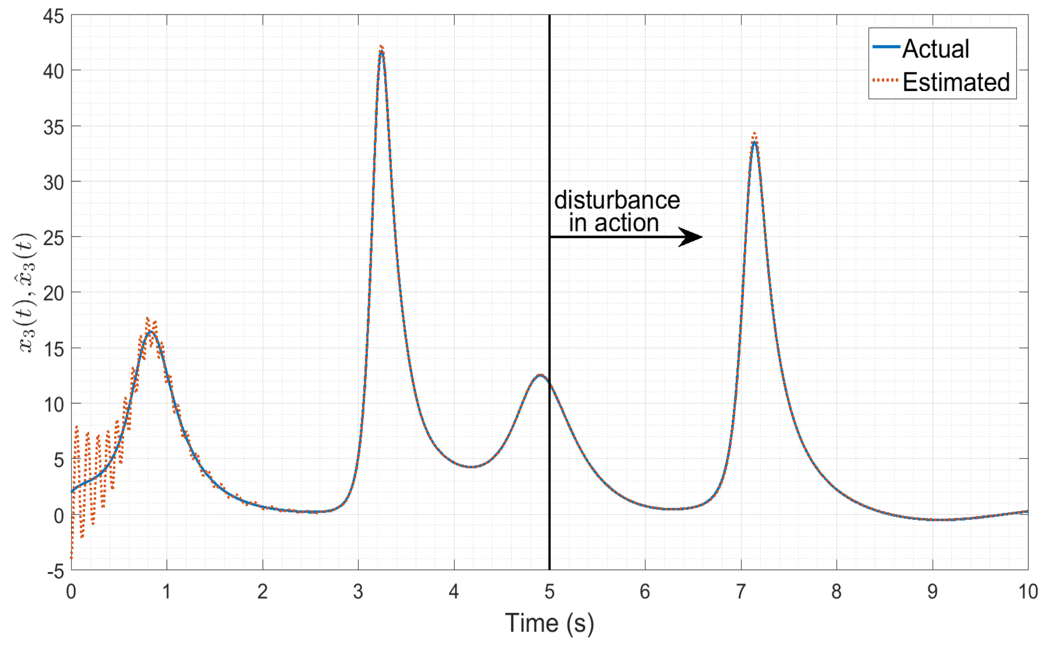


Figure 3.5: Performance in the estimation of x_3 when $\gamma_W = 0.5$

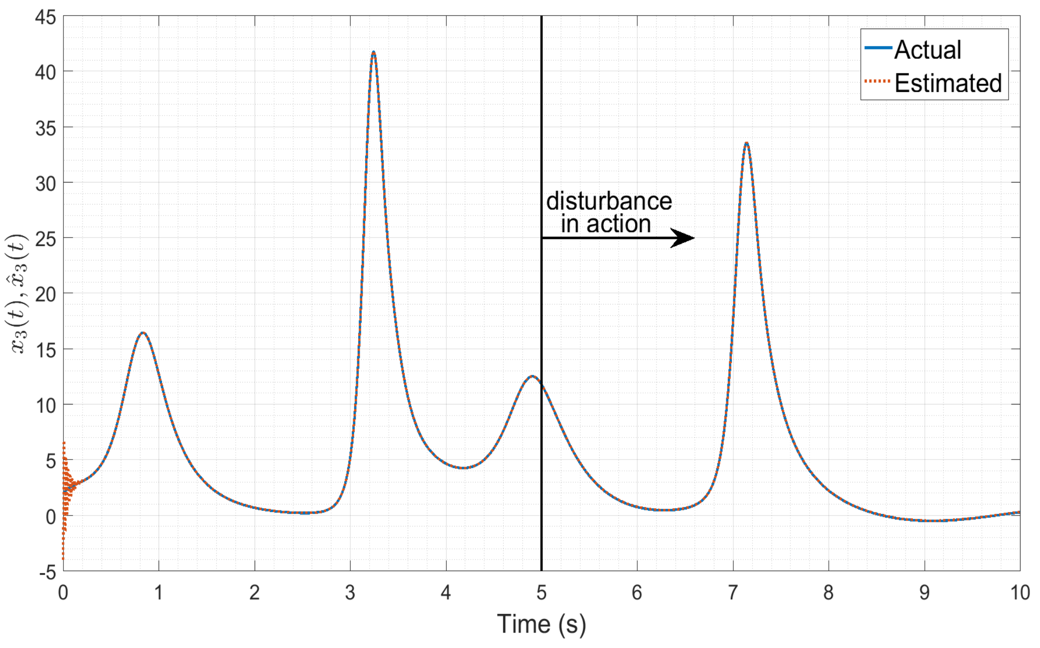


Figure 3.6: Performance in the estimation of x_3 when $\gamma_W = 15$

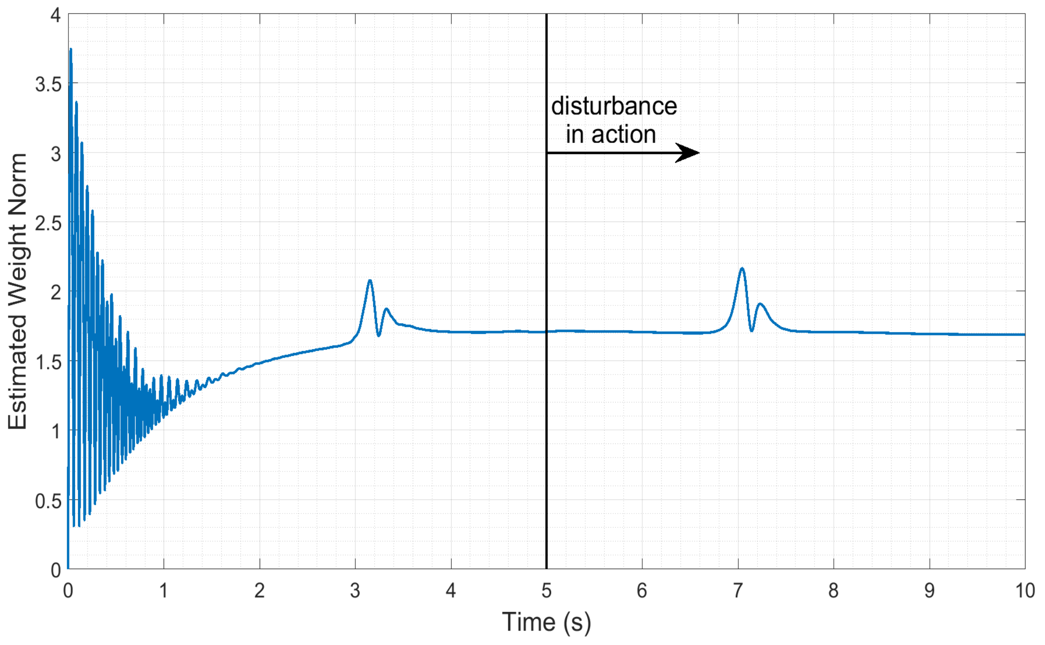


Figure 3.7: Frobenius norm of the estimated weight matrix W when $\gamma_W = 0.5$

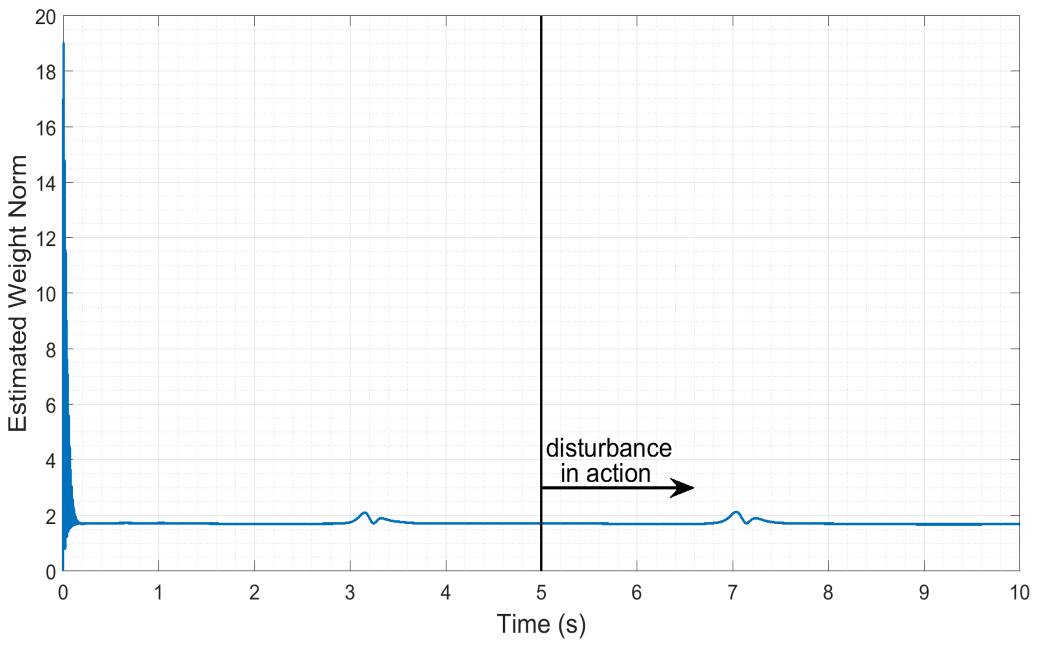


Figure 3.8: Frobenius norm of the estimated weight matrix W when $\gamma_W = 15$

Note that there is a transient time and that the residual state error, after this transient, is approximately zero. Figures 3.2, 3.4, and 3.6 show that there is a shorter transient time in relation to Figures 3.1, 3.3, and 3.5, although the steady-state error behaves similarly in both cases. Figures 3.7 and 3.8 show that the Frobenius norm of the estimated weights performs are similar after the transient, due to large initial uncertainty, and in both cases seems to converge to approximately 1.7,that is, it seems to converge to a constant. Thus, the algorithm is stable and the residual state error converges to a neighborhood of the origin.

The result is as expected, since γ_W, which is directly proportional to α, was the only parameter changed in the two cases. Consequently, it is expected that an increasing of γ_W does not affect the residual state error, and the transient time will be reduced. As can be seen, in the simulations Figures 3.2, 3.4, and 3.6, there is a shorter transient time than in the Figures 3.1, 3.3, and 3.5, and the residual state error appears to be unchanged in a steady state. In this way, it can be said that the results were as expected. In these simulations, L and P are chosen to reduce the residual error to desired one. It would be possible to have changed these values if a minor residual error is required. Note that the identification performed well even in the presence of disturbances from $t = 5s$.

3.4.2 Hyperchaotic Finance System

Consider the hyperchaotic finance system [108], which is described by

$$\begin{aligned}
\dot{x_1} &= x_3 + (x_2 - a)x_1 + x_4 + d_{x_1} \\
\dot{x_2} &= 1 - bx_2 - x_1^2 + d_{x_2} \\
\dot{x_3} &= -x - cx_3 + d_{x_3} \\
\dot{x_4} &= -dx_1x_2 - kx_4 + d_{x_4}
\end{aligned} \tag{3.65}$$

where x_1, x_2, x_3, and x_4 are state variables, where x_1 is the interest rate, x_2 investment demand, x_3 price exponent, and x_4 is the average profit margin.d_{x_1}, d_{x_2}, d_{x_3}, and d_{x_4} are unknown disturbances, and a, b, d, and d are parameters, being chosen as $a = 0.9$, $b = 0.2$, $c = 1.5$, $d = 0.2$, and $k = 0.17$. Note that system (3.65) satisfies the Assumption 1, since the state variables evolve into compact sets.

To identify the uncertain system (3.65), the proposed identification model (3.9) and the adaptive law (3.29) were implemented. The initial conditions for the chaotic system and for the identification model were $x_1(0) = 1$, $x_2(0) = 2$, $x_3(0) = 0.5$, $x_4(0) = 0.5$, $\hat{x}_1(0) = -2$, $\hat{x}_2(0) = -2$, $\hat{x}_3(0) = -2$, $\hat{x}_4(0) = -2$, and $\hat{W}(0) = 0$ in order to evaluate the performance of the proposed algorithm under adverse initial conditions. The design matrices were chosen as $P = 20I$ and $L = 2I$, where I is the identity matrix. The nonlinear vector function σ is equal to $\sigma = \begin{bmatrix} s(x_1) & s(x_2) & s(x_3) & s(x_4) & s(x_1)^2 & s(x_2)^2 & s(x_3)^2 & s(x_4)^2 \end{bmatrix}^T$ and the sigmoidal function used is a logistic function and is equal to $s(.) = \frac{5}{1+e^{-0.5(.)}}$. The design parameter γ_0 was chosen as $\gamma_0 = 1$ and W_0 was chosen as

$$W_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.66}$$

To verify the robustness of proposed method, it is considered the presence of the following disturbances: $d_{x_1} = 2.8cos(6t)$, $d_{x_2} = 4cos(5t)$, $d_{x_3} = 3.6sin(4t)$, and $d_{x_4} = 3.2sin(3t)$. To better verify each part, the disturbance was introduced from $t = 5s$.

From equation (3.62), it can be seen that the parameter α affects the transient time. We choose the parameter γ_W to control the value of α because γ_W is not related to the size of the residual state error, as can be seen in (3.51). In all states we analyzed for two different values of γ_W. In the first situation, a small γ_W was intentionally chosen to detect the transient and in the second case a high value was chosen with the intention of reducing this transient time.

In this example, the two values chosen for γ_W are $\gamma_W = 0.5$ and $\gamma_W = 15$. Figures 3.9, 3.11, 3.13, and 3.15 show the performances obtained in the estimation of the three states when $\gamma_W = 0.5$. Figures 3.10, 3.12, 3.14, and 3.16 show the performances obtained in the estimation of the three states when $\gamma_W = 20$. The Frobenius norm associated with the estimated weights matrix is shown in Figure 3.17 when $\gamma_W = 0.5$ and is shown in Figure 3.18 when $\gamma_W = 20$.

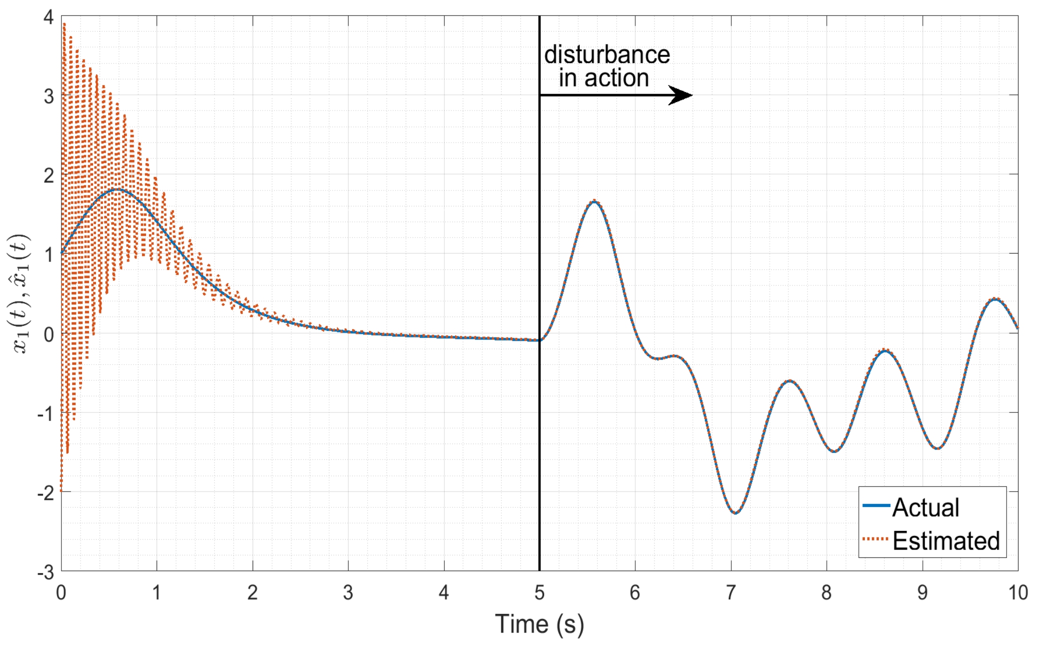


Figure 3.9: Performance in the estimation of x_1 when $\gamma_W = 0.5$

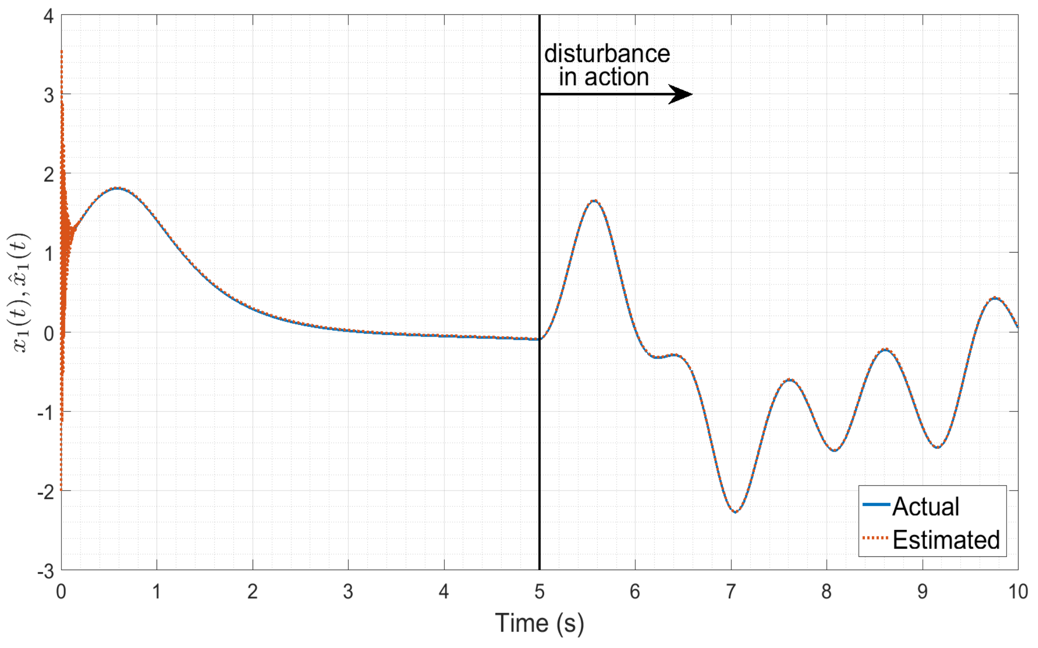


Figure 3.10: Performance in the estimation of x_1 when $\gamma_W = 20$

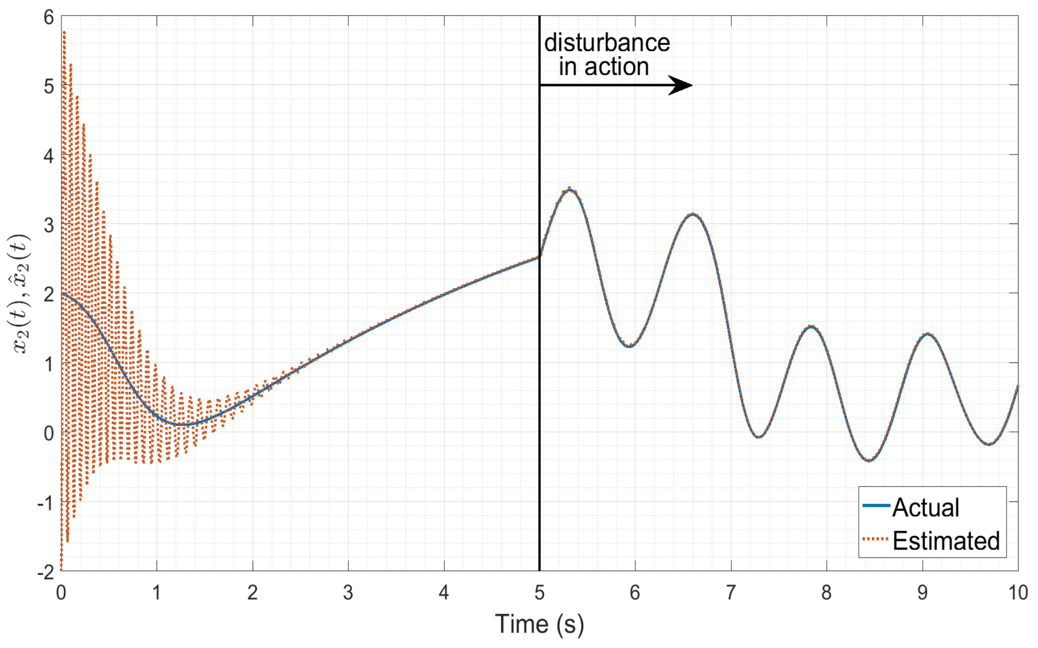


Figure 3.11: Performance in the estimation of x_2 when $\gamma_W = 0.5$

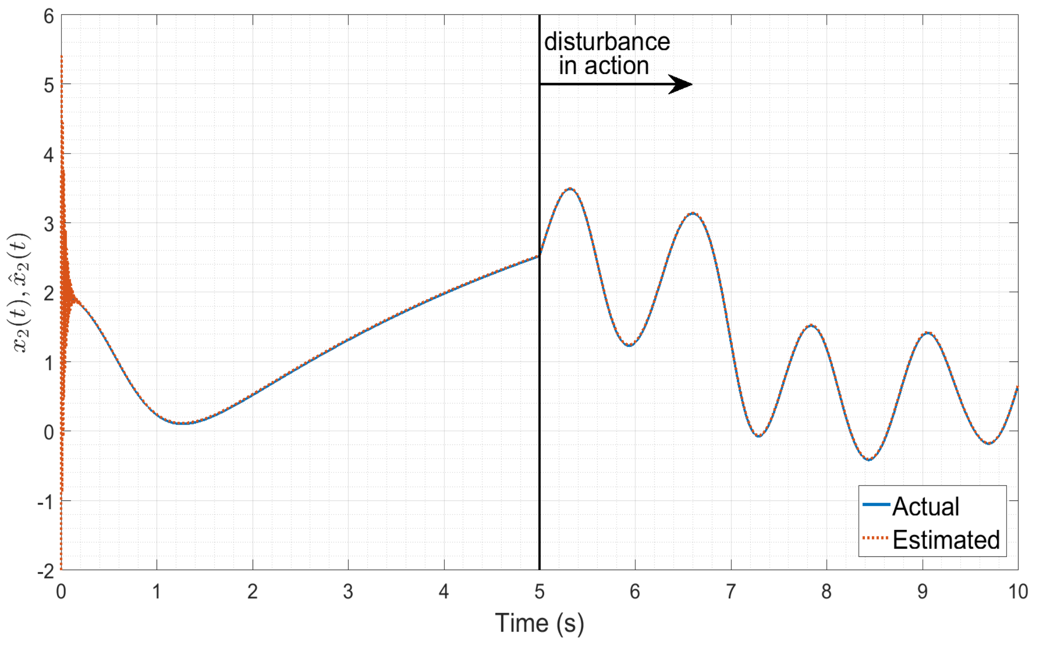


Figure 3.12: Performance in the estimation of x_2 when $\gamma_W = 20$

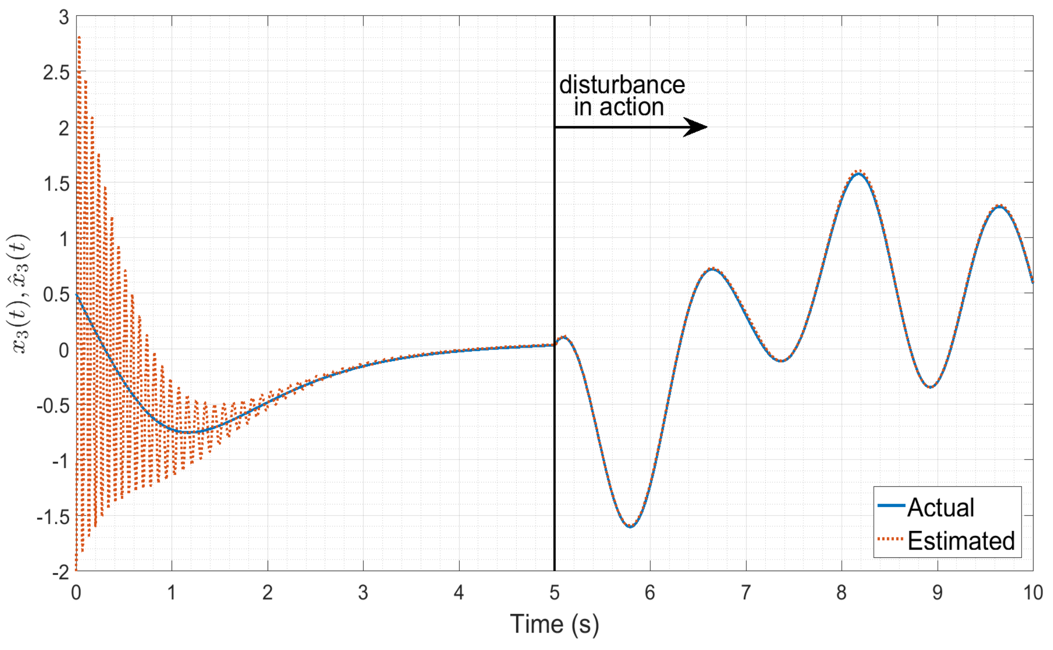


Figure 3.13: Performance in the estimation of x_3 when $\gamma_W = 0.5$

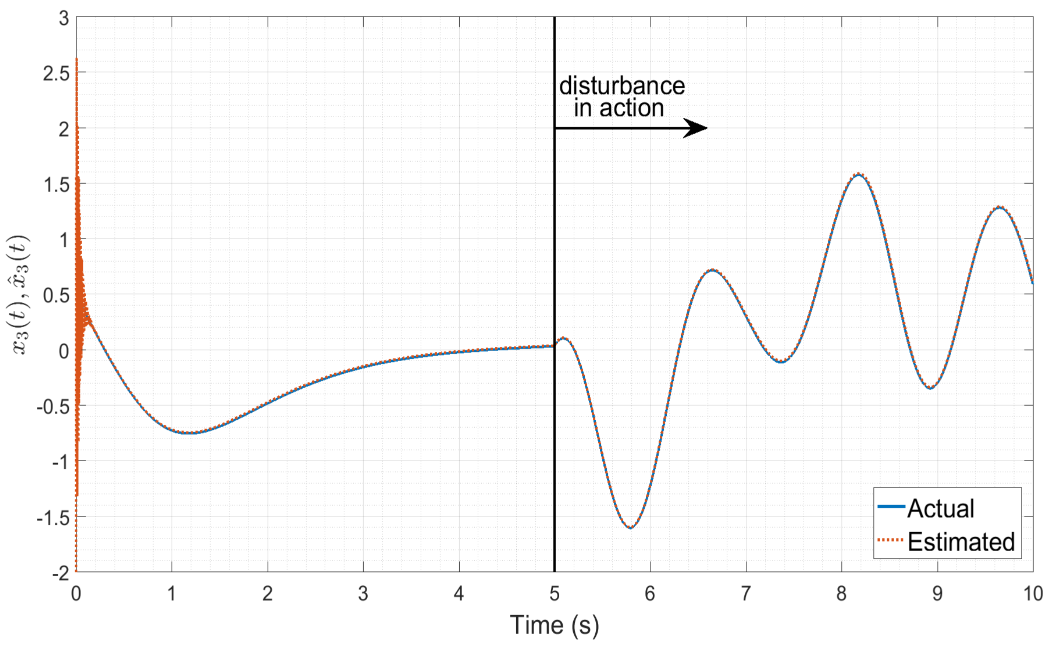


Figure 3.14: Performance in the estimation of x_3 when $\gamma_W = 20$

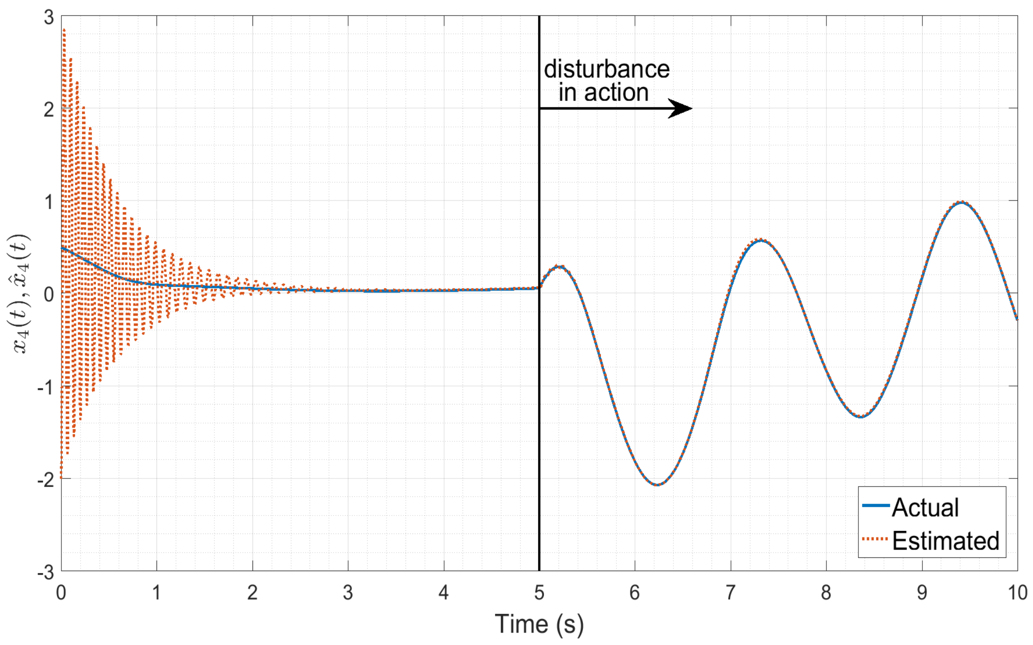


Figure 3.15: Performance in the estimation of x_4 when $\gamma_W = 0.5$

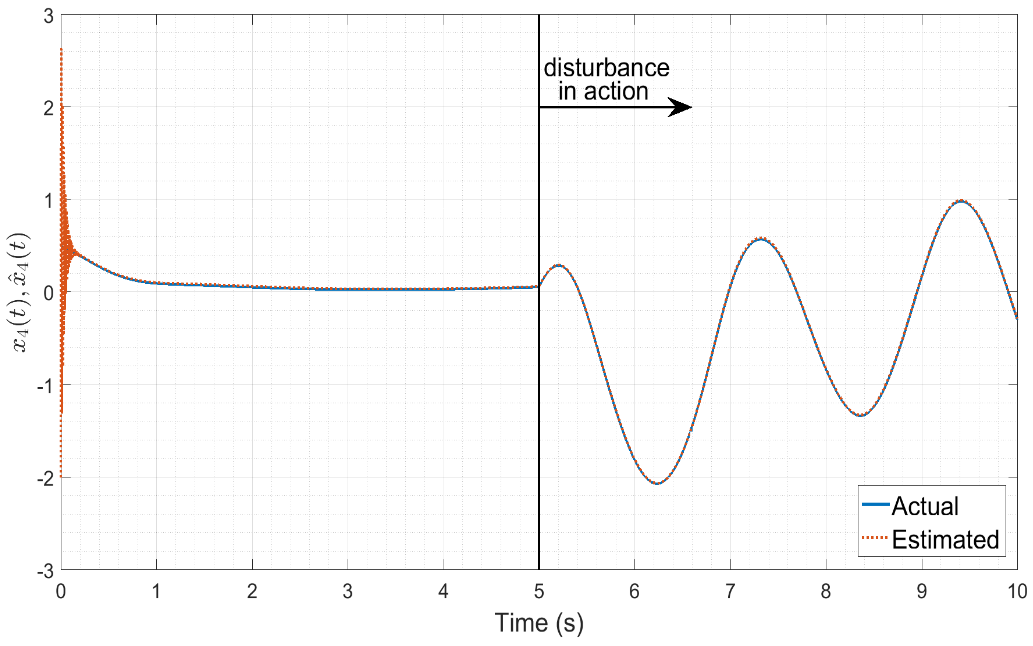


Figure 3.16: Performance in the estimation of x_4 when $\gamma_W = 20$

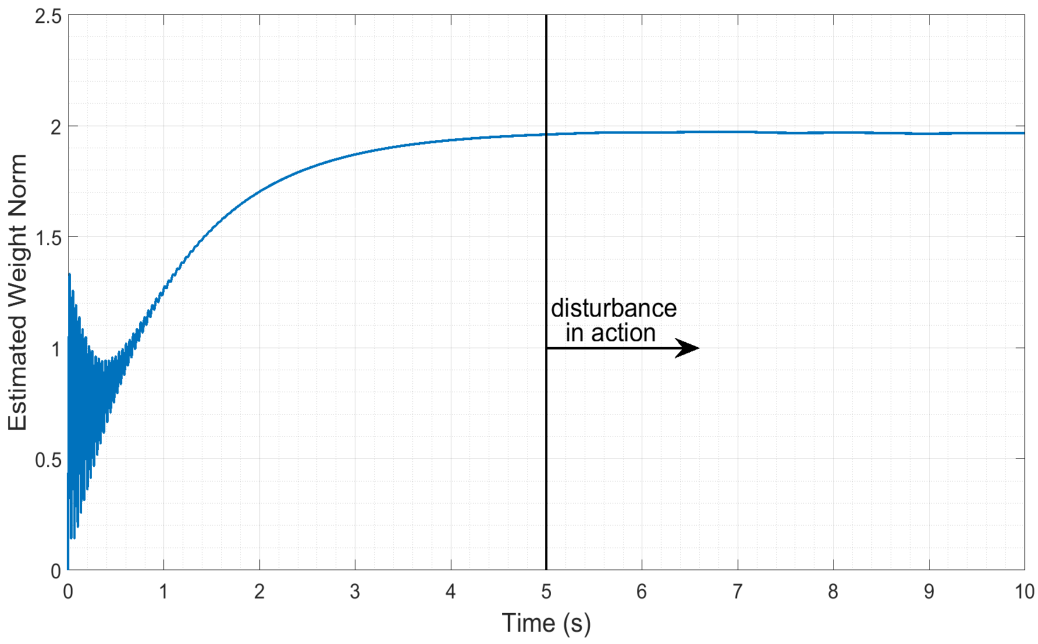


Figure 3.17: Frobenius norm of the estimated weight matrix W when $\gamma_W = 0.5$

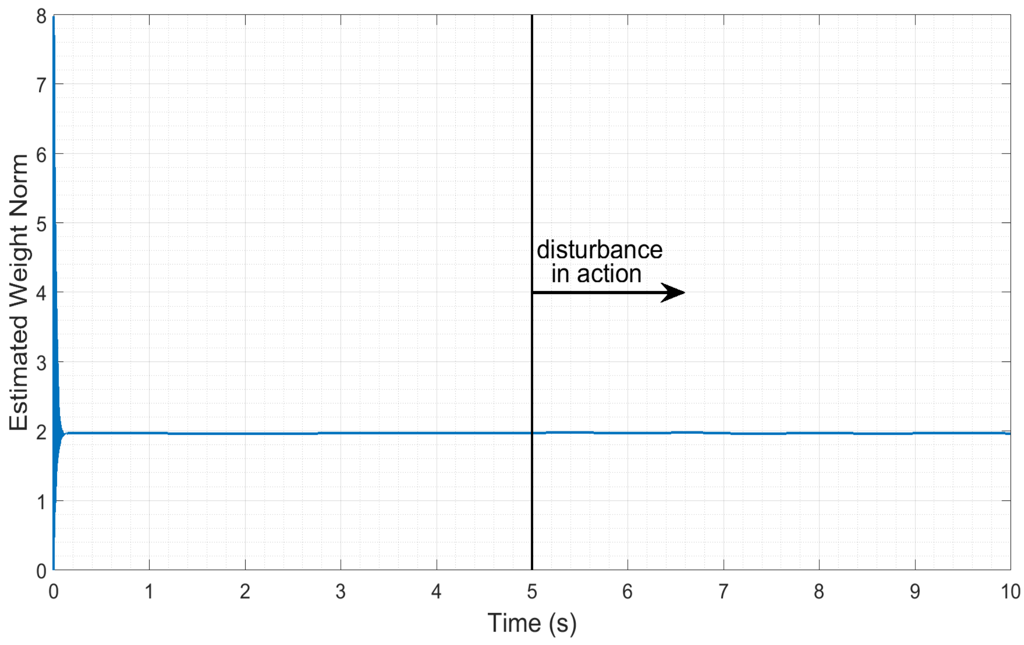


Figure 3.18: Frobenius norm of the estimated weight matrix W when $\gamma_W = 20$

Note that there is a transient time and that the residual state error, after this transient, is approximately zero. Figures 3.10, 3.12, 3.14, and 3.16 show that there is a minor transient time in relation to Figures 3.9, 3.11, 3.13, and 3.15, although the steady-state error behaves similarly in both cases. Figures 3.17 and 3.18 show that the Frobenius norm of the estimated weights performs are similar after the transient, due to large initial uncertainty, and in both cases seems to converge to approximately 2,that is, it seems to converge to a constant. Thus, the algorithm is stable and the residual state error converges to a neighborhood of the origin.

The result is as expected, since γ_W, which is directly proportional to α, was the only parameter changed in the two cases. Consequently, it is expected that an increasing of γ_W does not affect the residual state error, and the transient time will be reduced. As can be seen, in the simulations Figures 3.10, 3.12, 3.14, and 3.16, there is a shorter transient time than in the Figures 3.9, 3.11, 3.13, and 3.15, and the residual state error appears to be unchanged in a steady state. In this way, it can be said that the results were as expected. In these simulations, L and P are chosen to reduce the residual error to desired one. It would be possible to have changed these values if a minor residual error is required. Note that the identification performed well even in the presence of disturbances from $t = 5s$.

3.4.3 Hyperchaotic Welding System

Consider the hyperchaotic welding system [109], which is described by

$$\begin{aligned}
\dot{x_1} &= x_3 - \left(\frac{c_1}{\pi r_e^2}x_2 + \frac{c_2\rho}{\pi r_e^2}x_1x_2^2\right)\\
\dot{x_2} &= \frac{1}{L_s}\left(u_2 - (R_a + R_s + \rho x_1)x_2 - V_0 - E_a(l_c - x_1)\right)\\
\dot{x_3} &= \frac{1}{\tau_m}(k_m u_1 - x_3)\\
\dot{x_4} &= R_a\frac{1}{L_s}\left(u_2 - (R_a + R_s + \rho x_1)x_2 - V_0 - E_a(l_c - x_1)\right) - E_a\left(x_3 - (\frac{c_1x_2}{\pi r_e^2)} + \frac{c_2\rho}{\pi r_e^2})\right)
\end{aligned} \tag{3.67}$$

where x_1, x_2, x_3, and x_4 are state variables, u_1 and u_2 are inputs, and c_1, c_2, r_e, ρ, L_s, R_a, R_s, V_0, E_a, l_c, τ_m, and k_m are parameters, being chosen (according to [109]) as $c_1 = 3.3\times10^{-10}(m^3s^{-1}A^{-1})$, $c_2 = 0.78\times10^{-10}(m^3s^{-1}\Omega^{-1}A^{-2})$, $r_e = 0.6\times10^{-3}(m)$, $\rho = 0.43(\Omega m^{-1})$, $L_s = 306\times10^{-6}(H)$, $R_a = 0.0237(\Omega)$, $R_s = 6.8\times10^{-3}(\Omega)$, $V_0 = 15.5(V)$, $E_a = 400(Vm^{-1})$, $l_c = 0.025(m)$, $\tau_m = 50\times10^{-3}(s)$, and $k_m = 1(mV^{-1}s^{-1})$. Note that system (3.67) satisfies the Assumption 1, since the state variables evolve into compact sets.

The parameters and the state variables pf a welding system are: c_1 is the melting rate constant 1; c_2 is melting rate constant 2; r_e is the electrode radius; ρ is the resistivity of the electrode; L_s is the total inductance; R_a is the arc resistance; R_s is the total wire resistance; V_0 is the constant charge zone; E_a is the arc length factor; l_c is the contact tip to work piece distance; τ_m is the motor time constant; k_m is the motor steady state gain; u_1 is the motor armature voltage and u_2 is the open circuit voltage; x_1 is the stick out, x_2 is the welding current; x_3 is the welding wire speed, and x_4 is the arc voltage.

To identify the uncertain system (3.67), the proposed identification model (3.9) and the adaptive law (3.29) were implemented. The initial conditions for the chaotic system and for the identification model were $x_1(0) = 0.01$, $x_2(0) = 0$, $x_3(0) = 0$, $x_4(0) = 0.01$, $\hat{x}_1(0) = 0$, $\hat{x}_2(0) = 0$,

$\hat{x}_3(0) = 0$, $\hat{x}_4(0) = 0$, and $\hat{W}(0) = 0$ in order to evaluate the performance of the proposed algorithm under adverse initial conditions. The design matrices were chosen as

$$L = 200 \begin{bmatrix} 20 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix} \tag{3.68}$$

$$P = 2000 \begin{bmatrix} 20 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 6 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix} \tag{3.69}$$

The nonlinear vector function σ is equal to

$$\sigma = \begin{bmatrix} s(x_1) \\ s(x_2) \\ s(x_3) \\ s(x_4) \\ s(x_1)s(x_2) \\ s(x_2)^2 \\ s(x_5) \\ s(x_6) \end{bmatrix} \tag{3.70}$$

The sigmoidal function used is a logistic function and is equal to $s(.) = \frac{5}{1+e^{-0.5(.)}}$. The design parameters γ_0 and γ_W were chosen as $\gamma_0 = 0.001$ and $\gamma_W = 0.001$. W_0 was chosen as

$$W_0^T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.71}$$

Figures 3.19 and 3.20 show the inputs, figures 3.21 - 3.24 show the performances obtained in the estimation of the four states, figures 3.25 - 3.28 show the performances obtained in the estimation of the four states in monolog scale and Figure 3.28 shows the Frobenius norm associated with the estimated weights matrix.

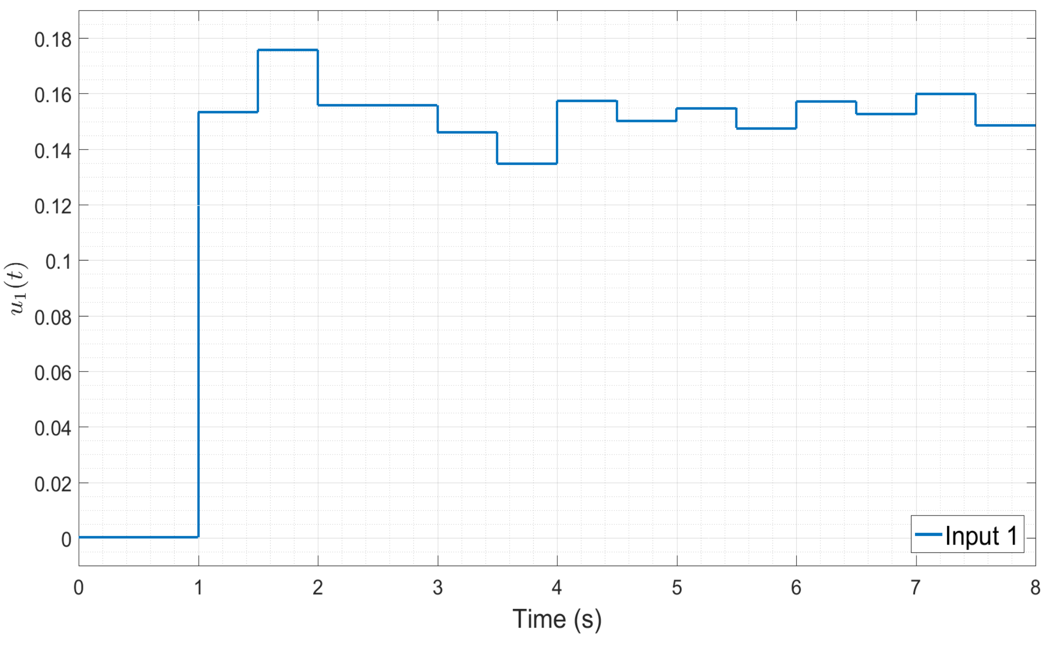


Figure 3.19: Input 1

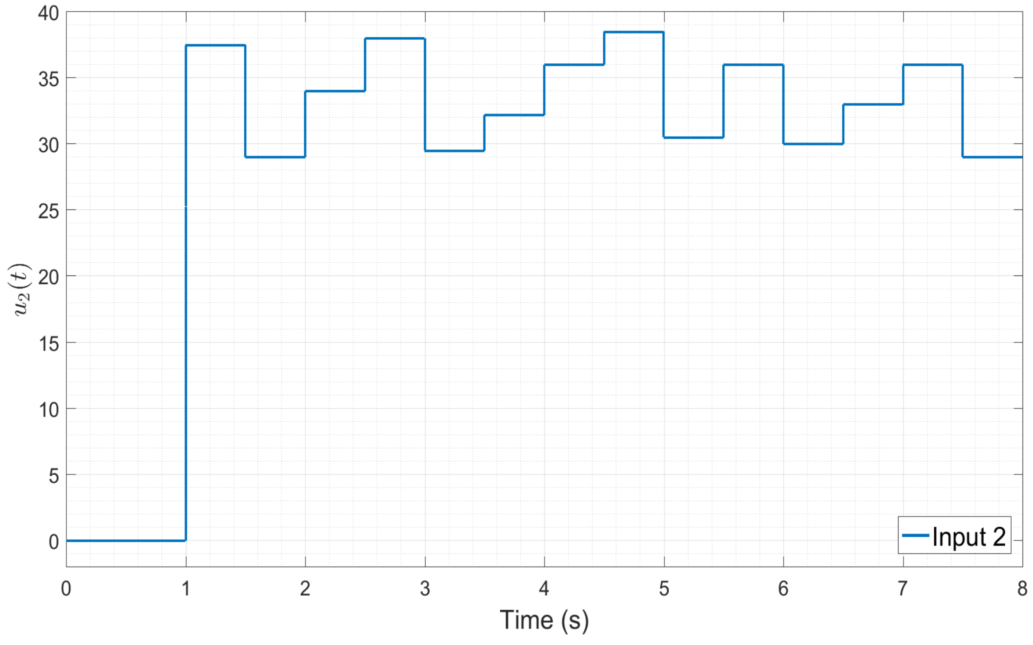


Figure 3.20: Input 2

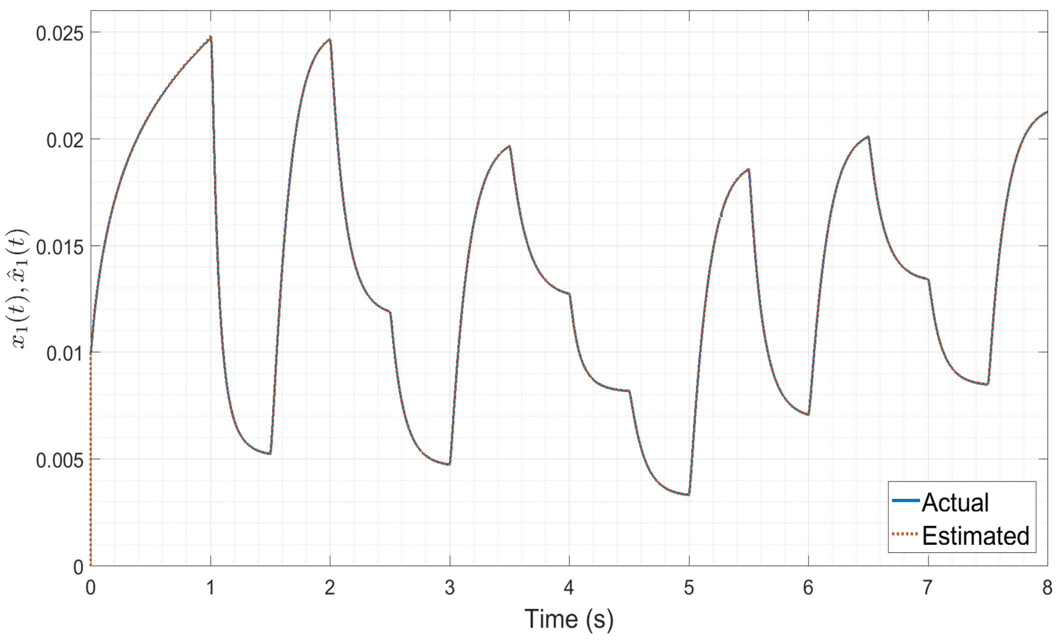


Figure 3.21: Performance in the estimation of x_1

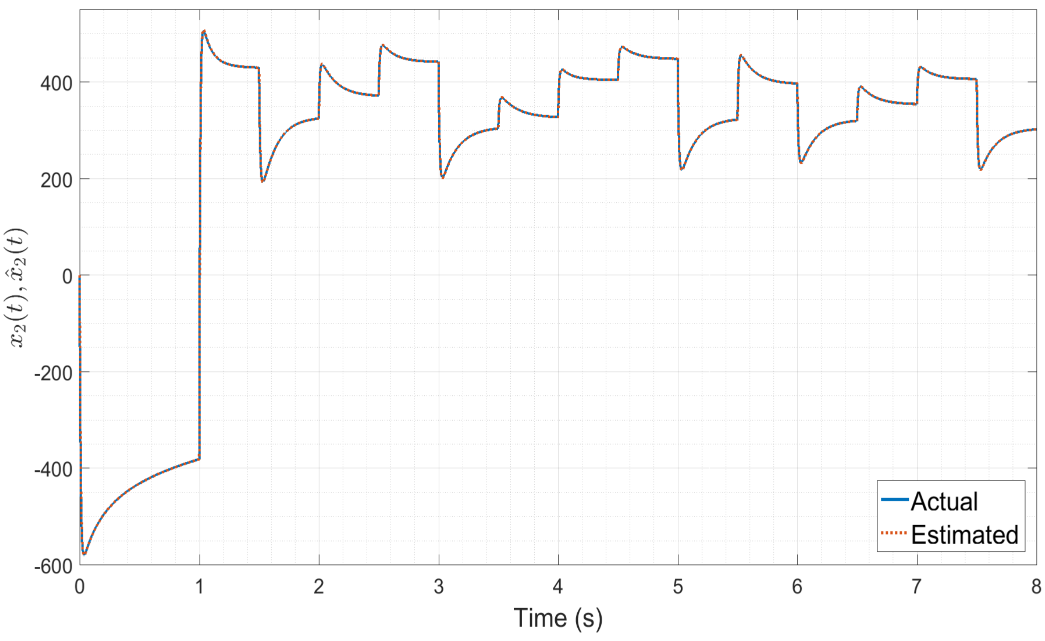


Figure 3.22: Performance in the estimation of x_2

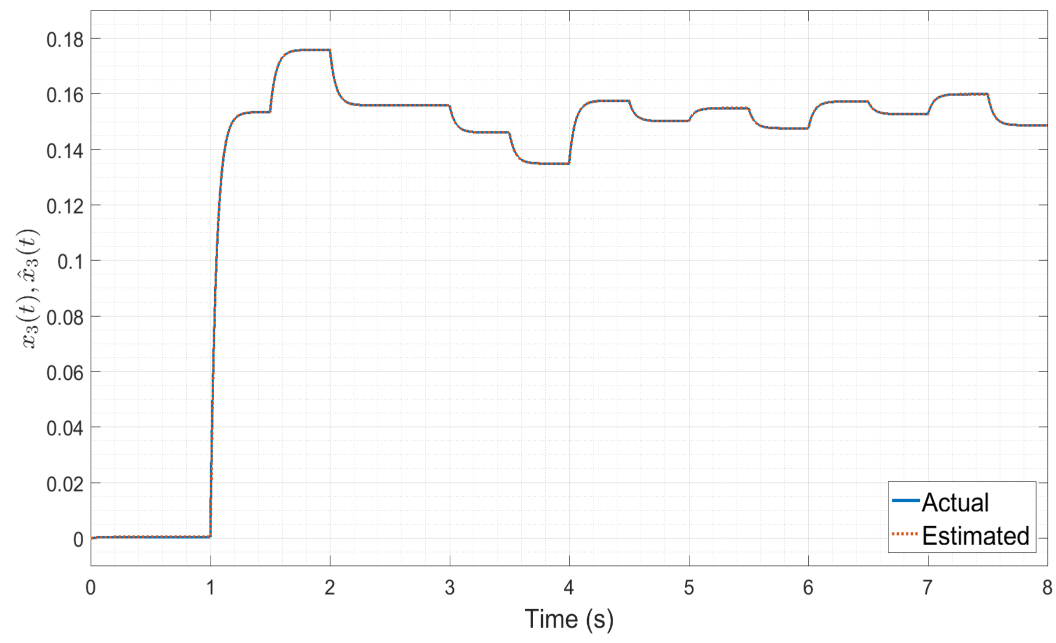


Figure 3.23: Performance in the estimation of x_3

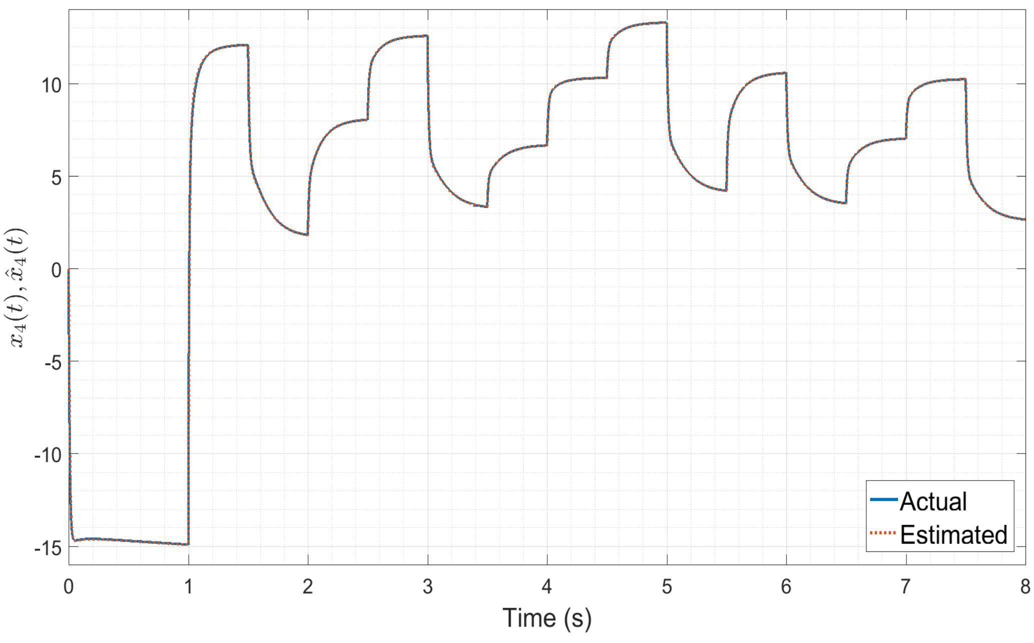


Figure 3.24: Performance in the estimation of x_4

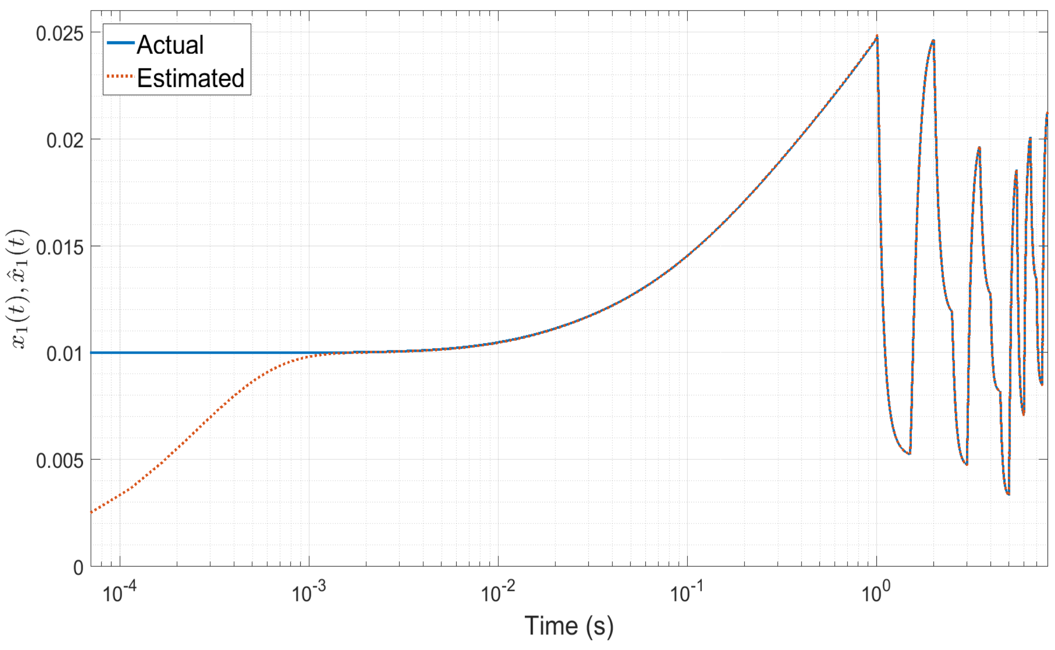


Figure 3.25: Performance in the estimation of x_1 - Monolog graph

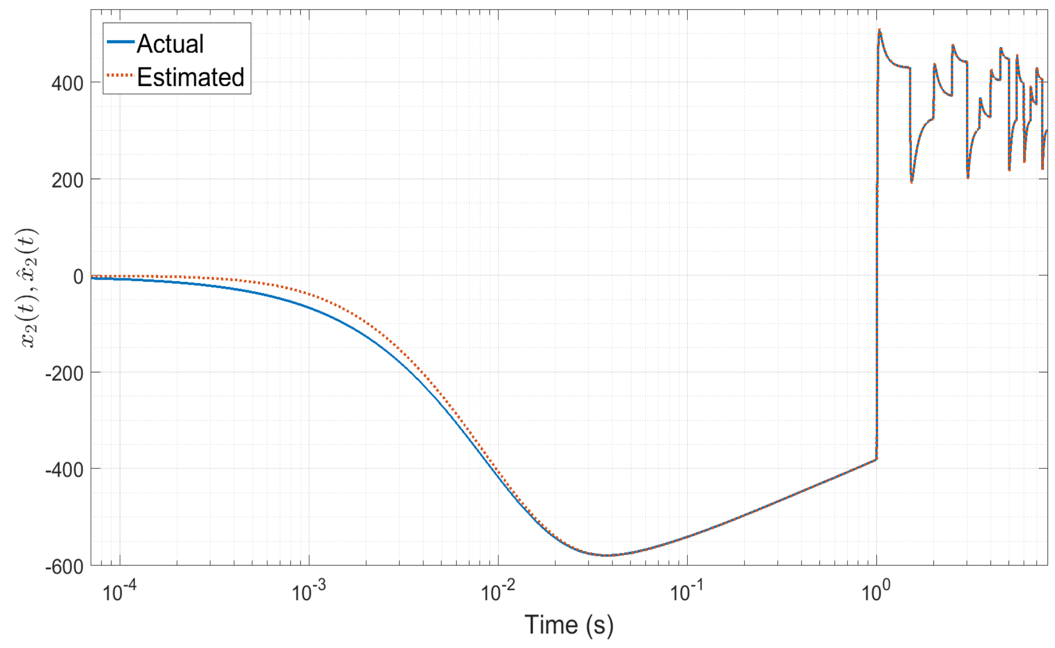


Figure 3.26: Performance in the estimation of x_2 - Monolog graph

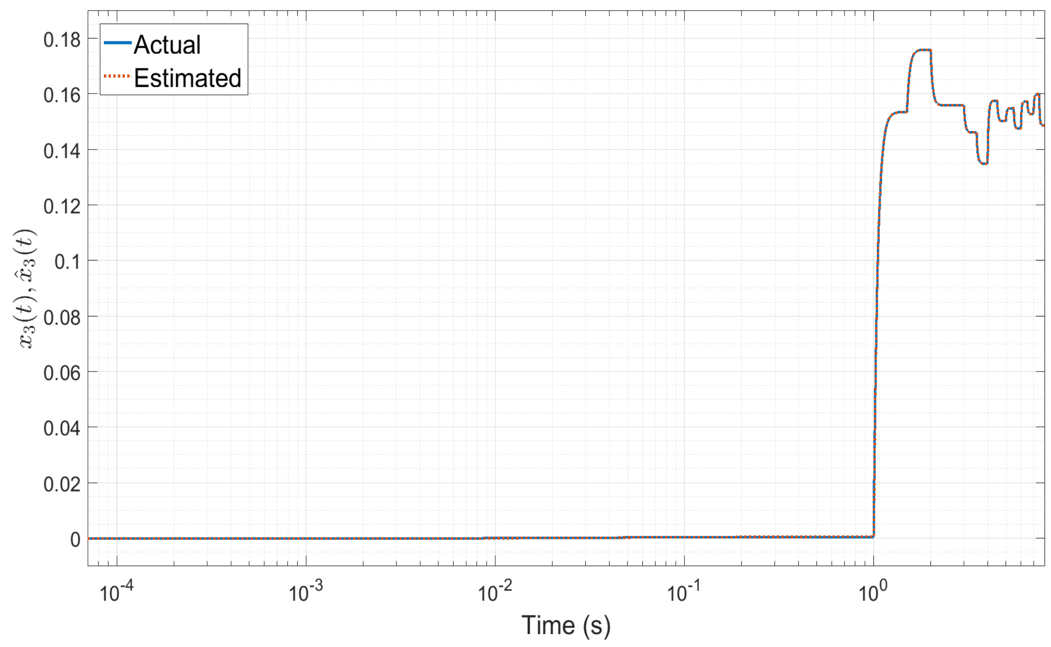


Figure 3.27: Performance in the estimation of x_3 - Monolog graph

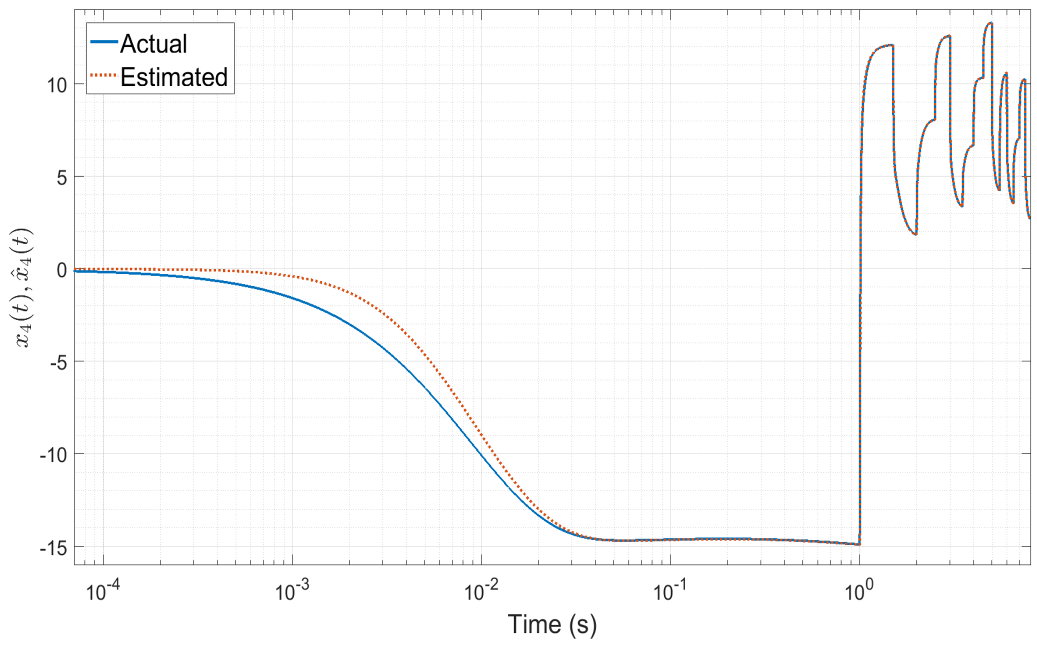


Figure 3.28: Performance in the estimation of x_4 - Monolog graph

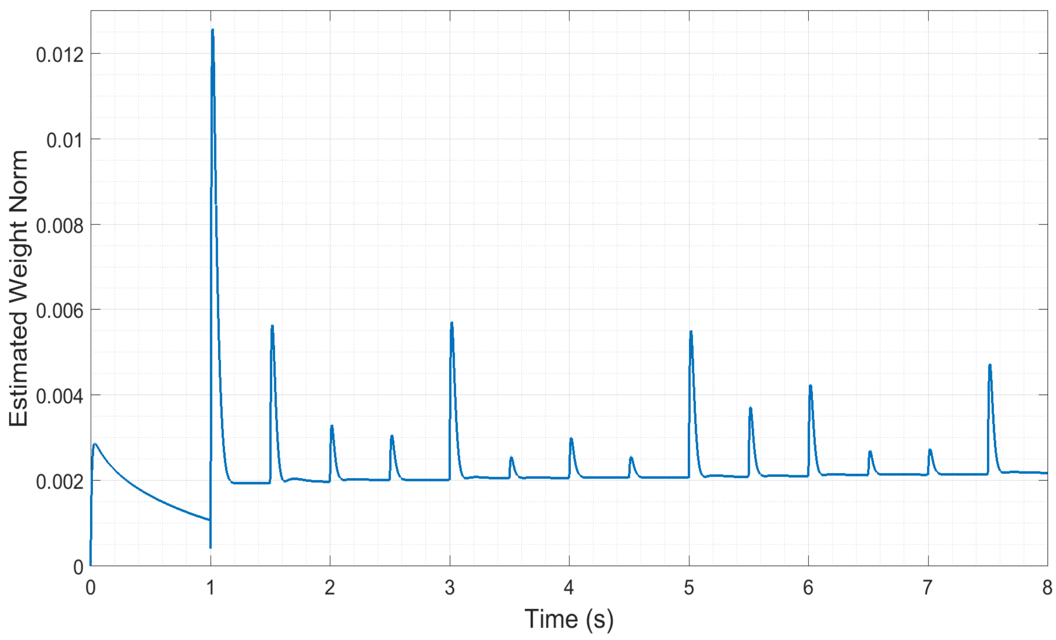


Figure 3.29: Frobenius norm of the estimated weight matrix W when $\gamma_W = 0.5$

It is observed that the simulations confirm the theoretical results: the algorithm is stable and the residual error of state converges to a neighborhood of the origin. Figures 3.17 and 3.18 show that the Frobenius norm of the estimated weights after the transient, due to large initial uncertainty, seems to converge to approximately 0.35,that is, it seems to converge to a constant. The result is as expected, since L and P are chosen to reduce the residual error to desired one. It would be possible to have changed these values if a minor residual error is required.

Remark 16. The identification of systems is important for welding since if there is a change of variable, the behavior of the model can change drastically and the existence of algorithms that allow to identify models of welding are denandaded.

3.5 Summary

In this chapter, by using neural networks and Lyapunov methods, a scheme was proposed to identify uncertain nonlinear systems. The proposed scheme is based on explicit feedback to ensure the convergence of the residual state error to a set defined from design parameters. The proposed scheme allows controlling the transient time of the system identification. In other words, it is possible to control the duration of the transient regime from one parameter and at the same time it is possible to control the size of the residual state error from another parameter independently. It was confirmed in the simulations that changes in some design parameters can be used to change the transient independently of the residual state error. An application of the identifier was done in a welding system with chaotic behavior. It is noteworthy that the proposed scheme can be used in identificaition of uncertain systems with great benefits.

Chapter 4

Adaptive Observer Design of Uncertain Systems With Control of Residual Error

Some works based on neural networks have been proposed to estimate adaptively the states of uncertain systems [48, 82, 89]. However, they are subject to several conditions such as previous knowledge of upper bounds for the weight and approximation errors, ideal switching, and previous sample data for an off-line learning phase, which difficult their application. In addition, no work was found in which the duration of the transient time can be adjusted arbitrarily and independently of the size of the steady-error. In this chapter, an adaptive observer for uncertain nonlinear systems in the presence of disturbances is proposed in order to avoid the above mentioned limitations. Based on a neural Luenberger-like observer, scaling, and Lyapunov theory, an adaptive scheme is proposed to make ultimately bounded the on-line observer error and to allow the adjustment of the transient duration from a design parameter. Besides, it is shown that the scaling of unknown nonlinearities, previous to the neural approximation, has a positive impact on performance and application of our algorithm, since it allows the residual state error manipulation without any additional linear matrix inequality solution. To validate the theoretical results, the state estimation of the Rössler oscillator system is performed.

4.1 Problem Formulation

Consider the following nonlinear differential equation

$$\dot{x} = Ax + B\left[f(x,u) + h(x,u,v,t)\right] \tag{4.1}$$

$$y = Cx \tag{4.2}$$

where $x \in X \subset \Re^n$ is the n-dimensional state vector, $u \in U \subset \Re^m$ is a m-dimensional admissible inputs vector, $y \in Y \subset \Re^q$ is a q-dimensional outputs vector, $v \in V \subset \Re^p$ is a p-dimensional vector of time varying uncertain variables, t is the time, $f : X \times U \mapsto \Re^r$ is a continuous map and $F : X \times U \times V \times [0,\infty) \mapsto \Re^r$ is a unknown disturbances vector. A ,B, and C are known matrices of appropriate dimensions. In order to have a well-posed problem, we assume that X, U, V are compact sets and f is locally Lipschitzian with respect to x in $X \times U \times V \times [0, \infty)$, such that (4.1) has a unique solution through x_0.

We assume that the following can be established:

Assumption 1. *On a region $X \times U \times V \times [0, \infty)$*

$$\|h(x, u, v, t)\| < h_0 \tag{4.3}$$

where h_0 is a positive constant

Assumption 2. *There is a symmetric positive definite $P \in \Re^{n\times n}$ matrix and a gain matrix $L \in \Re^{n\times q}$ such that*

$$P\left(A - LC\right) + \left(A - LC\right)^T P = -Q - 2\gamma_W \gamma_0 P \tag{4.4}$$

$$\delta^{-1} B^T P = C^* \tag{4.5}$$

where $\delta^{-1} = \|K\|_F^2$, $K \in \Re^{r\times r}$ is a diagonal matrix with non-zero elements, Q is a positive definite matrix, $\gamma_W > 0$, $\gamma_0 > 0$, $P = P^T > 0$, $\alpha > 0$, and C^* is in space generated by the lines of C [83].

Remark 1. The Assumption 1 imposes an upper bound on the disturbances norm. It is usual in literature.

Remark 2. The Assumption 2 imposes that the pair (C, A) is detectable and the linear part of the system is dissipative (strictly positive real [84]).

The objective is to design a continuous adaptive observer for (4.1), consequently free of chattering that does not require an off-line learning phase, prior upper bound for disturbances or errors and it is simple to apply as regards the adjustment of transient performance and steady error.

4.2 Adaptive Observer

The following lines presented follow a pattern adopted in [92]

We start by presenting the studied observer and then the observer error equation. For this, note that the system (4.1) can be written using scaling as

$$\dot{x} = Ax + B\left[Kg\left(x, u\right) + h\left(x, u, v, t\right)\right] \tag{4.6}$$

where $g(x, u) = K^{-1} f(x, u)$ and $K \in \Re^{r\times r}$ is a diagonal matrix with elements other than zero

Once the mapping $g(x, u)$ has unknown structure, it is replaced by the neural parametrization $W^* \sigma(x, u)$ plus an approximation error $\varepsilon(x, u)$. More exactly, (4.6) is rewritten as

$$\dot{x} = Ax + B\left[KW^* \sigma\left(x, u\right) + K\varepsilon\left(x, u\right) + h\left(x, u, v, t\right)\right] \tag{4.7}$$

where $\sigma(x, u)$ is a nonlinear vector function whose arguments are preprocessed by a scalar sigmoidal function $s(\cdot)$ and$W^* \in \Re^{n\times L}$ is the "optimal" or ideal matrix, only required for analytical purposes ,which can be defined as

$$W^* := \underset{(\hat{W}\in\Gamma)}{\arg\min} \left\{ \left\| g(x, u) - \hat{W}\sigma(x, u) \right\|_\infty \right\} \tag{4.8}$$

where $x \in X$, $u \in U$, $\Gamma = \left\{ \hat{W} \in \Re^{n\times L} : ||\, \hat{W}\, ||_F < \alpha_{\hat{w}} \right\}$, $\alpha_{\hat{w}}$ is a strictly positive constant, $\hat{W}$ is an estimate of W^* and $\varepsilon(x, u)$ is an approximation error term, corresponding to W^*, which can be defined as

$$\varepsilon(x, u) := g(x, u) - W^*\sigma(x, u) \tag{4.9}$$

The approximation, reconstruction, or modeling error ε in (4.9) is a quantity that exists due to the incapacity of LPNNs to match the unknown map $g(x, u)$.

Assumption 3. *On a region $X \times U$, the approximation error is upper bounded by*

$$\|\varepsilon(x, u)\| < \varepsilon_0 \tag{4.10}$$

where $\varepsilon_0 \geq 0$.

Remark 3. The model (4.1) and the parametrization (4.7) are mainly motivated to emphasize that the studied observer is also valid in the case of sudden changes in dynamics that may result from failures, equipment wear or changes in operating conditions. If there are no such conditions, the disturbance depends exclusively on t. The structure (4.7) suggests a Luenberger observer of the form

$$\dot{\hat{x}} = A\hat{x} + BK\hat{W}\sigma(\hat{x}, u) - LC(\hat{x} - x) \tag{4.11}$$

where $\hat{x}$ is the estimated state and $C(\hat{x} - x) = C\tilde{x} = \tilde{y}$ is the output estimation error.

Remark 4. Note in (4.11) that the scaling matrix K provides an additional degree of freedom for adjustment of the transient $\hat{x}$. In Section 4.4 will be shown that the matrix K is directly related to the residual error of observation.

Remark 5. Aiming to establish a relationship between the estimated weights and the observation error, the learning law is defined as $\dot{\hat{W}} = \rho(\hat{x}, u, C^*\tilde{x})$, where $\tilde{x} = \hat{x} - x$ is the observation error. Although the state error is not available for measurement, the feedback $C^*\tilde{x}$ can be determined in function of which is available.

Remark 6. Note that there is a matrix T such that $C^* = TC$ because C^* is constructed from C. Then $T = C^*C^+$, where C^+ is the pseudo-inverse [89] of C because this matrix satisfies the equation $C^* = TC$ and consequently $C^*\tilde{x} = TC\tilde{x} = T\tilde{y}$ [83, 89].

Based on (4.7) and (4.11) we obtain the observation equation error

$$\dot{\tilde{x}} = (A - LC)\,\tilde{x} + B\left[K\tilde{W}\hat{\sigma} + KW^{*}\tilde{\sigma} - K\varepsilon\,(x,u) - h\,(x,u,v,t)\right] \tag{4.12}$$

where $\hat{\sigma} = \sigma(\hat{x}, u)$ and $\tilde{\sigma} = \sigma(\hat{x}, u) - \sigma(x, u)$.

4.3 Adaptive Laws and Stability Analysis

Before presenting the main theorem, We state some facts, which will be used in the stability analysis.

Fact 1. In our problem, the following equation is valid:

$$tr\left(\tilde{W}^{T}K^{T}C^{*}\tilde{x}\hat{\sigma}^{T}\right) = \tilde{x}^{T}PBK\tilde{W}\hat{\sigma} \tag{4.13}$$

Fact 2. Let $W^{*}, W_0, \hat{W}, \tilde{W} \in \Re^{n\times L}$. Then, with the definition of $\tilde{W} = \hat{W} - W^{*}$, the following equations are true:

$$2tr\left[\tilde{W}^{T}\left(\hat{W} - W_0\right)\right] = \left\|\tilde{W}\right\|_F^2 + \left\|\hat{W} - W_0\right\|_F^2 - \left\|W^{*} - W_0\right\|_F^2 \tag{4.14}$$

Fact 3. Let $A \in \Re^{c\times d}$, $b \in \Re^{c}$, where $c > 0$ and $d > 0$ are whole numbers. Then, the following expressions are true:

$$tr\left(A^{T} + A\right) = tr\,(2A) = 2tr\,(A) \tag{4.15}$$

$$-\,b^{T}Ab \le -b^{T}\lambda_{min}\,(A)\,b \tag{4.16}$$

where $\lambda(A)$ is its eigenvalues

Fact 4. Whereas that $a, b,$ and $c \in \Re^{+}$, so

$$a\left\|\tilde{x}\right\|^2 - b\left\|\tilde{x}\right\| - c > 0 \tag{4.17}$$

$$\left\|\tilde{x}\right\|^2 - \frac{b}{a}\left\|\tilde{x}\right\| > \frac{c}{a} \tag{4.18}$$

$$\left\|\tilde{x}\right\|^2 - \frac{b}{a}\left\|\tilde{x}\right\| + \frac{b^2}{4a^2} > \frac{c}{a} + \frac{b^2}{4a^2} \tag{4.19}$$

$$\left(\left\|\tilde{x}\right\| - \frac{b}{2a}\right)^2 > \frac{4ac + b^2}{4a^2} \tag{4.20}$$

$$\left\|\tilde{x}\right\| - \frac{b}{2a} > \frac{\pm\sqrt{4ac + b^2}}{2a} \tag{4.21}$$

$$\|\tilde{x}\| > \frac{b \pm \sqrt{4ac + b^2}}{2a} \tag{4.22}$$

As $b - \sqrt{4ac + b^2} < 0$ and $\|\tilde{x}\| \geq 0$, this is an invalid solution, so

$$\|\tilde{x}\| > \frac{b + \sqrt{4ac + b^2}}{2a} \tag{4.23}$$

$$\|\tilde{x}\| > \frac{\frac{b}{2} + \sqrt{ac + \left(\frac{b}{2}\right)^2}}{a} \tag{4.24}$$

Fact 5. Whereas that a, b, and $c \in \Re^+$, so

$$m\left(\tilde{x}\right) = -a\left\|\tilde{x}\right\|^2 + b\left\|\tilde{x}\right\| + c \tag{4.25}$$

The derivative of equation (4.25) is equal to

$$\dot{m} = -2a\left\|\tilde{x}\right\| + b \tag{4.26}$$

The maximum value of (4.25) occurs when $\dot{m} = 0$

$$\|\tilde{x}\| = \frac{b}{2a} \tag{4.27}$$

Replacing this value in (4.25)

$$m = -a\left(\frac{b}{2a}\right)^2 + b\left(\frac{b}{2a}\right) + c \tag{4.28}$$

$$m) = -\frac{b^2}{4a} + \frac{2b^2}{4a} + c \tag{4.29}$$

Thus, the maximum value of (4.25) is equal to

$$m\left(\tilde{x}\right) = \frac{4ac + b^2}{4a} \tag{4.30}$$

We now state and prove the main theorem of this chapter.

Theorem 4.3.1. *Consider the class of general nonlinear systems described by (4.1)-(4.2), which satisfies Assumptions 1-3 and the learning law*

$$\dot{\hat{W}} = -2\gamma_W\left[\gamma_0(\hat{W} - W_0) + K^T C^* \tilde{x}\sigma(\hat{x}, u)^T\right] \tag{4.31}$$

Where W_0 is a constant matrix

Then, the signals erros $\tilde{x}$, $\tilde{W}$ are bounded and $\tilde{x}$ is uniformly ultimately bounded with ultimate bound ρ_2, where $\rho_2 = \frac{\frac{b}{2}+\sqrt{\lambda_{min}(Q)c+(\frac{b}{2})^2}}{\lambda_{min}(Q)}$, $b = 2\|K\|_F^{-1}\|C^*\|_F\left(\beta_\sigma\|W^*\|_F+\bar{\varepsilon}_0+\|K\|_F^{-1}\bar{h}_0\right)$, and $c = \gamma_0\|W^*-W_0\|_F^2$.

Proof. Consider the Lyapunov function candidate

$$V = \tilde{x}^T P\tilde{x} + \frac{tr\left(\tilde{W}^T\gamma_W^{-1}\tilde{W}\right)}{2} \tag{4.32}$$

By Deriving (4.32), we obtain

$$\dot{V} = \dot{\tilde{x}}^T P\tilde{x} + \tilde{x}^T P\dot{\tilde{x}} + \frac{\gamma_W^{-1} tr\left(\tilde{W}^T\dot{\tilde{W}} + \dot{\tilde{W}}^T\tilde{W}\right)}{2} \tag{4.33}$$

$$\dot{V} = \tilde{x}^T P\dot{\tilde{x}} + (\tilde{x}^T P\dot{\tilde{x}})^T + \frac{\gamma_W^{-1} tr\left[\tilde{W}^T\dot{\tilde{W}} + \left(\tilde{W}^T\dot{\tilde{W}}\right)^T\right]}{2} \tag{4.34}$$

Using equation (4.15), this results

$$\dot{V} = \tilde{x}^T P\dot{\tilde{x}} + (\tilde{x}^T P\dot{\tilde{x}})^T + \gamma_W^{-1} tr\left(\tilde{W}^T\dot{\tilde{W}}\right) \tag{4.35}$$

Replacing equations (4.12) and (4.31)

$$\begin{aligned}\dot{V} &= \tilde{x}^T P\left[(A-LC)\tilde{x} + B\left(K\tilde{W}\hat{\sigma} + KW^*\tilde{\sigma} - K\varepsilon - h\right)\right]\\ &+ \left\{\tilde{x}^T P\left[(A-LC)\tilde{x} + B\left(K\tilde{W}\hat{\sigma} + KW^*\tilde{\sigma} - K\varepsilon - h\right)\right]\right\}^T\\ &+ \gamma_W^{-1} tr\left\{\tilde{W}^T\left[-2\gamma_W\left(\gamma_0\left(\hat{W}-W_0\right) + K^T C^*\tilde{x}\hat{\sigma}^T\right)\right]\right\}\end{aligned} \tag{4.36}$$

$$\begin{aligned}\dot{V} &= \tilde{x}^T\left[(A-LC)^T P + P(A-LC)\right]\tilde{x}\\ &+ \tilde{x}^T\left[PB\left(K\tilde{W}\hat{\sigma} + KW^*\tilde{\sigma} - K\varepsilon - h\right)\right]\\ &+ \left\{\tilde{x}^T\left[PB\left(K\tilde{W}\hat{\sigma} + KW^*\tilde{\sigma} - K\varepsilon - h\right)\right]\right\}^T\\ &- 2\gamma_0 tr\left[\tilde{W}^T\left(\hat{W}-W_0\right)\right] - 2tr\left(\tilde{W}K^T C^*\tilde{x}\hat{\sigma}^T\right)\end{aligned} \tag{4.37}$$

Since $\tilde{x}^T\left[PB\left(K\tilde{W}\hat{\sigma} + KW^*\tilde{\sigma} - K\varepsilon - h\right)\right]$ is a scalar number, so the transpose of this number is itself and employing facts 1 and 2 and equation (4.4), it results

$$\begin{aligned}\dot{V} &= -\tilde{x}^T\left(Q - 2\gamma_W\gamma_0 P\right)\tilde{x} \\ &+ 2\tilde{x}^T PB\left(K\tilde{W}\hat{\sigma} + KW^*\tilde{\sigma} - K\varepsilon - h\right) \\ &- \gamma_0\left(\left\|\tilde{W}\right\|_F^2 + \left\|\hat{W} - W_0\right\|_F^2 - \left\|W^* - W_0\right\|_F^2\right) - 2\tilde{x}^T PBK\tilde{W}\hat{\sigma}\end{aligned} \tag{4.38}$$

Using equation (4.5)

$$\begin{aligned}\dot{V} &= -\tilde{x}^T Q\tilde{x} - 2\gamma_W\gamma_0\tilde{x}^T P\tilde{x} + 2\delta\tilde{x}^T C^{*T}\left(KW^*\tilde{\sigma} - K\varepsilon - h\right) \\ &- \gamma_0\left(\left\|\tilde{W}\right\|_F^2 + \left\|\hat{W} - W_0\right\|_F^2 - \left\|W^* - W_0\right\|_F^2\right)\end{aligned} \tag{4.39}$$

Turning to an inequality, taking into account (4.16) and assuming that $\beta_\sigma = sup\ \tilde{\sigma}\left(\hat{x}(t), x(t), u(t)\right)$, where $t \geq 0$

$$\begin{aligned}\dot{V} &\leq -\lambda_{min}\left(Q\right)\|\tilde{x}\|^2 - 2\gamma_W\gamma_0\tilde{x}^T P\tilde{x} + 2\delta\left\|C^*\tilde{x}\right\|\left(\beta_\sigma\|K\|_F\|W^*\|_F + \|K\|_F\|\varepsilon\| + \|h\|\right) \\ &- \gamma_0\left(\left\|\tilde{W}\right\|_F^2 + \left\|\hat{W} - W_0\right\|_F^2 - \left\|W^* - W_0\right\|_F^2\right)\end{aligned} \tag{4.40}$$

Considering that $\|C^*\tilde{x}\| \leq \|C^*\|_F\|\tilde{x}\|$, $\|\varepsilon\| < \bar{\varepsilon}_0$, $\|h\| < \bar{h}_0$, rembering that $\delta^{-1} = \|K\|_F^2$, and rearranging (4.40) implies

$$\begin{aligned}\dot{V} &\leq -\|\tilde{x}\|^2\left[\lambda_{min}\left(Q\right)\right] + \|\tilde{x}\|\left[2\|K\|_F^{-1}\|C^*\|_F\left(\beta_\sigma\|W^*\|_F + \bar{\varepsilon}_0 + \|K\|_F^{-1}\bar{h}_0\right)\right] \\ &+ \gamma_0\|W^* - W_0\|_F^2 - \gamma_0\left\|\tilde{W}\right\|_F^2 - \gamma_0\left\|\hat{W} - W0\right\|_F^2 - 2\gamma_W\gamma_0\tilde{x}^T P\tilde{x}\end{aligned} \tag{4.41}$$

Assuming that $a = \lambda_{min}\left(Q\right)$, $b = 2\|K\|_F^{-1}\|C^*\|_F\left(\beta_\sigma\|W^*\|_F + \bar{\varepsilon}_0 + \|K\|_F^{-1}\bar{h}_0\right)$, $c = \gamma_0\|W^* - W_0\|_F^2$, where $a \geq 0$, $b \geq 0$, and $c \geq 0$, then

$$\begin{aligned}\dot{V} &\leq -a\|\tilde{x}\|^2 + b\|\tilde{x}\| + c \\ &- \gamma_0\left\|\tilde{W}\right\|_F^2 - \gamma_0\left\|\hat{W} - W0\right\|_F^2 - 2\gamma_W\gamma_0\tilde{x}^T P\tilde{x}\end{aligned} \tag{4.42}$$

Case 1. For analysis of the limitation of $\tilde{W}$, resuming (4.42) and disregarding some negative terms

$$\dot{V} \leq -a\|\tilde{x}\|^2 + b\|\tilde{x}\| + c - \gamma_0\left\|\tilde{W}\right\|_F^2 \tag{4.43}$$

Using fact 5, we have

$$\dot{V} \leq \frac{4ac + b^2}{4a} - \gamma_0\left\|\tilde{W}\right\|_F^2 \tag{4.44}$$

Hence, $\dot{V} < 0$ as long as

$$\gamma_0 \left\| \tilde{W} \right\|_F^2 > \frac{4ac + b^2}{4a} \tag{4.45}$$

$$\left\| \tilde{W} \right\|_F > \pm \sqrt{\frac{4ac + b^2}{4a\gamma_0}} \tag{4.46}$$

As $-\sqrt{\frac{4ac+b^2}{4a\gamma_0}} < 0$ and $\left\| \tilde{W} \right\|_F \geq 0$, this is an invalid solution, so

$$\left\| \tilde{W} \right\|_F > \sqrt{\frac{4ac + b^2}{4a\gamma_0}} \tag{4.47}$$

$$\left\| \tilde{W} \right\|_F > \sqrt{\frac{c}{\gamma_0} + \frac{1}{a\gamma_0}\left(\frac{b}{2}\right)^2} \tag{4.48}$$

Replacing a,b, and c

$$\left\| \tilde{W} \right\|_F > \sqrt{\left\| W^* - W_0 \right\|_F^2 + \frac{\left[\left\| K \right\|_F^{-1} \left\| C^* \right\|_F \left(\beta_\sigma \left\| W^* \right\|_F + \bar{\varepsilon}_0 + \left\| K \right\|_F^{-1} \bar{h}_0 \right) \right]^2}{\gamma_0 \lambda_{min}\left(Q\right)}} = \rho_1 \tag{4.49}$$

Thus, since ρ_1 is a constant, by using Lyapunov arguments [34], we concluded that $\tilde{W}$ is uniformly ultimately bounded, with ultimate bound equal to ρ_1 . Note that if, by any reason, $\left\| \tilde{W} \right\|_F$ escapes of the residual set Ω_1, where $\Omega_1 = \left\{ \tilde{W} : \left\| \tilde{W} \right\|_F \leq \rho_1 \right\}$, $\dot{V}$ becomes negative definite again, and it forces the convergence of the weight error to the residual set Ω_1.

Case 2. For analysis of the limitation of $\tilde{x}$, resuming (4.42) and disregarding some negative terms

$$\dot{V} \leq -a \left\| \tilde{x} \right\|^2 + b \left\| \tilde{x} \right\| + c \tag{4.50}$$

Hence, $\dot{V} < 0$ as long as

$$a \left\| \tilde{x} \right\|^2 - b \left\| \tilde{x} \right\| - c > 0 \tag{4.51}$$

Using fact 4, we have

$$\left\| \tilde{x} \right\| > \frac{\frac{b}{2} + \sqrt{ac + (\frac{b}{2})^2}}{a} \tag{4.52}$$

Replacing a,b, and c

$$\begin{aligned} \|\tilde{x}\| > & \frac{\|K\|_F^{-1}\|C^*\|_F\left(\beta_\sigma\|W^*\|_F+\bar{\varepsilon}_0+\|K\|_F^{-1}\bar{h}_0\right)}{\lambda_{min}(Q)} \\ & + \frac{\sqrt{\lambda_{min}(Q)\gamma_0\|W^*-W_0\|_F^2+\left[\|K\|_F^{-1}\|C^*\|_F\left(\beta_\sigma\|W^*\|_F+\bar{\varepsilon}_0+\|K\|_F^{-1}\bar{h}_0\right)\right]^2}}{\lambda_{min}(Q)} = \rho_2 \end{aligned} \tag{4.53}$$

Thus, since ρ_2 is a constant, by using Lyapunov arguments [34], we concluded that $\tilde{x}$ is uniformly ultimately bounded, with ultimate bound equal to ρ_2 . Note that if, by any reason, $\|\tilde{x}\|$ escapes of the residual set Ω_2, where $\Omega_2 = \{\tilde{x} : \|\tilde{x}\| \leq \rho_2\}$, $\dot{V}$ becomes negative definite again, and it forces the convergence of the state error to the residual set Ω_2.

Case 3. For analysis of the transient time, resuming (4.42) and doing some manipulations

$$\begin{aligned} \dot{V} \leq & -\alpha V + \alpha V - a\|\tilde{x}\|^2 + b\|\tilde{x}\| + c \\ & -\gamma_0\left\|\tilde{W}\right\|_F^2 - \gamma_0\left\|\hat{W}-W0\right\|_F^2 - 2\gamma_W\gamma_0\tilde{x}^TP\tilde{x} \end{aligned} \tag{4.54}$$

We analyze when $-a\|\tilde{x}\|^2 + b\|\tilde{x}\| + c < 0$. This consideration is necessary, since it is in this period that the transient is occurring

$$\begin{aligned} \dot{V} \leq & -\alpha V + \alpha\left(\tilde{x}^TP\tilde{x} + \frac{\gamma_W^{-1}}{2}\left\|\tilde{W}\right\|_F^2\right) - \gamma_0\left\|\hat{W}-W0\right\|_F^2 \\ & -\gamma_0\left\|\tilde{W}\right\|_F^2 - 2\gamma_W\gamma_0\tilde{x}^TP\tilde{x} \end{aligned} \tag{4.55}$$

$$\begin{aligned} \dot{V} \leq & -\alpha V - \gamma_0\left\|\hat{W}-W0\right\|_F^2 \\ & + \left\|\tilde{W}\right\|_F^2\left(\frac{\alpha}{2\gamma_W} - \gamma_0\right) + \tilde{x}^TP\tilde{x}\left(\alpha - 2\gamma_W\gamma_0\right) \end{aligned} \tag{4.56}$$

Considering that $\alpha = 2\gamma_W\gamma_0$, then

$$\dot{V} \leq -\alpha V - \gamma_0\left\|\hat{W}-W0\right\|_F^2 \tag{4.57}$$

$$\dot{V} \leq -\alpha V \tag{4.58}$$

Using Lemma 3.2.4 [84]. It can be stated that:

$$V(t) \leq e^{-\alpha(t-t_0)}V(t_0), \qquad \forall t \geq t_0 \geq 0 \tag{4.59}$$

Assuming that $t_0 = 0$

$$V(t) \leq V(0)e^{-\alpha t} \tag{4.60}$$

$$ln\left[V(t)\right] \leq -\alpha t + ln\left[V(0)\right] \tag{4.61}$$

$$ln\left[\frac{V(t)}{V(0)}\right] \leq -\alpha t \tag{4.62}$$

$$ln\left[\frac{V(0)}{V(t)}\right] \geq \alpha t \tag{4.63}$$

$$t \leq \frac{ln\left[\frac{V(0)}{V(t)}\right]}{\alpha} \tag{4.64}$$

$\square$

Remark 7. Based on (4.53) it can be concluded that the observarion residual error can be adjusted through the scaling matrix K, since it attenuates the errors of approximation and disturbances. This is an important result as it facilitates arbitrary error of the stady-error without the need to solve any matrix inequality linear, since the last term within the right-hand bracket of (4.53) can be set via γ_0 or W_0 [92].

Remark 8. It is important to note that the choice of values of δ, and consequently of K, does not imply a new calculation of the matrices P and L in (4.4)-(4.5) to maintain the desired allocation of the eigenvalues of Q, since the matrix C^* is freely chosen on the basis of C [92]. In this way it is possible to change the residual status error value without affecting the transient duration.

Remark 9. Note that the time t in (4.64) refers to the transient regime duration in relation to the residual state error determined in (4.53). Thus, we can not say anything about the transient duration for a residual state error smaller than that.

Remark 10. It is possible to verify in (4.64) that the maximum transient duration is inversely proportional to the value of α. Since α is related to γ_0 and γ_W, it is concluded that it is possible to increase or decrease the transient time by changing these design parameters.

Remark 11. Note in equation (4.4) that the choice of different values of γ_W leads to the change of the matrices P, L, and Q. In most cases it is possible to choose matrices P and L in order to keep the matrix Q unchanged. Thus, the choice of a different γ_W will not affect the size of the residual state error and at the same time will affect the duration of the transient regime.

Remark 12. The matrix K controls the size of the residual state error and the parameter γ_W controls the duration of the transient regime independently. Thus, it is possible to decouple the steady-error of the transient performance.

Remark 13. In the equation (4.53) in the numerators there is the term $\|K\|_F^{-1}\bar{h}_0$, thus the external disturbances are present in that part. In this way, it can be stated that by changing the value of the matrix L, the size of the residual state error is also adjusted, even in the presence of limited disturbances.

Remark 14. In previous works, for example in [83, 89, 92], in spite of the residual state error is bounded, they can not increase or reduce the transient duration independently of residual state error size. To allow this, it is necessary to have a parameter related to the transient duration that does change the residual state error. In this work this was possible because the observer model and the learning law were chosen in order to allow an independent control of the transient duration.

4.4 Simulation

This section presents three examples to validate the theoretical results and to show the performance in the presence of disturbances. In the simulation, Solver ode45(Dormand-Pince) of Matlab/Simulink®, with variable-step and a relative tolerance of 1e-10 was used to obtain the numerical solutions. The observation of a chaotic system under disturbances has been proposed. An analysis is made of the duration of the transient regime in relation to the steady-error.

4.4.1 Rösller oscillator system

Consider the following chaotic system [76], which is described by

$$\begin{aligned} \dot{x_1} &= -(x_2 + x_3) + d_{x_1} \\ \dot{x_2} &= x_1 + a x_2 + d_{x_2} \\ \dot{x_3} &= b + x_3\,(x_1 - c) + d_{x_3} \\ y &= Cx \end{aligned} \tag{4.65}$$

where x_1, x_2, and x_3 are state variables, $A = \begin{bmatrix} 0 & -1 & -1 \\ 1 & a & 0 \\ 0 & 0 & -c \end{bmatrix}$, $C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, d_{x_1}, d_{x_2}, and d_{x_3} are unknown disturbances, and a, b, and c are parameters, being chosen as $a = 0.398$, $b = 2$, and $c = 4$.

To observe the uncertain system (4.65), the proposed observer model (4.11) and the adaptive law (4.31) were implemented. The initial conditions for the chaotic system and for the identification model were $x_1(0) = 2$, $x_2(0) = 1$, $x_3(0) = 2$, $\hat{x}_1(0) = 0$, $\hat{x}_2(0) = 0$, $\hat{x}_3(0) = 0$, and $\hat{W}(0) = 0$ in order to evaluate the performance of the proposed algorithm under adverse initial conditions. The nonlinear vector function σ is equal to

$$\sigma = \begin{bmatrix} s(x_1) \\ s(\hat{x_2}) \\ s(x_3) \\ s(x_1)s(\hat{x_2}) \\ s(x_3)s(\hat{x_2}) \\ s(x_1)s(x_3) \\ s(x_1)^2 \\ s(\hat{x_2})^2 \\ s(x_3)^2 \end{bmatrix} \tag{4.66}$$

The sigmoidal function used is a logistic function and is equal to $s(.) = \frac{5}{1+e^{-0.5(.)}}$. The design parameter γ_0 was chosen as $\gamma_0 = 1$, K was chosen as $K = 3$ and W_0 was chosen as

$$W_0 = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \tag{4.67}$$

The matrix is chosen as $Q = I$, where I is the identity matrix. The matrices P and L have different values depending on the γ_W value. When $\gamma_W = 0.5$

$$L = \begin{bmatrix} -0.4429 & -4 \\ -2.3808 & 3.194 \\ 10.8409 & 0 \end{bmatrix} \tag{4.68}$$

$$P = \begin{bmatrix} 5.9544 & 1.398 & 1 \\ 1.398 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \tag{4.69}$$

when $\gamma_W = 8$

$$L = \begin{bmatrix} 34.9271 & 3.5 \\ -247.046 & -41.041 \\ -32.0291 & 0 \end{bmatrix} \tag{4.70}$$

$$P = \begin{bmatrix} 83.1744 & 8.898 & 1 \\ 8.898 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \tag{4.71}$$

We found these matrix values from equations (4.4) and (4.5).A positive matrix defined P was arbitrated and the values of the arguments of the matrix L were found.

To verify the robustness of proposed method, it is considered the presence of the following disturbances: $d_{x_1} = cos(2t)$, $d_{x_2} = 2sin(t)$, and $d_{x_3} = 2sin(2t)$. To better verify each part, the disturbance was introduced from $t = 5s$.

From equation (4.62), it can be seen that the parameter α affects the transient time. We choose the parameter γ_W to control the value of α because γ_W is not related to the size of the residual state error, as can be seen in (4.53). In all states we analyzed for two different values of γ_W. In the first situation, a small γ_W was intentionally chosen to detect the transient and in the second case a high value was chosen with the intention of reducing this transient time.

In this example, the two values chosen for γ_W are $\gamma_W = 0.5$ and $\gamma_W = 8$. Figures 4.1, 4.3, and 4.4 show the performances obtained in the estimation of the three states when $\gamma_W = 0.5$. Figures 4.2, 4.4, and 4.6 show the performances obtained in the estimation of the three states when $\gamma_W = 8$. The Frobenius norm associated with the estimated weights matrix is shown in Figure 4.7 when $\gamma_W = 0.5$ and is shown in Figure 4.8 when $\gamma_W = 8$.

Note that the estimation of the state x_2 is not available for measurement.

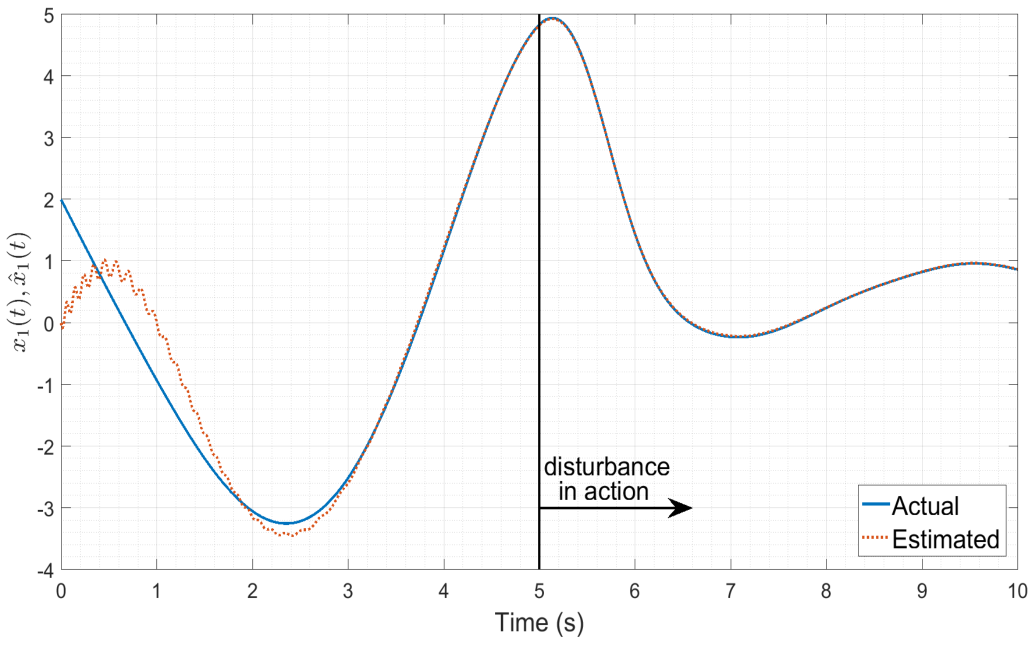


Figure 4.1: Performance in the estimation of x_1 when $\gamma_W = 0.5$

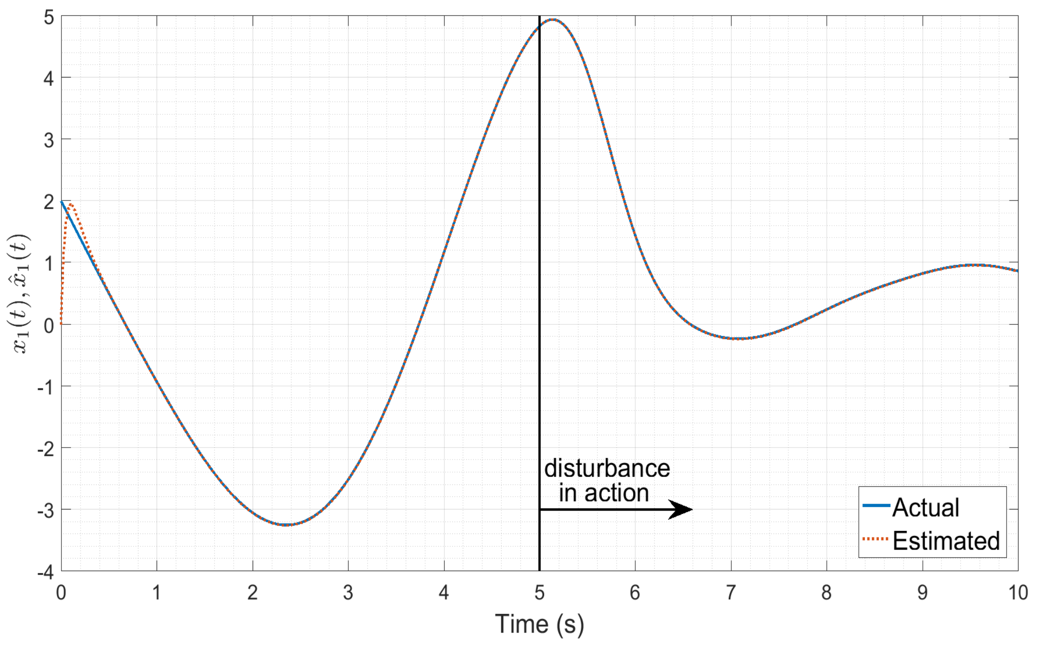


Figure 4.2: Performance in the estimation of x_1 when $\gamma_W = 8$

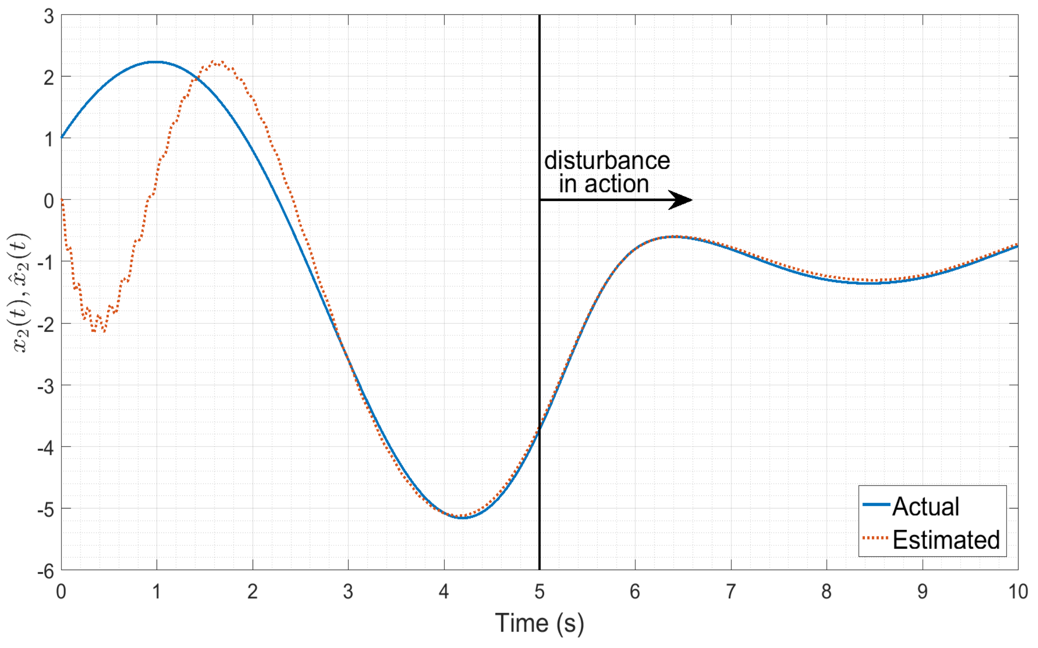


Figure 4.3: Performance in the estimation of x_2 when $\gamma_W = 0.5$

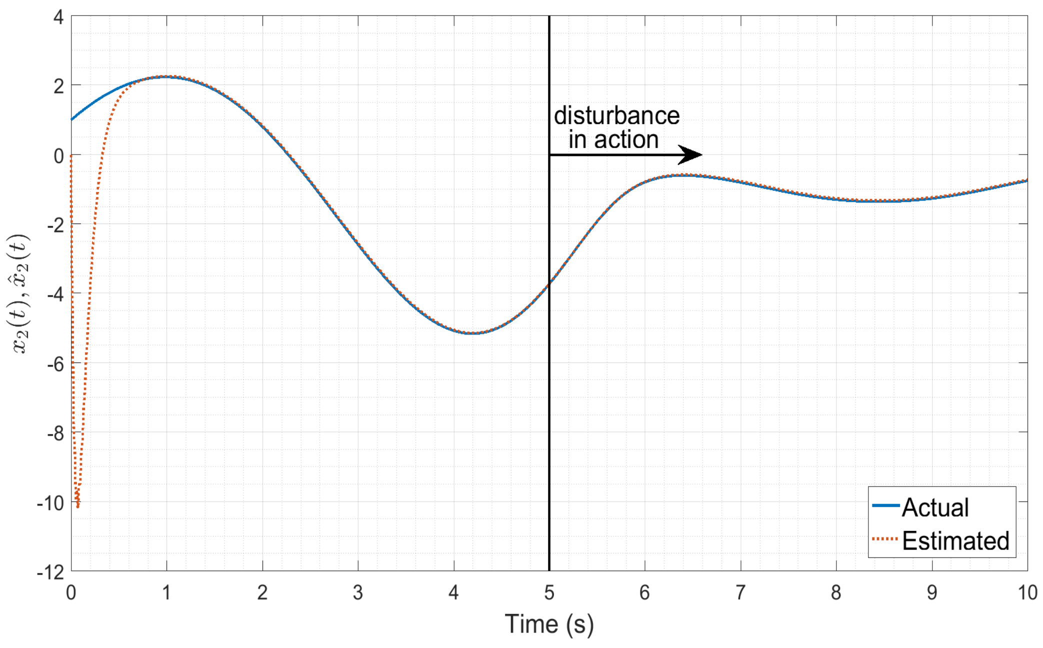


Figure 4.4: Performance in the estimation of x_2 when $\gamma_W = 8$

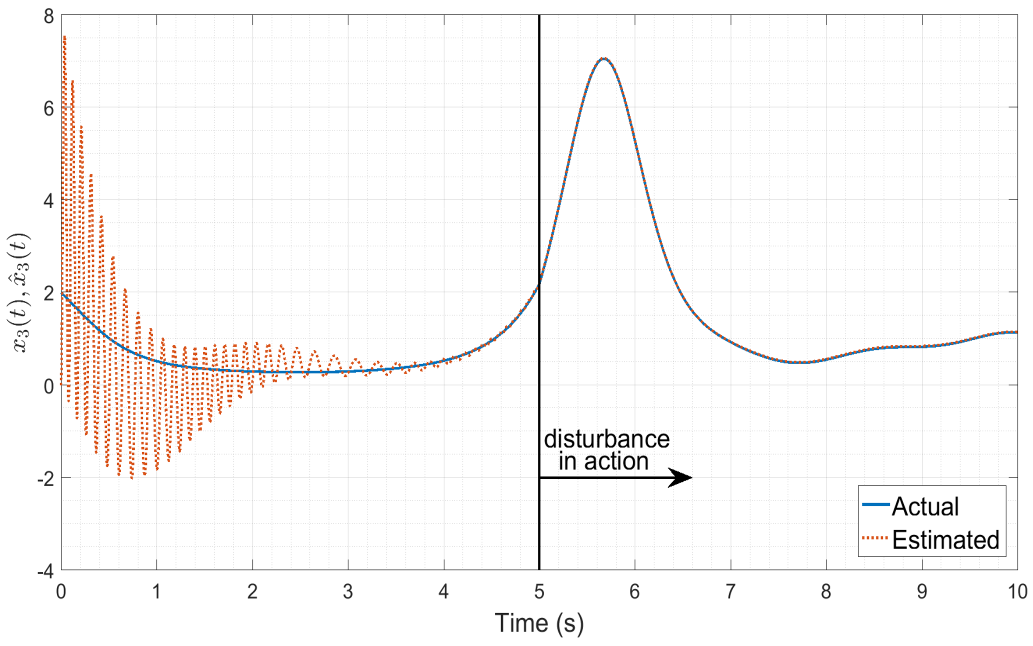


Figure 4.5: Performance in the estimation of x_3 when $\gamma_W = 0.5$

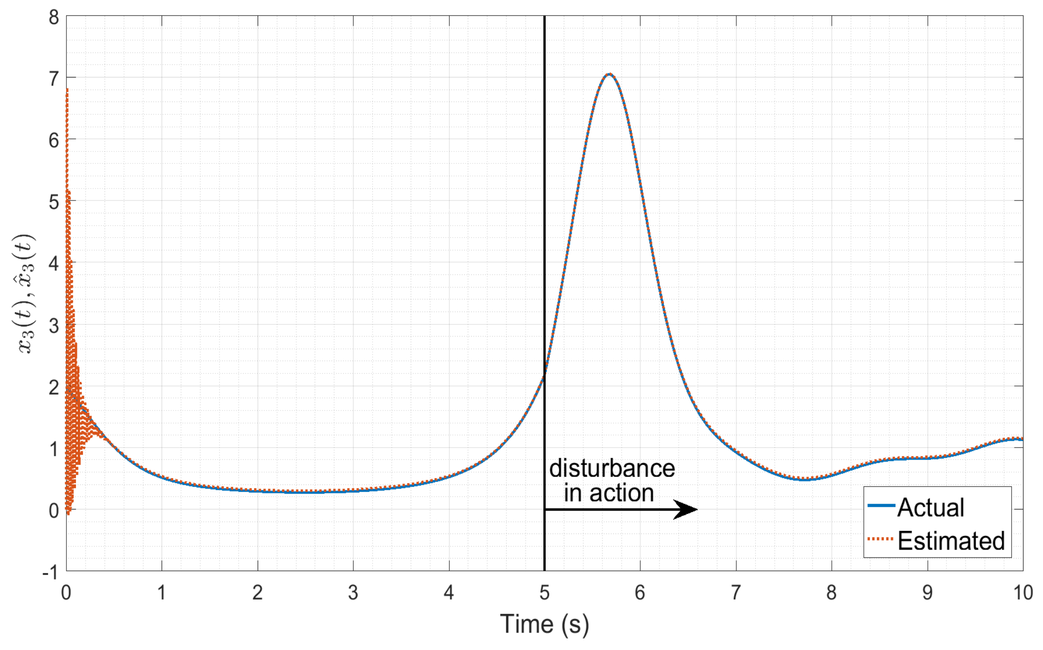


Figure 4.6: Performance in the estimation of x_3 when $\gamma_W = 8$

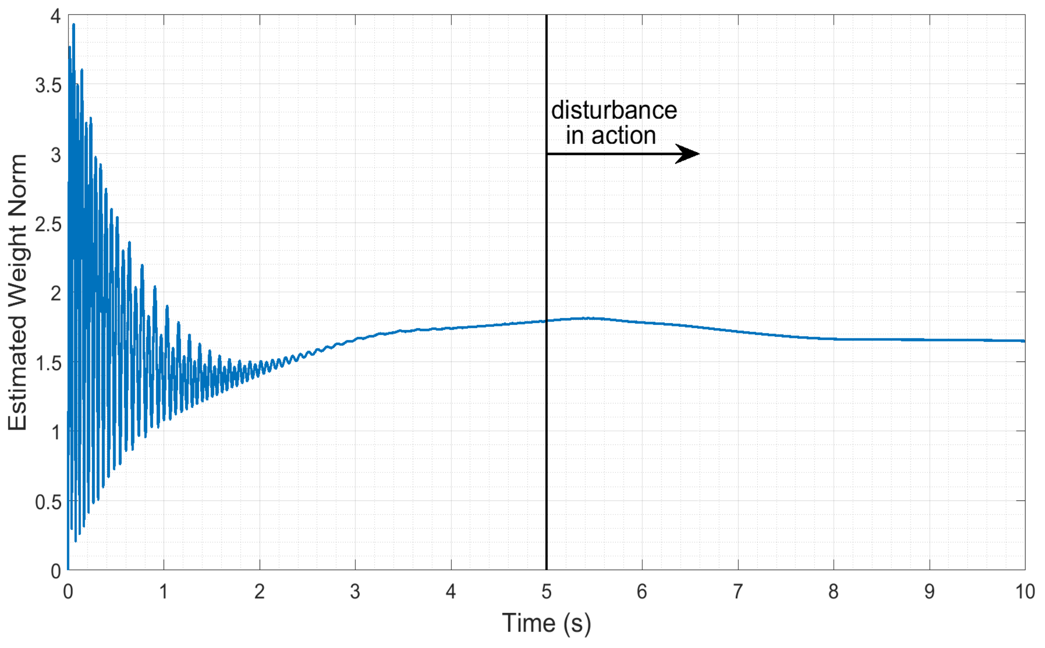


Figure 4.7: Frobenius norm of the estimated weight matrix W when $\gamma_W = 0.5$

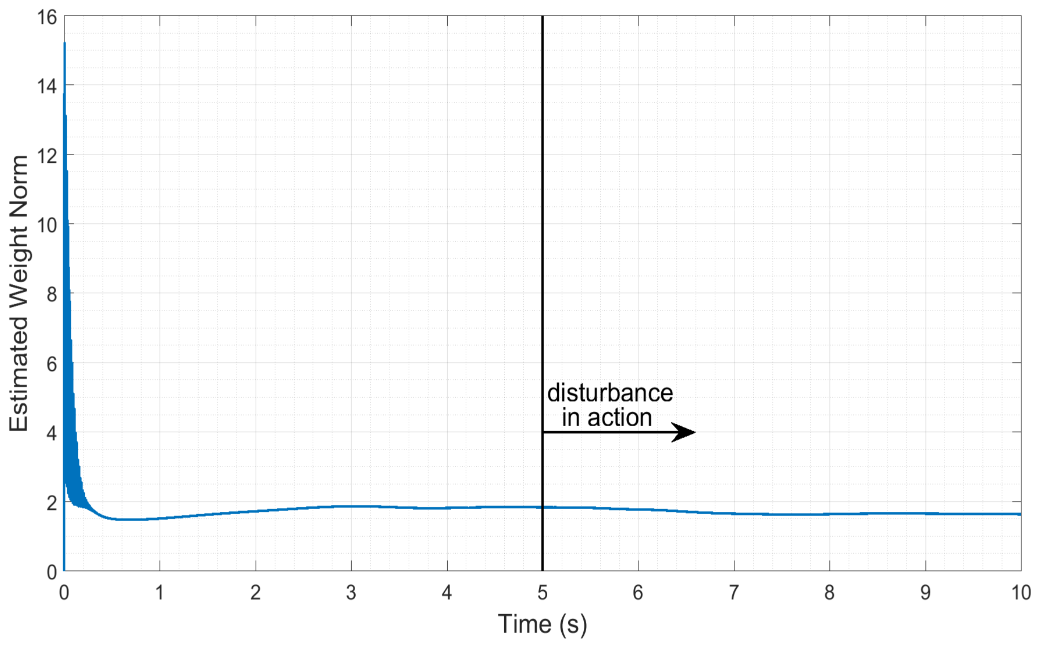


Figure 4.8: Frobenius norm of the estimated weight matrix W when $\gamma_W = 8$

Note that there is a transient time and that the residual state error, after this transient, is approximately zero. Figures 4.2, 4.4, and 4.6 show that there is a minor transient time in relation to Figures 4.1, 4.3, and 4.5, although the steady-state error behaves similarly in both cases. Figures 4.7 and 4.8 show that the Frobenius form of the estimated weights performs are similar after the transient and in both cases seems to converge to approximately 1.7,that is, it seems to converge to a constant.

The result is as expected, since γ_W, which is directly proportional to α, was the only parameter changed in the two cases. Consequently, it is expected that an increasing of γ_W does not affect the residual state error, and the transient time will be reduced. As can be seen, in the simulations Figures 4.2, 4.4, and 4.6, there is a shorter transient time than in the Figures 4.1, 4.3, and 4.5, and the residual state error appears to be unchanged in a steady state. In this way, it can be said that the results were as expected. In these simulations, L and P are chosen to reduce the residual error to desired one. It would be possible to have changed these values if a minor residual error is required. Note that the identification performed well even in the presence of disturbances from $t = 5s$.

4.5 Summary

In this chapter we studied a continuous neural adaptive observer based on Lyapunov stability theory and scaling for a class of uncertain nonlinear systems. Using the direct Lyapunov method it has been shown that the estimation errors are limited and the observation error converges to a residual set around zero, whose size is directly related to a parameter δ^{-1}. In addition, from the values chosen for the paramenter γ_w and for matrices L and P, the transient regime duration can be adjusted. The main peculiarities of the observer are : 1) it does not present a chattering, 2) it does not require a previous off-line phase, 3) it allows the adjustment of the transient and steady-error independently, and 4) it does not need the resolution of linear matrix inequalities to adjust the residual observation error . To verify the performance of the studied observer, the estimation of the states of a Rössler oscillator was considered.

Chapter 5

Neural Controller in Closed Loop with Control of Residual Error and Transient Time: An application in Welding

Many works have been done in the control [101, 102, 103, 104, 105] and identification [110, 111, 112] area of welding systems. Thus, algorithms involving welding processes have been an explored subject in the literature, but the dynamic nonlinear control subject has still been little investigated. The systems identification is useful for modeling a process or a plant of unknown parameters, which usually occurs in welding systems. It is important to note that the welding process usually is a time-variant system and nonlinear such as in [113], therefore, they are systems in which modeling tends to be more challenging. The online identification has as advantages with regard to offline identification the lack of need to store a large amount of data and the possibility of tracking time-variant parameters in time without the need to know the model structure [114].

Note that in these works [101, 102, 103, 104, 105, 112] no artificial neural networks were used for the control or identification algorithms. In [110] and [111] artificial neural networks were used, but in [110] a predictive control was proposed, not an adaptive control, and in [111] a discrete Hammerstein model was used and linear approximations were made. Thus, in none of the previous works an adaptive controller using artificial neural networks was designed to control in continuous time a non-linear welding system with an arbitrarily small tracking error. In addition, none of the previous works explored the possibility of adjusting the transient duration from a design parameter.

Therefore, based on a neural controller model with feedback, scaling, and Lyapunov based weight adjustment law, a control algorithm is proposed to make ultimately bounded the tracking error. In addition, it is desired that in this algorithm: at least one designed matrix is related to the size of the tracking error, even in the presence of bounded disturbances; and a designed parameter not related to the tracking error size is related to the transient duration. In this chapter the control of a GMAW welding system is performed to validate the theoretical results. It is interesting to note that according [115] GMAW welding processes have a chaotic behavior.

5.1 Problem Formulation

Consider the following nonlinear differential equation

$$\dot{x} = F\left(x, v, t\right) + u(e, x, x_r), \quad x\left(0\right) = x_0 \tag{5.1}$$

where $x \in X \subset \Re^n$ is the n-dimensional state vector, $v \in V \subset \Re^p$ is p-dimensional a vector of time varying uncertain variables, t is the time and $F : X \times U \times V \times [0, \infty) \mapsto \Re^n$ is a continuous map, $e(x_r, x) = x_r - x$ and x_r is an arbitrary reference line . In order to have a well-posed problem, we assume that X, U, V are compact sets and F is locally Lipschitzian with respect to x in $X \times U \times V \times [0, \infty)$, such that (5.1) has a unique solution through x_0.

We assume that the following can be established:

Assumption 1. *On a region $X \times U \times V \times [0, \infty)$*

$$\|h(x, v, t)\| \leq h_0 \tag{5.2}$$

where

$$h\left(x, v, t\right) = F\left(x, v, t\right) - f\left(x\right) \tag{5.3}$$

f is an unknown map, h are internal or external disturbances, and $\bar{h}_0$, such that $\bar{h}_0 > h_0 \geq 0$, is a unknown constant. Note that (5.2) is verified when x and u evolve on compact sets and the temporal disturbances are bounded.

Thus, except for the Assumption 1, we say that $F(x, u, v, t)$ is an unknown map and our aim is to design a controller based on NNs for (5.1) to ensure the state error convergence, which will be accomplished despite the presence of approximation error and disturbances.

5.2 Controller Model and Tracking Equation

We start by presenting the controller model and the definition of the relevant errors associated with the problem.

Let $\bar{f}$ be the best known approximation of f, $P \in \Re^{n \times n}$ a scaling matrix defined as $P = P^T > 0$, $\bar{g} = P^{-1}g$, and $g(x) = f(x) - \bar{f}(x)$.Then, by adding and subtracting $\bar{f}(x)$, the system (5.1) becomes

$$\dot{x} = \bar{f}\left(x\right) + P\bar{g}\left(x\right) + h\left(x, v, t\right) + u \tag{5.4}$$

Remark 1. It should be noted that if the designer has no previous knowledge of f, so $\bar{f}$ is simply assumed as being the zero vector. From (5.4), by using LPNNs, the nonlinear mapping $\bar{g}(x)$ can be replaced by the neural parametrization $W^{*}\sigma(x)$ plus an approximation error term $\varepsilon(x)$. More exactly, (5.4) can be rewritten as

$$\dot{x} = \bar{f}\left(x\right) + PW^{*}\sigma\left(x\right) + P\varepsilon\left(x\right) + h\left(x, v, t\right) + u(e, x, x_r) \tag{5.5}$$

where $\sigma\left(x, u\right)$ is a nonlinear vector function whose arguments are preprocessed by a scalar sigmoidal function $s(\cdot)$ and $W^{*} \in \Re^{n\times L}$ is the "optimal" or ideal matrix, only required for analytically purposes, which can be defined as

$$W^{*} := \underset{(\hat{W}\in\Gamma)}{\arg\min}\left\{\left\|\bar{g}\left(x\right) - \hat{W}\sigma\left(x\right)\right\|_{\infty}\right\} \tag{5.6}$$

where $x \in X$, $u \in U$, $\Gamma = \left\{\hat{W} \in \Re^{n\times L} : ||\,\hat{W}\,||_F \leq \alpha_{\hat{w}}\right\}$, $\alpha_{\hat{w}}$ is a strictly positive constant, $\hat{W}$ is an estimate of W^{*} and $\varepsilon(x, u)$ is an approximation error term, corresponding to W^{*}, which can be defined as

$$\varepsilon\left(x\right) := \bar{g}\left(x\right) - W^{*}\sigma\left(x\right) \tag{5.7}$$

The approximation, reconstruction, or modeling error ε in (5.7) is a quantity that exists due to the incapacity of LPNNs to match the unknown map $\bar{g}(x, u)$. Since X, U are compact sets and from (2.2), the following can be established.

Assumption 2. *On a region $X \times U$, the approximation error is upper bounded by*

$$\|\varepsilon(x, u)\| \leq \varepsilon_0 \tag{5.8}$$

where $\bar{\varepsilon_0}$, such that $\bar{\varepsilon_0} > \varepsilon_0 \geq 0$, is an unknown constant.

Remark 2. The assumption 1 is usual in robust control literature. The assumption 2 is quite natural since $\bar{g}$ is continuous and their arguments evolve on compact sets and σ satisfies (I.17).

Remark 3. Note that any σ_0, h_0, and ε_0 are the smallest constants such that (I.17), (5.2), and (5.8) are satisfied.

Remark 4. It should be noted that W^{*} and $\varepsilon(x, u)$ might be nonunique. However, the uniqueness of $||\varepsilon(x, u)||$ is ensured by (5.6).

Remark 5. It should be noted that W^{*} was defined as being the value of $\hat{W}$ that minimizes the L_{∞}- norm difference between $\bar{g}(x)$ and $\hat{W}\sigma(x)$. The scaling matrix P from (5.4) is introduced to manipulate the magnitude of uncertainties and hence the magnitude of the approximation error. This procedure improves the performance of the control process.

Remark 6. Notice that the proposed neuro-control scheme is a black-box methodology, hence the external disturbances and approximation error are related. Based on the system input and state measurements, the uncertain system (including the disturbances) is parametrized by a neural network model plus an approximation error term. However, the parametrization (5.5) is motivated by the fact that neural networks are not adequate for approximating external disturbances, since the basis function depends on the input and states, whereas the disturbances depend on the time and external variables. The aim for presenting the uncertain system in the form (5.5), where the disturbance h is explicitly considered, is also to highlight that the proposed scheme is in addition valid in the presence of unexpected changes in the systems dynamics that can emerge, for instance, due to environment change, aging of equipment or faults. We propose

$$u(\hat{W}, x, x_r, \dot{x_r}) = -\left[-Le - \gamma_W \gamma_0 e + P\hat{W}\sigma(x) - \dot{x_r}\right] \tag{5.9}$$

where e is the tracking error, $\gamma_W > 0$, $\gamma_0 > 0$, and $L \in \Re^{n\times n}$ is a positive definite feedback gain matrix introduced to attenuate the effect of the uncertainties and disturbances. It will be demonstrated that the control model (5.9) used in conjunction with a convenient adjustment law for $\hat{W}$, to be proposed in the next section, ensures the convergence of the state error to a neighborhood of the origin, even in the presence of the approximation error and disturbances, whose radius depends on the design parameters. Note that from equations (5.5) and (5.9), and assuming that $\bar{f}(x) = 0$ we can obtain

$$\begin{aligned} \dot{x} &= PW^*\sigma(x) + P\varepsilon(x) + h(x, v, t) \\ &\quad - \left[-Le - \gamma_W \gamma_0 e + P\hat{W}\sigma(x) - \dot{x_r}\right] \end{aligned} \tag{5.10}$$

by using the definition of $\tilde{W} = \hat{W} - W^*$, thus the system becomes

$$\dot{x} = -P\tilde{W}\sigma(x) + P\varepsilon(x) + h(x, v, t) + Le(x_r, x) + \gamma_W \gamma_0 e(x_r, x) + \dot{x_r} \tag{5.11}$$

Remark 7. Note that the identification model requires states are available to measure. However, the main relevance of the method is to provide a parametrization for the uncertain system (5.1) that can be later used to project adaptive control and observation schemes.

Remark 8. It should be noted that in our formulation, the LPNN is only required to approximate $P^{-1}[f(x) - \bar{f}(x)]$ (whose magnitude is often small) instead of the entire function $P^{-1}[f(x)]$. Hence, standard control methods (to obtain some previous $\bar{f}$]) can be used together with the proposed algorithm to improve performance. The estimation of loop error can be defined as

$$\dot{e} = \dot{x_r} - \dot{x} \tag{5.12}$$

so

$$\dot{e} = P\tilde{W}\sigma(x) - P\varepsilon(x) - h(x, v, t) - Le(x_r, x) - \gamma_W \gamma_0 e(x_r, x) \tag{5.13}$$

5.3 Adaptive Laws and Stability Analysis

Before presenting the main theorem, We state some facts, which will be used in the stability analysis.

Fact 1. In our problem, the following equation is valid:

$$tr\left(\tilde{W}^T e\sigma^T\right) = e^T\tilde{W}\sigma \tag{5.14}$$

Fact 2. Let $W^*, W_0, \hat{W}, \tilde{W} \in \Re^{n\times L}$. Then, with the definition of $\tilde{W} = \hat{W} - W^*$, the following equations are true:

$$2tr\left[\tilde{W}^T\left(\hat{W} - W_0\right)\right] = \left\|\tilde{W}\right\|_F^2 + \left\|\hat{W} - W_0\right\|_F^2 - \|W^* - W_0\|_F^2 \tag{5.15}$$

Fact 3. Let $A \in \Re^{c\times d}$, $b \in \Re^c$, where $c > 0$ and $d > 0$ are whole numbers. Then, the following expressions are true:

$$tr\left(A^T + A\right) = tr\left(2A\right) = 2tr\left(A\right) \tag{5.16}$$

$$-b^T Ab \leq -b^T\lambda_{min}\left(A\right)b \tag{5.17}$$

where $\lambda(A)$ is its eigenvalues

Fact 4. Whereas that a, b, and $c \in \Re^+$, so

$$a\left\|e\right\|^2 - b\left\|e\right\| - c > 0 \tag{5.18}$$

$$\left\|e\right\|^2 - \frac{b}{a}\left\|e\right\| > \frac{c}{a} \tag{5.19}$$

$$\left\|e\right\|^2 - \frac{b}{a}\left\|e\right\| + \frac{b^2}{4a^2} > \frac{c}{a} + \frac{b^2}{4a^2} \tag{5.20}$$

$$\left(\left\|e\right\| - \frac{b}{2a}\right)^2 > \frac{4ac + b^2}{4a^2} \tag{5.21}$$

$$\left\|e\right\| - \frac{b}{2a} > \frac{\pm\sqrt{4ac + b^2}}{2a} \tag{5.22}$$

$$\left\|e\right\| > \frac{b \pm \sqrt{4ac + b^2}}{2a} \tag{5.23}$$

As $b - \sqrt{4ac + b^2} < 0$ and $\|e\| \geq 0$, this is an invalid solution, so

$$\left\|e\right\| > \frac{b + \sqrt{4ac + b^2}}{2a} \tag{5.24}$$

$$\|e\| > \frac{\frac{b}{2} + \sqrt{ac + \left(\frac{b}{2}\right)^2}}{a} \tag{5.25}$$

Fact 5. Whereas that a, b, and $c \in \Re^{+}$, so

$$m(e) = -a\|e\|^2 + b\|e\| + c \tag{5.26}$$

The derivative of equation (5.26) is equal to

$$\dot{m} = -2a\|e\| + b \tag{5.27}$$

The maximum value of (5.26) occurs when $\dot{m} = 0$

$$\|e\| = \frac{b}{2a} \tag{5.28}$$

Replacing this value in (5.26)

$$m = -a\left(\frac{b}{2a}\right)^2 + b\left(\frac{b}{2a}\right) + c \tag{5.29}$$

$$m = -\frac{b^2}{4a} + \frac{2b^2}{4a} + c \tag{5.30}$$

Thus, the maximum value of (5.26) is equal to

$$m(e) = \frac{4ac + b^2}{4a} \tag{5.31}$$

We now state and prove the main theorem of this chapter.

Theorem 5.3.1. *Consider the class of general nonlinear systems described by (5.1) which satisfies Assumptions 1-2 and the control model (5.9). Let the weight law be given by*

$$\dot{\hat{W}} = -2\gamma_W\left[\gamma_0\left(\hat{W} - W_0\right) + e\sigma^T\right] \tag{5.32}$$

where $\gamma_t > 0$, $\dot{\hat{W}} = \dot{\tilde{W}}$, W_0 is a constant matrix and P is arbitrary, since $P = P^T > 0$, then the following is valid

$$L^T P^{-1} + P^{-1} L = Q \tag{5.33}$$

where $L > 0$ and $Q > 0$. So the errors e, $\tilde{W}$ are bounded and e is uniformly ultimately bounded with ultimate bound ρ_2, where $\rho_2 = \frac{\frac{b}{2} + \sqrt{\lambda_{min}(Q)c + (\frac{b}{2})^2}}{\lambda_{min}(Q)}$, $b = 2\bar{\varepsilon}_0 + 2\left\|P^{-1}\right\|_F \bar{h}_0$, and $c = \gamma_0\left\|W^* - W_0\right\|_F^2$.

Proof. Consider the Lyapunov function candidate

$$V = e^T P^{-1} e + \frac{tr\left(\tilde{W}^T \gamma_W^{-1} \tilde{W}\right)}{2} \tag{5.34}$$

By Deriving (5.34), we obtain

$$\dot{V} = \dot{e}^T P^{-1} e + e^T P^{-1} \dot{e} + \frac{\gamma_W^{-1} tr\left(\tilde{W}^T \dot{\tilde{W}} + \dot{\tilde{W}}^T \tilde{W}\right)}{2} \tag{5.35}$$

$$\dot{V} = e^T P^{-1} \dot{e} + \left(e^T P^{-1} \dot{e}\right)^T + \frac{\gamma_W^{-1} tr\left[\tilde{W}^T \dot{\tilde{W}} + \left(\tilde{W}^T \dot{\tilde{W}}\right)^T\right]}{2} \tag{5.36}$$

Using equation (5.16), this results

$$\dot{V} = e^T P^{-1} \dot{e} + (e^T P^{-1} \dot{e})^T + \gamma_W^{-1} tr\left(\tilde{W}^T \dot{\tilde{W}}\right) \tag{5.37}$$

Replacing equations (5.13) and (5.32)

$$\begin{aligned} \dot{V} &= e^T P^{-1}\left(-Le - \gamma_W \gamma_0 e + P\tilde{W}\sigma - P\varepsilon - h\right) \\ &+ \left[e^T P^{-1}\left(-Le - \gamma_W \gamma_0 e + P\tilde{W}\sigma - P\varepsilon - h\right)\right]^T \\ &+ \gamma_W^{-1} tr\left\{\tilde{W}^T\left[-2\gamma_W\left(\gamma_0\left(\hat{W} - W_0\right) + e\sigma^T\right)\right]\right\} \end{aligned} \tag{5.38}$$

$$\begin{aligned} \dot{V} &= -e^T\left(P^{-1}L + L^T P^{-1}\right)e - \gamma_W \gamma_0\left[e^T P^{-1} e + \left(e^T P^{-1} e\right)^T\right] \\ & e^T\left(\tilde{W}\sigma - \varepsilon - P^{-1}h\right) + \left(e^T\left(\tilde{W}\sigma - \varepsilon - P^{-1}h\right)\right)^T \\ &- 2\gamma_0 tr\left[\tilde{W}^T\left(\hat{W} - W_0\right)\right] - 2tr\left(\tilde{W}^T e\sigma^T\right) \end{aligned} \tag{5.39}$$

Since $e^T\left(\tilde{W}\sigma - \varepsilon - P^{-1}h\right)$ is a scalar number, so the transpose of this number is itself and employing facts 1 and 2 and equation (5.33), it results

$$\begin{aligned} \dot{V} &= -e^T Q e - 2\gamma_W \gamma_0 e^T P^{-1} e + 2e^T\left(\tilde{W}\sigma - \varepsilon - P^{-1}h\right) \\ &- \gamma_0\left(\left\|\tilde{W}\right\|_F^2 + \left\|\hat{W} - W_0\right\|_F^2 - \left\|W^* - W_0\right\|_F^2\right) \\ &- 2e^T\tilde{W}\sigma \end{aligned} \tag{5.40}$$

Turning to an inequality and taking into account (5.17)

$$\begin{aligned} \dot{V} &\leq -\lambda_{min}(Q)\left\|e\right\|^2 - 2\gamma_W \gamma_0 e^T P^{-1} e + 2\left\|e\right\|\left(\left\|\varepsilon\right\| + \left\|P^{-1}\right\|_F \left\|h\right\|\right) \\ &- \gamma_0\left(\left\|\tilde{W}\right\|_F^2 + \left\|\hat{W} - W_0\right\|_F^2 - \left\|W^* - W_0\right\|_F^2\right) \end{aligned} \tag{5.41}$$

Considering that $\|\varepsilon\| < \bar{\varepsilon}_0$, $\|h\| < \bar{h}_0$, and rearranging (5.41) implies

$$\begin{aligned} \dot{V} \leq & -\|e\|^2 \left[\lambda_{min}(Q)\right] + \|e\| \left(2\bar{\varepsilon}_0 + 2\left\|P^{-1}\right\|_F \bar{h}_0\right) + \gamma_0 \|W^* - W_0\|_F^2 \\ & -\gamma_0 \left\|\tilde{W}\right\|_F^2 - \gamma_0 \left\|\hat{W} - W_0\right\|_F^2 - 2\gamma_W \gamma_0 e^T P^{-1} e \end{aligned} \tag{5.42}$$

Considering that $a = \lambda_{min}(Q)$, $b = 2\bar{\varepsilon}_0 + 2\left\|P^{-1}\right\|_F \bar{h}_0$, $c = \gamma_0 \|W^* - W_0\|_F^2$,where $a \geq 0$, $b \geq 0$, and $c \geq 0$, then

$$\begin{aligned} \dot{V} \leq & -a\|e\|^2 + b\|e\| + c \\ & -\gamma_0 \left\|\tilde{W}\right\|_F^2 - \gamma_0 \left\|\hat{W} - W_0\right\|_F^2 - 2\gamma_W \gamma_0 e^T P^{-1} e \end{aligned} \tag{5.43}$$

Case 1. For analysis of the limitation of $\tilde{W}$, resuming (5.44) and disregarding some negative terms

$$\dot{V} \leq -a\|e\|^2 + b\|e\| + c - \gamma_0 \left\|\tilde{W}\right\|_F^2 \tag{5.44}$$

Using fact 5, we have

$$\dot{V} \leq \frac{4ac + b^2}{4a} - \gamma_0 \left\|\tilde{W}\right\|_F^2 \tag{5.45}$$

Hence, $\dot{V} < 0$ as long as

$$\gamma_0 \left\|\tilde{W}\right\|_F^2 > \frac{4ac + b^2}{4a} \tag{5.46}$$

$$\left\|\tilde{W}\right\|_F > \pm\sqrt{\frac{4ac + b^2}{4a\gamma_0}} \tag{5.47}$$

As $-\sqrt{\frac{4ac+b^2}{4a\gamma_0}} < 0$ and $\left\|\tilde{W}\right\|_F \geq 0$, this is an invalid solution, so

$$\left\|\tilde{W}\right\|_F > \sqrt{\frac{4ac + b^2}{4a\gamma_0}} \tag{5.48}$$

$$\left\|\tilde{W}\right\|_F > \sqrt{\frac{\frac{ac}{\gamma_0} + \frac{1}{\gamma_0}\left(\frac{b}{2}\right)^2}{a}} \tag{5.49}$$

Replacing a,b, and c

$$\left\|\tilde{W}\right\|_F > \sqrt{\frac{\lambda_{min}(Q)\|W^* - W_0\|_F^2 + \frac{1}{\gamma_0}\left(\bar{\varepsilon}_0 + \|P^{-1}\|_F \bar{h}_0\right)^2}{\lambda_{min}(Q)}} = \rho_1 \tag{5.50}$$

Thus, since ρ_1 is a constant, by using Lyapunov arguments [34], we concluded that $\tilde{W}$ is uniformly ultimately bounded, with ultimate bound equal to ρ_1 . Note that if, by any reason,

$\left\|\tilde{W}\right\|_F$ escapes of the residual set Ω_1, where $\Omega_1 = \left\{\tilde{W} : \left\|\tilde{W}\right\|_F \leq \rho_1\right\}$, $\dot{V}$ becomes negative definite again, and it forces the convergence of the weight error to the residual set Ω_1.

Case 2. For analysis of the limitation of e, resuming (5.43) and disregarding some negative terms

$$\dot{V} \leq -a \left\|e\right\|^2 + b \left\|e\right\| + c \tag{5.51}$$

Hence, $\dot{V} < 0$ as long as

$$a \left\|e\right\|^2 - b \left\|e\right\| - c > 0 \tag{5.52}$$

Using fact 4, we have

$$\left\|e\right\| > \frac{\frac{b}{2} + \sqrt{ac + (\frac{b}{2})^2}}{a} \tag{5.53}$$

Replacing a,b, and c

$$\left\|e\right\| > \frac{\bar{\varepsilon}_0 + \left\|P^{-1}\right\|_F \bar{h}_0 + \sqrt{\lambda_{min}\left(Q\right) \gamma_0 \left\|W^* - W_0\right\|_F^2 + \left(\bar{\varepsilon}_0 + \left\|P^{-1}\right\|_F \bar{h}_0\right)^2}}{\lambda_{min}\left(Q\right)} = \rho_2 \tag{5.54}$$

Thus, since ρ_2 is a constant, by using Lyapunov arguments [34], we concluded that e is uniformly ultimately bounded, with ultimate bound equal to ρ_2 . Note that if, by any reason, $\left\|e\right\|$ escapes of the residual set Ω_2, where $\Omega_2 = \{e : \left\|e\right\| \leq \rho_2\}$, $\dot{V}$ becomes negative definite again, and it forces the convergence of the state error to the residual set Ω_2.

Case 3. For analysis of the transient time, resuming (5.43) and doing some manipulations

$$\begin{aligned} \dot{V} &\leq -\alpha V + \alpha V - a \left\|e\right\|^2 + b \left\|e\right\| + c \\ &- \gamma_0 \left\|\tilde{W}\right\|_F^2 - \gamma_0 \left\|\hat{W} - W_0\right\|_F^2 - 2\gamma_W \gamma_0 e^T P^{-1} e \end{aligned} \tag{5.55}$$

We analyze when $-a \left\|e\right\|^2 + b \left\|e\right\| + c < 0$. This consideration is necessary, since it is in this period that the transient is occurring

$$\begin{aligned} \dot{V} &\leq -\alpha V + \alpha \left(e^T P^{-1} e + \frac{\gamma_W^{-1}}{2} \left\|\tilde{W}\right\|_F^2 \right) \\ &- \gamma_0 \left\|\tilde{W}\right\|_F^2 - \gamma_0 \left\|\hat{W} - W_0\right\|_F^2 - 2\gamma_W \gamma_0 e^T P^{-1} e \end{aligned} \tag{5.56}$$

$$\begin{aligned} \dot{V} &\leq -\alpha V - \gamma_0 \left\|\hat{W} - W_0\right\|_F^2 \\ &+ \left\|\tilde{W}\right\|_F^2 \left(\frac{\alpha}{2\gamma_W} - \gamma_0\right) + e^T P^{-1} e \left(\alpha - 2\gamma_W \gamma_0\right) \end{aligned} \tag{5.57}$$

Considering that $\alpha = 2\gamma_W\gamma_0$, then

$$\dot{V} \leq -\alpha V - \gamma_0 \left\| \hat{W} - W_0 \right\|_F^2 \tag{5.58}$$

$$\dot{V} \leq -\alpha V \tag{5.59}$$

Using Lemma 3.2.4 [84]. It can be stated that:

$$V(t) \leq e^{-\alpha(t-t_0)}V(t_0), \qquad \forall t \geq t_0 \geq 0 \tag{5.60}$$

Assuming that $t_0 = 0$

$$V(t) \leq V(0)e^{-\alpha t} \tag{5.61}$$

$$ln\left[V(t)\right] \leq -\alpha t + ln\left[V(0)\right] \tag{5.62}$$

$$ln\left[\frac{V(t)}{V(0)}\right] \leq -\alpha t \tag{5.63}$$

$$ln\left[\frac{V(0)}{V(t)}\right] \geq \alpha t \tag{5.64}$$

$$t \leq \frac{ln\left[\frac{V(0)}{V(t)}\right]}{\alpha} \tag{5.65}$$

□

Remark 9. Note that the scaling of unknown nonlinearities has a positive impact on the performance of the identification. The scaling matrix P is introduced to attenuate the effect of approximation errors and disturbances, as can be seen in (5.54).

Remark 10. It is perceived from (5.54), that the size of the residual state error is inversely proportional to $\lambda_{min}(Q)$, where the eigenvalues of Q can be arbitrarily manipulated while changing the values of matrices L and P. Thus, it is possible from these arbitrary design matrices to control the residual state error size.

Remark 11. Note that the time t in (5.65) refers to the maximum transient regime duration in relation to the residual state error determined in (5.54). Thus, we can not say anything about the transient duration for a residual state error smaller than that.

Remark 12. It is possible to verify in (5.65) that the transient regime duration is inversely proportional to the value of α. Since α is related to γ_0 and γ_W, it is concluded that it is possible to increase or decrease the transient time by changing these design parameters.

Remark 13. Note that the choice of different values of γ_W does not imply a new calculation of γ_0 or the matrices P and L to maintain the desired allocation of the eigenvalues of Q. In addition, it is verified in (5.54) that γ_W does not influence the size of the residual state error norm. Thus, it can be stated that from γ_W it is possible to decouple the transient performance of the steady-state error.

Remark 14. In the equation (5.54) in the numerator there is the term $\left\|P^{-1}\right\|_F \bar{h}_0$, thus the external disturbances are present in that part. In this way, it can be stated that by changing the eigenvalues of the matrix Q, the size of the residual state error is also adjusted, even in the presence of limited disturbances.

Remark 15. In previous works, in spite of the parameters related to the residual state, we can not increase or reduce the transient duration independently of residual state error size. To allow this, it is necessary to have a parameter related to the transient duration that does change the residual state error. In this work this was possible because the controller model and the learning law were chosen in order to allow an independent adjust of the transient duration.

Remark 16. In previous works, for example in [99, 100], in spite of the residual state error is bounded, they can not increase or reduce the transient duration independently of residual state error size. To allow this, it is necessary to have a parameter related to the transient duration that does change the residual state error. In this work this was possible because the controller model and the learning law were chosen in order to allow an independent adjustment of the transient duration.

5.4 Simulation

This section presents an example to validate the theoretical results. In the simulation, Solver ode113(Adams) of Matlab/Simulink®, with variable-step and a relative tolerance of 1e-10 was used to obtain the numerical solutions. The identification of a chaotic three-dimensional welding system with chaotic behavior has been proposed. An analysis is made to estimate the size of the steady-error.

5.4.1 Welding System

Consider the chaotic welding system [109], which is described by

$$\begin{aligned}
\dot{x_1} &= x_3 - \left(\frac{c_1}{\pi r_e^2} x_2 + \frac{c_2 \rho}{\pi r_e^2} x_1 x_2^2\right) + u_1 \\
\dot{x_2} &= \frac{1}{L_s}\left(u_2 - (R_a + R_s + \rho x_1) x_2 - V_0 - E_a(l_c - x_1)\right) \\
\dot{x_3} &= \frac{1}{\tau_m}(k_m u_3 - x_3)
\end{aligned} \tag{5.66}$$

where x_1, x_2, and x_3 are state variables, u_1, u_2, and u_3 are inputs, and c_1, c_2, r_e, ρ, L_s, R_a, R_s, V_0, E_a, l_c, τ_m, and k_m are parameters, being chosen (according to [109]) as $c_1 = 3.3{\times}10^{-10}(m^3 s^{-1} A^{-1})$,

$c_2 = 0.78\times10^{-10}(m^3s^{-1}\Omega^{-1}A^{-2})$, $r_e = 0.6\times10^{-3}(m)$, $\rho = 0.43(\Omega m^{-1})$, $L_s = 306\times10^{-6}(H)$, $R_a = 0.0237(\Omega)$, $R_s = 6.8\times10^{-3}(\Omega)$, $V_0 = 15.5(V)$, $E_a = 400(Vm^{-1})$, $l_c = 0.025(m)$, $\tau_m = 50\times10^{-3}(s)$, and $k_m = 1(mV^{-1}s^{-1})$. Note that system (5.66) satisfies the Assumption 1, since the state variables evolve into compact sets.

The parameters and the state variables of a welding system are: c_1 is the melting rate constant 1; c_2 is melting rate constant 2; r_e is the electrode radius; ρ is the resistivity of the electrode; L_s is the total inductance; R_a is the arc resistance; R_s is the total wire resistance; V_0 is the constant charge zone; E_a is the arc length factor; l_c is the contact tip to work piece distance; τ_m is the motor time constant; k_m is the motor steady state gain;u_1 is the motor armature voltage associated to welding speed, u_2 is the open circuit voltage and u_3 is the motor armature voltage associated to wire feeder; x_1 is the stick out, x_2 is the welding current and x_3 is the welding wire speed.

To control the uncertain system (5.66), the proposed control model (5.9) and the adaptive law (5.32) were implemented. The initial conditions for the chaotic system and for the identification model were $x_1(0) = 0.01$, $x_2(0) = 0$, $x_3(0) = 0$, $\hat{x}_1(0) = 0$, $\hat{x}_2(0) = 0$, $\hat{x}_3(0) = 0$, and $\hat{W}(0) = 0$ in order to evaluate the performance of the proposed algorithm under adverse initial conditions. The design matrices were chosen as

$$L = 10\begin{bmatrix} 50 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 5 \end{bmatrix} \tag{5.67}$$

$$P = 20\begin{bmatrix} 50 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 5 \end{bmatrix} \tag{5.68}$$

The nonlinear vector function σ is equal to

$$\sigma = \begin{bmatrix} s(x_1) \\ s(x_2) \\ s(x_3) \\ s(x_1)s(x_2) \\ s(x_1)s(x_2)^2 \end{bmatrix} \tag{5.69}$$

The sigmoidal function used is a logistic function and is equal to $s(.) = \frac{10}{1+e^{-0.5(.)}}$. The design parameters γ_0 and γ_W were chosen as $\gamma_0 = 0.005$ and $\gamma_W = 0.005$. W_0 was chosen as

$$W_0^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{5.70}$$

Figures 5.2, 5.3, and 5.4 show the reference lines, where x_{nref} is the reference unfiltered and x_{ref} is the reference filtered. The filter used is of second order and is shown below:

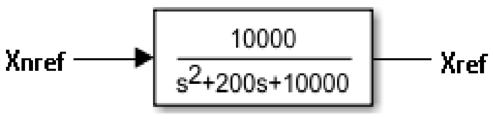


Figure 5.1: Filter

Figures 5.5 - 5.7 show the performances obtained in the estimation of the three states, figures 5.8 - 5.10 show the performances obtained in the estimation of the three states in monolog scale and figures 5.11 - 5.14 shows the control inputs.

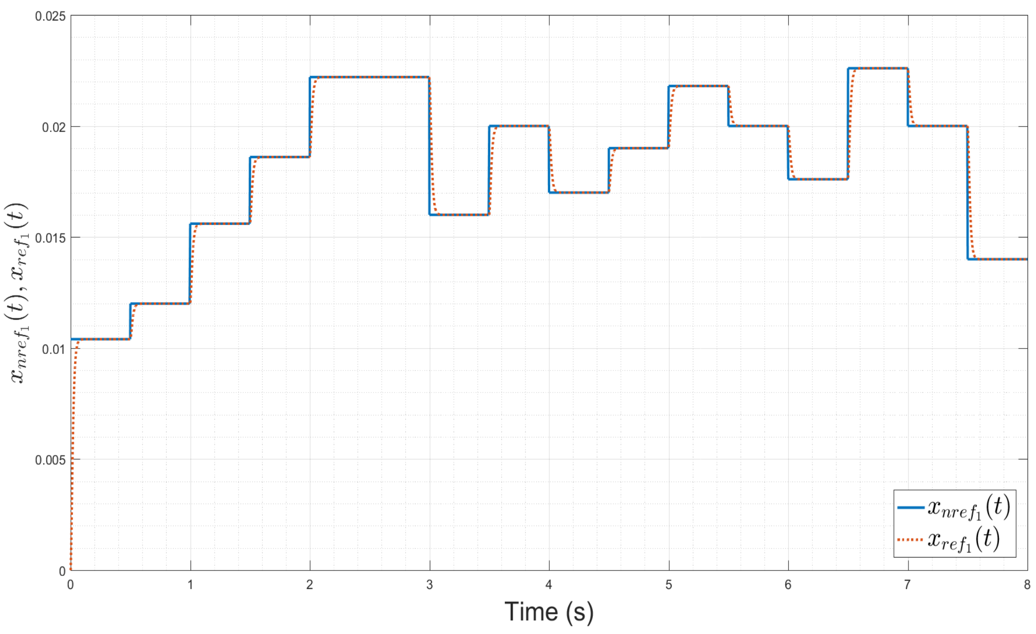


Figure 5.2: Filtered Reference 1

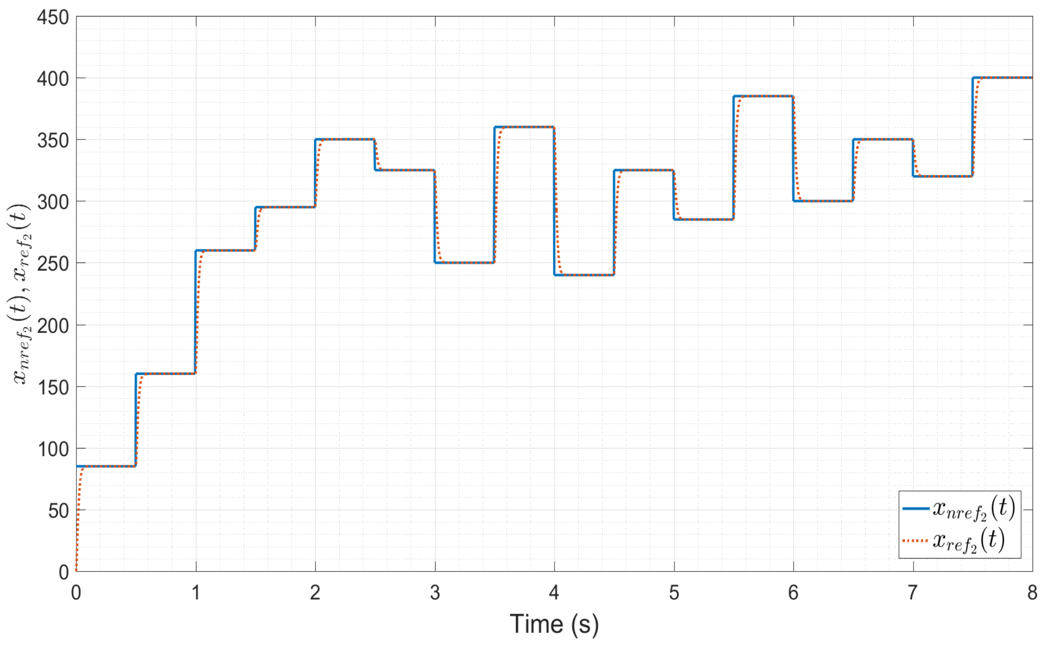


Figure 5.3: Filtered Reference 2

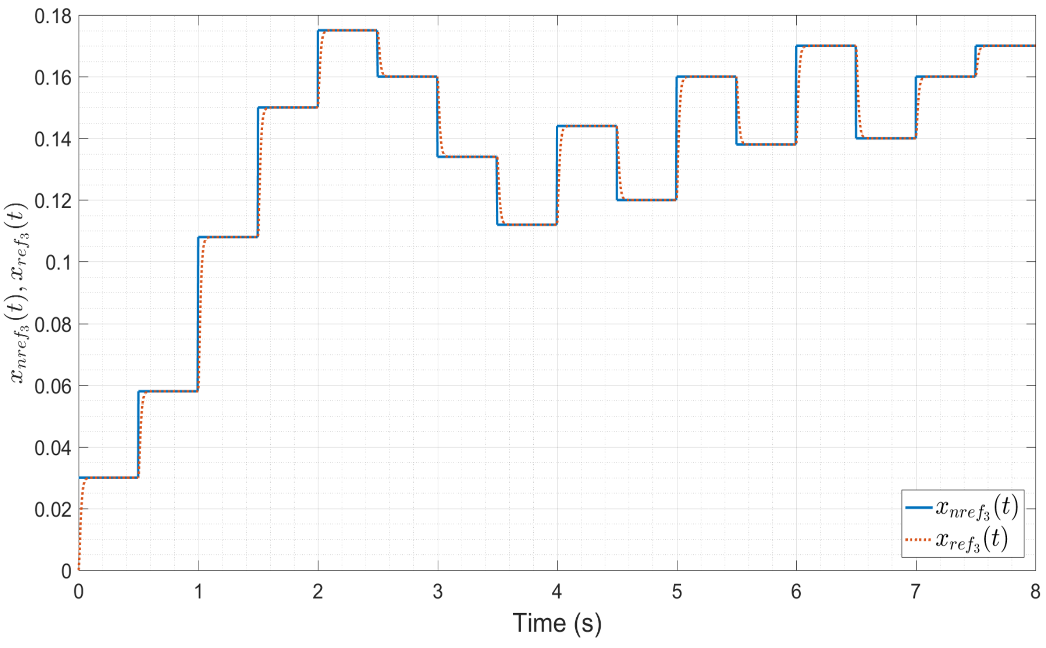


Figure 5.4: Filtered Reference 3

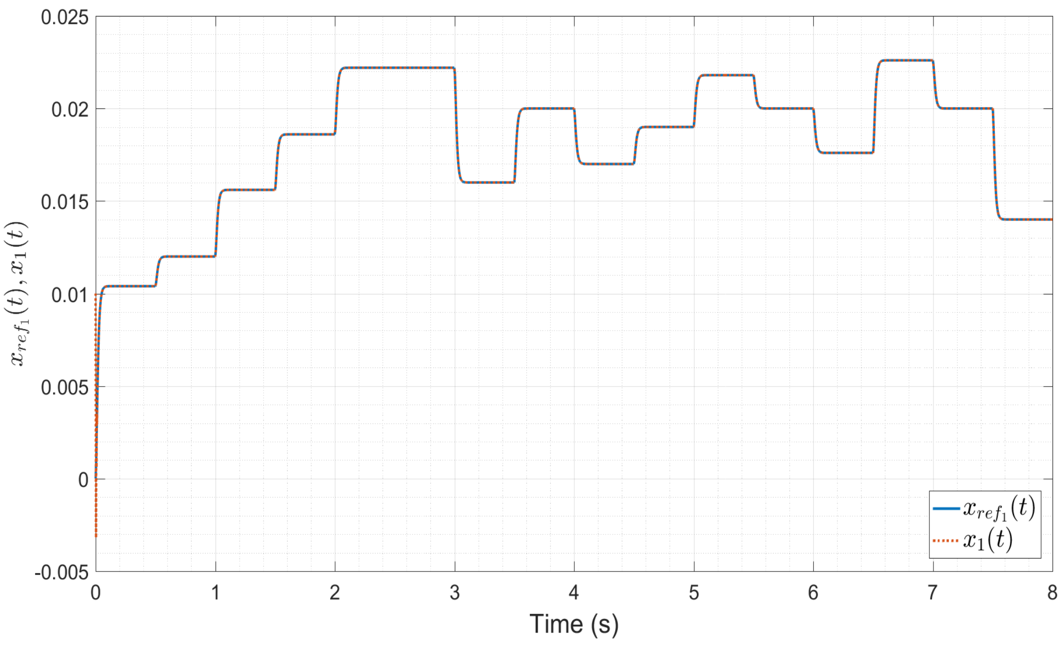


Figure 5.5: Performance in the estimation of x_1

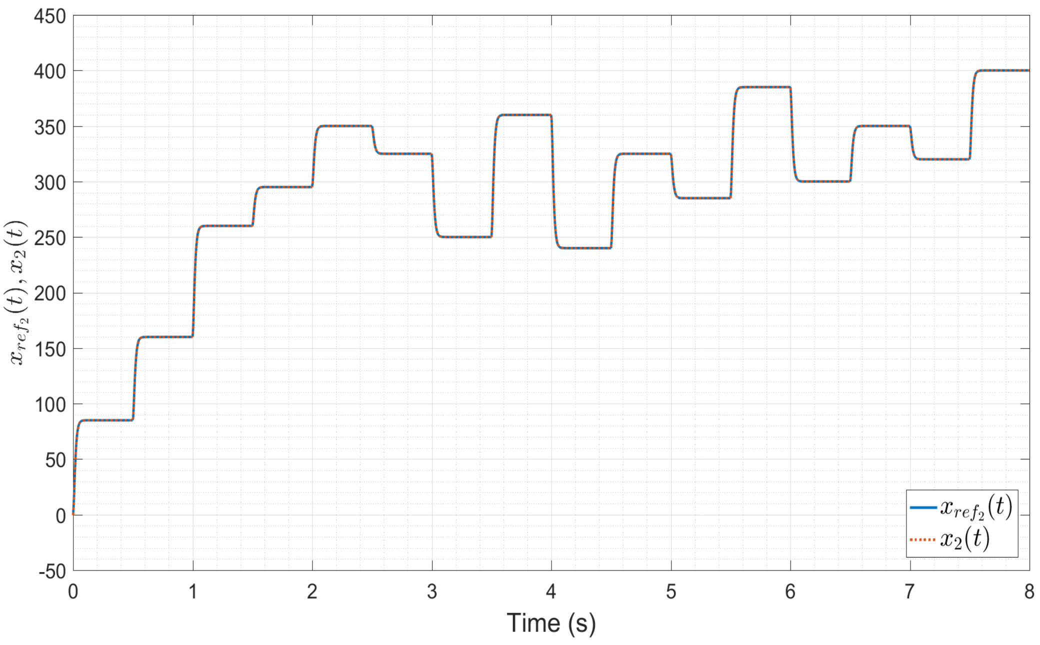


Figure 5.6: Performance in the estimation of x_2

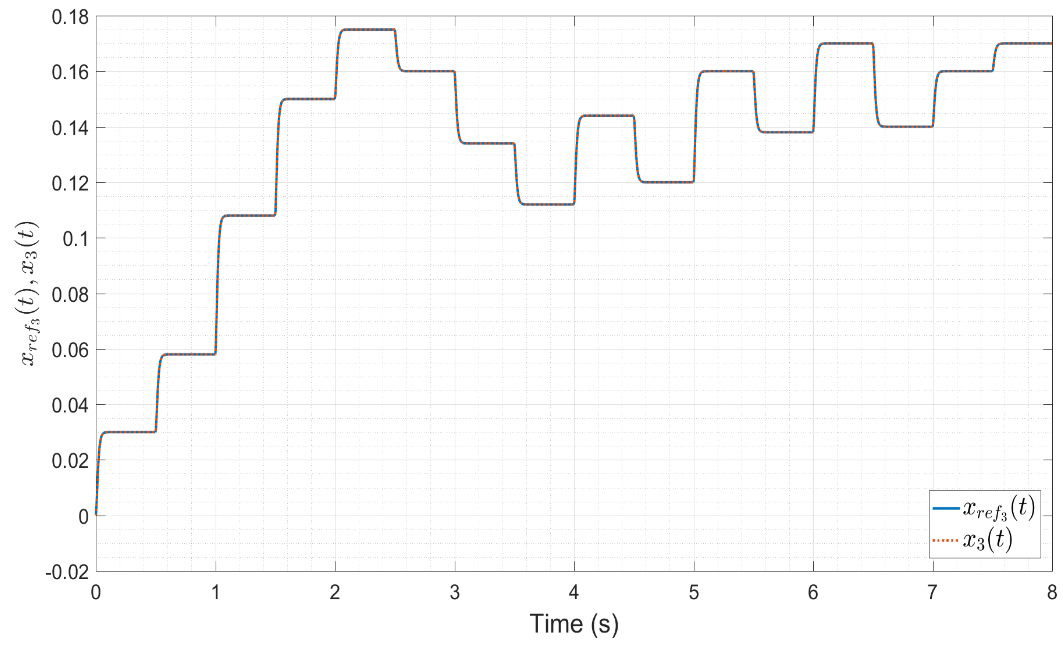


Figure 5.7: Performance in the estimation of x_3

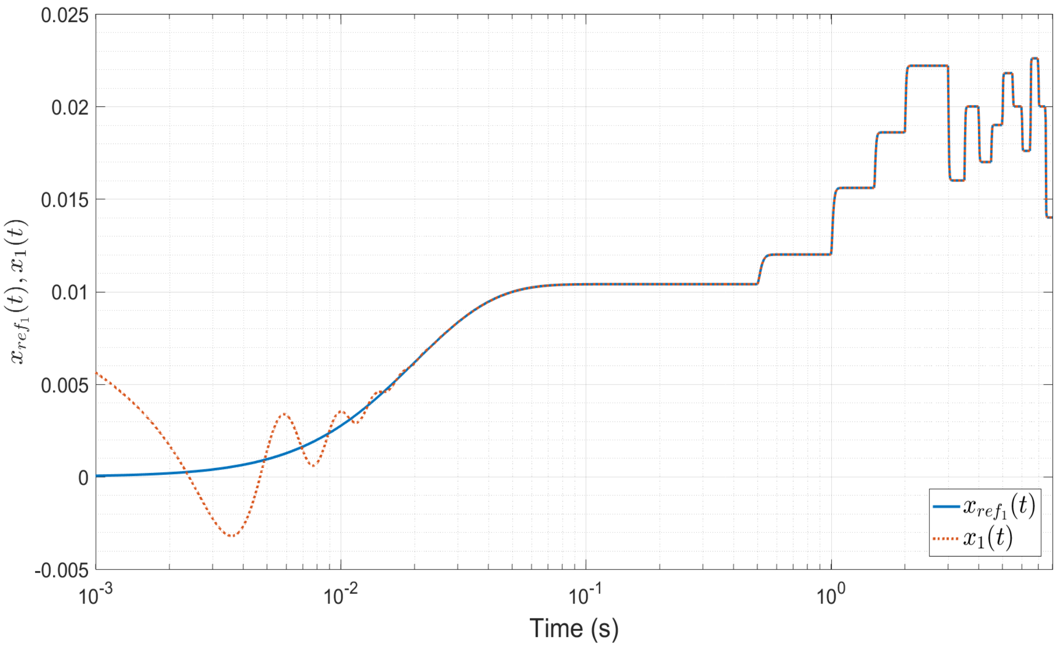


Figure 5.8: Performance in the estimation of x_1 - Monolog graph

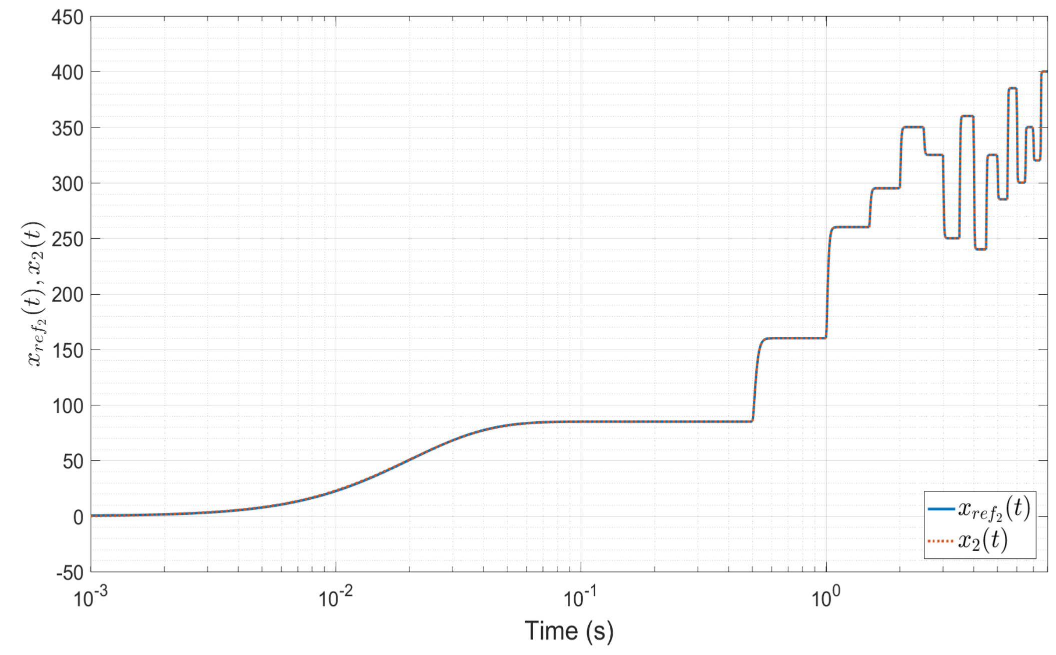


Figure 5.9: Performance in the estimation of x_2 - Monolog graph

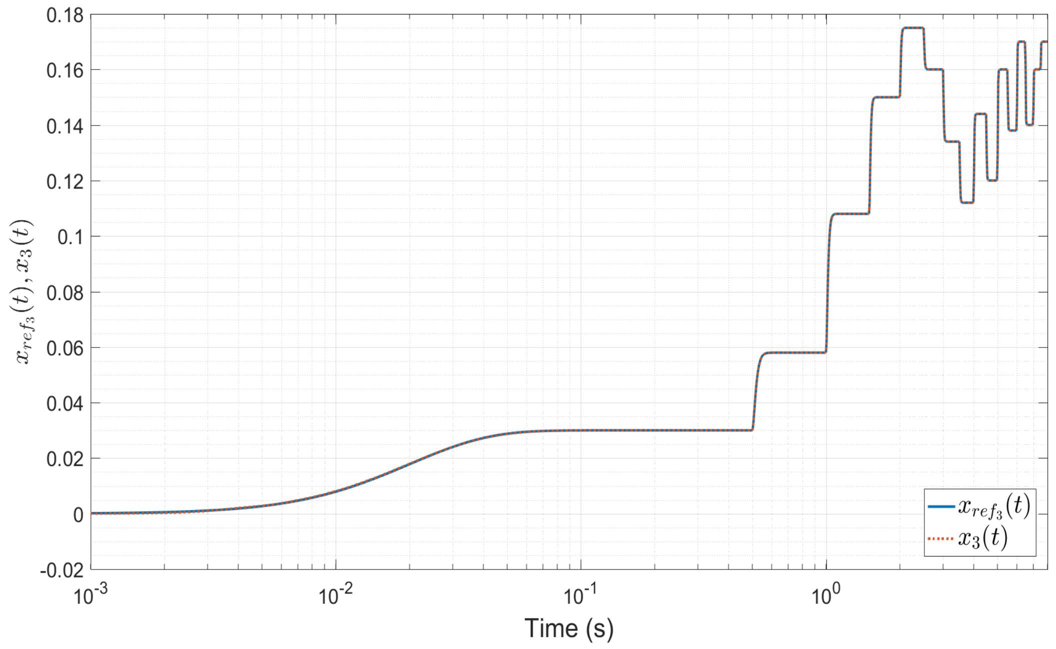


Figure 5.10: Performance in the estimation of x_3 - Monolog graph

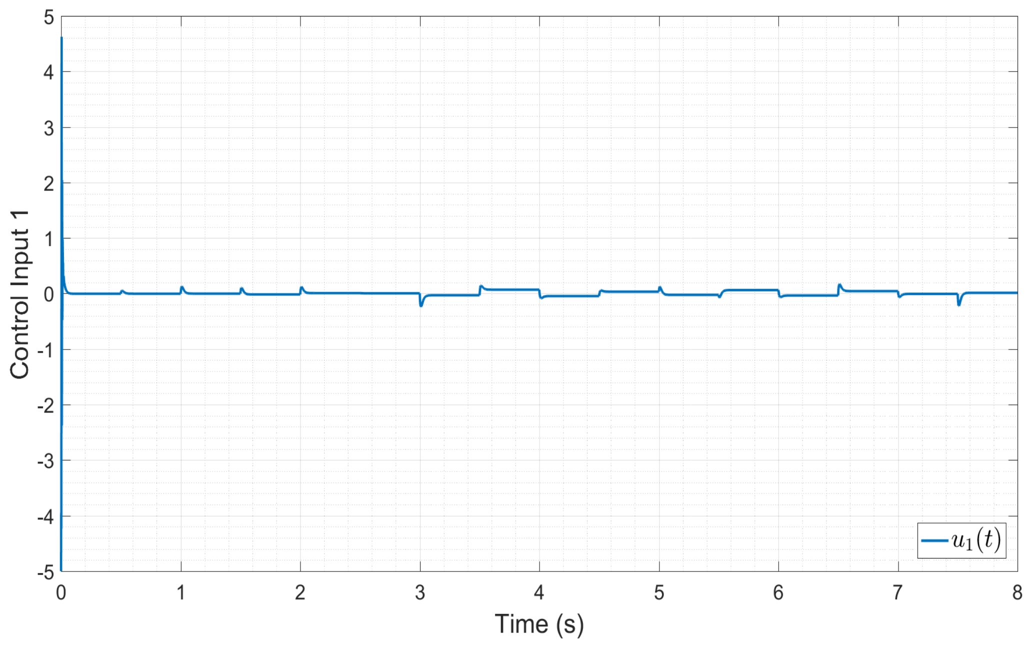


Figure 5.11: Control Input 1

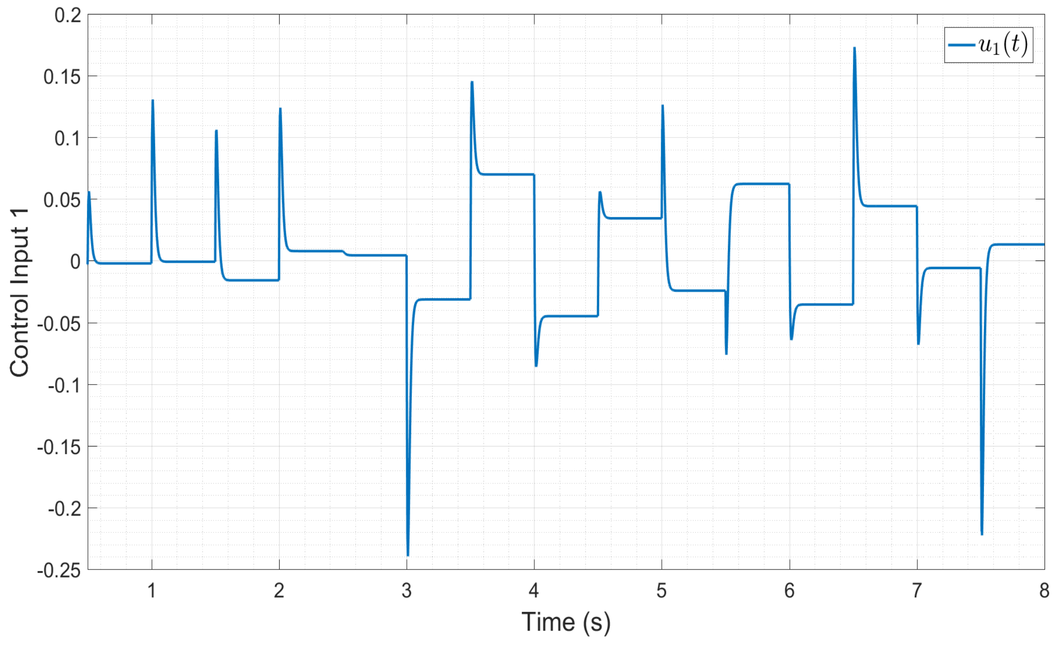


Figure 5.12: Control Input 1 - time varying between [0.5,8]

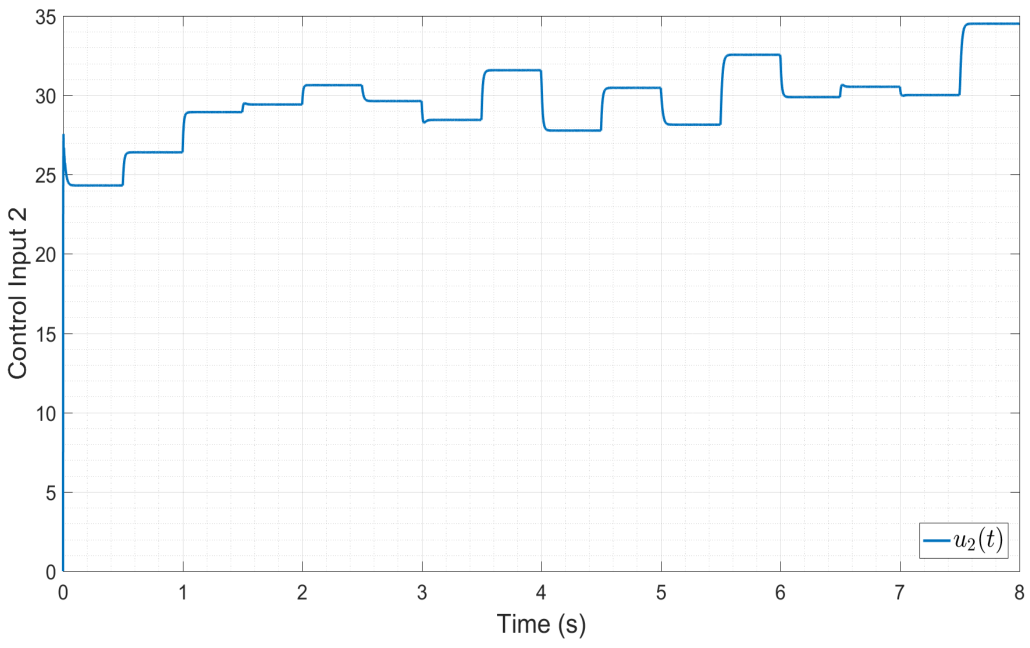


Figure 5.13: Control Input 2

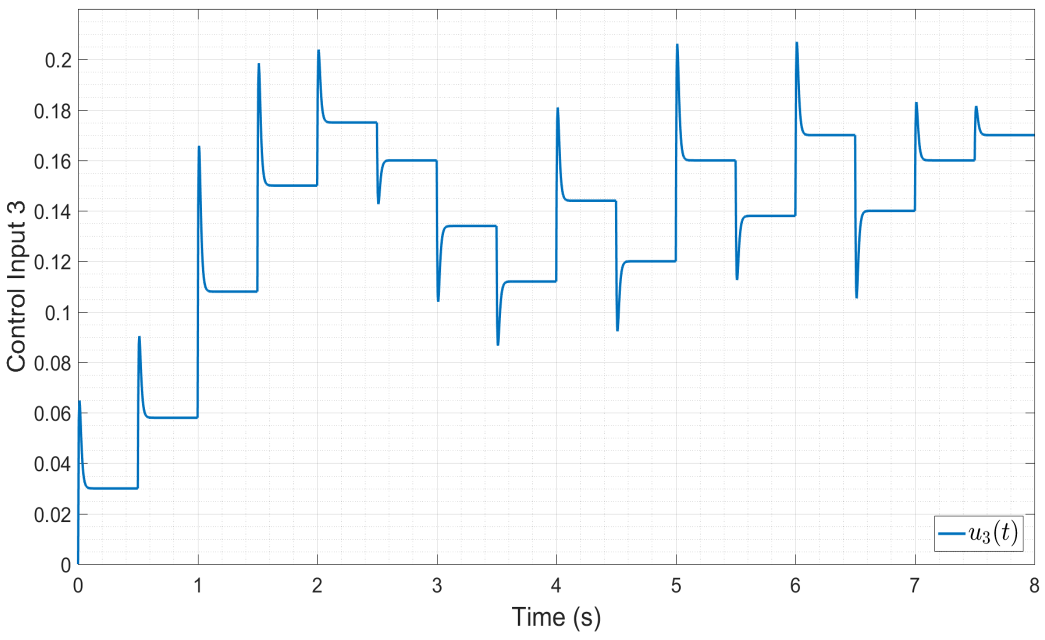


Figure 5.14: Control Input 3

Figures 5.5 - 5.10 show that the state converges to reference signal and the algorithm is stable. Figures 5.11 - 5.14 show that the control inputs seem to converge to constant values. The result is as expected, since L and P are chosen to reduce the residual error to desired one. It would be possible to have changed these values if a minor residual error is required.

Remark 17. The control of systems is important for welding since if there is a change of variable, the behavior of the model can change drastically and the existence of algorithms that allow to identify models of welding are demanded.

5.5 Summary

In this chapter, by using neural networks and Lyapunov methods, a scheme was proposed to control uncertain nonlinear systems. The proposed scheme is based on explicit feedback to ensure the convergence of the residual state error to a set defined from design parameters. The proposed scheme allows the states to converge to reference signals. It was verified in the simulations that changes in some design parameters can be done to make the residual state error converges to a neighborhood of the origin. An application of the controller was done in a welding system with chaotic behavior. It is observed that the simulations confirm the theoretical results.

Chapter 6

Conclusions

In this work, identification, observation, and control schemes of uncertain nonlinear systems based on neural networks and Lyapunov theory have been studied. Initially, all issues related to identification, observation, and adaptive control based on neural networks and Lyapunov methods that are relevant to our work have been considered.

In the sequence, three schemes about the above-mentioned issues based on Lyapunov arguments have been proposed in order to relate the state, observation, and tracking errors to independent design parameters with the aim to decouple the transient and steady-error performances. Although there are several works in literature which consider the identification and control based on neural networks, it is noteworthy that the decoupling of the transient and steady-error performances in these problems has rarely been investigated. In particular, secure communication based on analog chaos and control of welding systems are two topics which have motivated enormous technological and scientific interest in the last years. Hence, in this work, chaotic and welding systems have been employed to validate our approaches.

On the other hand, the presence of disturbances in all simulation has also been considered to evaluate the robustness of the proposed algorithms. Several classes of disturbances have been used to show that the proposed schemes corroborate the theoretical results, which are, that the algorithms are stable and the residual errors converge to an arbitrary neighborhood of the origin, where the transient and steady-error performances can be adjusted in an irrespective way.

Exhaustive simulations were carried out in order to evaluate the influence of the design parameters in the performance of the algorithms. The independence of the transient and steady-error performances has been fully confirmed. In particular, the identification and control of a welding system have been accomplished, which showed that the proposed schemes can be used successfully in this case, where the transient and steady-state can be adjusted according to any desired geometric parameters of the weld bead.

For future work, the following research lines are suggested:

- It is well-known that linearly parameterized neural networks, as the considered in this work, suffer from the "curse of dimensionality". Therefore, a natural sequence to alleviate the aforementioned drawback lies in the use of nonlinearly parameterized neural networks. In this sense, the identifier in [116] can be used as a starting point.

- The choice of the design matrices in the adaptive observer in Chapter 4 is not trivial since both the detectability and geometric conditions must be satisfied. Hence, another interesting research line lies in the development of numerical procedures based on linear matrix inequalities (LMI) [117] to alleviate the selection of the design matrices. Hence, more complex cases, such as the observation of hyperchaotic systems [118] should be easily addressed.
- The control of underactuated nonlinear systems is another way of extension. In this case, nonlinear techniques, such as integrator backstepping [85], can be applied.
- Last, but not least, there is much room in the control area with restrictions based on neural networks. For instance, the control with prescribed performance and saturation is a topic which deserves investigation [100, 119].

References

[1] POZNYAK, A. S.; SANCHEZ, E. N.; YU, W. *Differential Neural Networks for Robust Nonlinear Control - Identification, State Estimation and Trajectory Tracking.* 1. ed. [S.l.]: World Scientific, 2001.

[2] HAYKIN, S. *Neural networks and learning machines.* 3. ed. [S.l.]: Pearson Prentice Hall, 2008.

[3] ISERMANN, R.; MÜNCHHOF, M. *Identification of Dynamic Systems - An Introduction with Applications.* 1. ed. [S.l.]: Springer, Verlag Berlin Heidelberg, 2011.

[4] LJUNG, L. *System Identification - Theory for the User.* 2. ed. [S.l.]: Prentice-Hall International, New Jersey, U.S.A, 1999.

[5] SJÖBERG, J. et al. Nonlinear black-box modeling in system identification: A unified overview. *Automatica*, p. 1691–1724, 1996.

[6] DHALIWAL, S. S. *State Estimation and Parameter Identification of Continuous-time Nonlinear Systems.* Thesis (Master) — Queen's University, 2011.

[7] LINDSKOG, P.; LJUNG, L. Tools for Semi-Physical Modeling. *IFAC Proceedings Volumes*, v. 27, p. 1199–1204, 1994.

[8] NELLES, O. *Nonlinear System Identification - From Classical Approaches to Neural Networks and Fuzzy Models.* 1. ed. [S.l.]: Springer, Verlag Berlin Heidelberg, 2001.

[9] SONTAG, E. D. Some new directions in control theory inspired by systems biology. *IEEE Proceedings - Systems Biology*, v. 1, n. 1, p. 9–18, 2004.

[10] BERNHARDT, M. *Advances in System Identification, Neuromuscular Modeling and Repetitive Peripheral Magnetic Stimulation.* Thesis (Master) — Technische UniversitÄt MÜnchen, 2009.

[11] ZADEH, L. On the Identification Problem. *IRE Transactions on Circuit Theory, IEEE*, v. 3, p. 277–281, 1956.

[12] RUSSELL, S. J.; NORVIG, P. *Artificial Intelligence: A Modern Approach.* 1. ed. [S.l.]: Prentice-Hall, 1995.

[13] MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, v. 5, p. 115–133, 2004.

[14] JAIN, A. K.; MAO, J.; MOHIUDDIN, K. M. Artificial neural networks: a tutorial. *IEEE Computational Science and Engineering*, v. 29, n. 3, p. 31–44, 1996.

[15] JAHANGIRI, F. Identification of twing-tanks dynamics using adaptive wavelet differential neural networks. *International Joint conference on Neural Networks*, p. 1–5, 2010.

[16] XUAN, H. *Nonlinear System Identification and Control using Dynamic Multi-Time Scales Neural Networks.* Thesis (Master) — Concordia University, 2010.

[17] NAGESH, S. B. *Adaptive fuzzy observer and robust controller for a 2-DOF robot arm.* Thesis (Master) — Delft University of Technology, 2011.

[18] NARENDRA, K.; PARTHASARATHY, K. Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, v. 1, n. 1, p. 4–27, 1990.

[19] GE, S. et al. *Stable Adaptive Neural Network Control.* 1. ed. [S.l.]: Springer, Kluwer academic publishers, 2002.

[20] KOSMATOPOULOS, E. et al. High-order neural network structures for identification of dynamical systems. *IEEE Transactions on Neural Networks*, v. 6, n. 2, p. 422–431, 1995.

[21] ALANIS, A. et al. Discrete-time recurrent high order neural networks for nonlinear identification. *Journal of the Franklin Institute*, v. 347, n. 7, p. 1253–1265, 2010.

[22] HAN, X. et al. Nonlinear systems identification using dynamic multi-time scale neural networks. *Neurocomputing*, v. 74, n. 17, p. 3428–3439, 2011.

[23] RUBIO, J.; YU, W. Stability Analysis of Nonlinear System Identification via Delayed Neural Networks. *IEEE Transactions on Circuits and Systems II: Express Briefs*, v. 54, n. 2, p. 161–165, 2007.

[24] SELMIC, R. R.; LEWIS, F. L. Multimodel neural networks identification and failure detection of nonlinear systems. *Proceedings of the 40th IEEE Conference on Decision and Control*, v. 4, p. 3128–3133, 2001.

[25] ROVITHAKIS, G. A.; CHRISTODOULOU, M. A. Adaptive control of unknown plants using dynamical neural networks. *IEEE Transactions on Systems, Man, and Cybernetics*, v. 24, n. 3, p. 400–412, 1994.

[26] BHASIN, S.; JOHNSON, M.; DIXON, W. E. A model-free robust policy iteration algorithm for optimal control of nonlinear systems. *49th IEEE Conference on Decision and Control (CDC)*, p. 3060–3065, 2010.

[27] YILMAZ, S.; OYSAL, Y. A Fuzzy Wavelet Neural Network Model for System Identification. *Ninth International Conference on Intelligent Systems Design and Applications*, p. 1284–1289, 2009.

[28] BANAKAR, A.; AZEEM, M. F. Identification and Prediction of Nonlinear Dynamical Plants Using TSK and Wavelet Neuro-Fuzzy Models. *3rd International IEEE Conference Intelligent Systems*, p. 617–620, 2006.

[29] LIU, G. P. *Nonlinear Identification and Control: A Neural Network Approach*. 1. ed. [S.l.]: Springer, 2001.

[30] SANNER, R. M.; SLOTINE, J. J. E. Gaussian networks for direct adaptive control. *IEEE Transactions on Neural Networks*, v. 3, n. 6, p. 837–863, 1992.

[31] GRZEIDAK, E. *Identification of Nonlinear Systems Based on Extreme Learning Machine and Multilayer Neural Networks*. Thesis (Master) — Universidade de Brasília, 2016.

[32] KUSCHEWSKI, J. G.; HUI, S.; ZAK, S. H. Application of feedforward neural networks to dynamical system identification and control. *IEEE Transactions on Control Systems Technology*, v. 1, n. 1, p. 37–49, 1993.

[33] TUTUNJI, T. A. Parametric system identification using neural networks. *Applied Soft Computing*, v. 47, p. 251–261, 2016.

[34] KHALIL, H. K. *Nonlinear systems*. 3. ed. [S.l.]: Prentice Hall - New York, 2002.

[35] AHMADI, A. A. *Algebraic Relaxations and Hardness Results in Polynomial Optimization and Lyapunov Analysis*. Thesis (Phd) — MIT, 2011.

[36] LYAPUNOV, A. M. *General problem of the stability of motion*. Thesis (Phd) — Kharkov Mathematical Society, 1892.

[37] HONG-YAN, W. et al. The deformation time series prediction based on wavelet and neural network. *International Conference on Electric Information and Control Engineering (ICEICE)*, p. 6242–6245, 2011.

[38] ABIYEV, R. H.; KAYNAK, O. Fuzzy Wavelet Neural Networks for Identification and Control of Dynamic Plants—A Novel Structure and Comparative Study. *IEEE Transactions on Industrial Electronics*, v. 55, n. 8, p. 3133–3140, 2008.

[39] WILLIAMS, R.; ZIPSER, D. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation*, v. 1, n. 2, p. 270–280, 1989.

[40] NARENDRA, K.; PARTHASARATHY, K. Gradient methods for the optimization of dynamical systems containing neural networks. *IEEE Transactions on Neural Networks*, v. 2, n. 2, p. 252–262, 1991.

[41] CANELON, J. I.; SHIEH, L. S.; KARAYIANNIS, N. B. A new approach for neural control of nonlinear discrete dynamic systems. *Information Sciences*, v. 174, n. 3-4, p. 177–196, 2005.

[42] HORNIK, K.; STINCHCOMBE, M.; WHITE, H. Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, v. 2, n. 5, p. 359–366, 1989.

[43] HORNIK, K. Approximation Capabilities of Multilayer Feedforward Networks. *Neural Networks*, v. 4, n. 2, p. 251–257, 1991.

[44] HAN, X. *Nonlinear System Identification and Control using Dynamic Multi-Time Scales Neural Networks*. Thesis (Master) — Concordia University, 2010.

[45] GORJI, A. A.; MENHAJ, M. B. Identification of nonlinear state space models using an MLP network trained by the EM algorithm. *IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, p. 53–60, 2008.

[46] ABDOLLAHI, F.; TALEBI, H. A.; PATEL, R. V. Stable identification of nonlinear systems using neural networks: theory and experiments. *IEEE/ASME Transactions on Mechatronics*, v. 11, n. 4, p. 488–495, 2006.

[47] BAZAEI, A.; MOALLEM, M. Online neural identification of multi-input multi-output systems. *IET Control Theory Applications*, v. 1, n. 1, p. 44–50, 2007.

[48] CHAIREZ, I.; POZNYAK, A.; POZNYAK, T. Stable weights dynamics for a class of differential neural network observer. *IET Control Theory Applications*, v. 3, n. 10, p. 1437–1447, 2009.

[49] ZHAO, X.; LI, Z.; LI, S. Synchronization of a chaotic finance system. *Applied Mathematics and Computation*, v. 217, n. 13, p. 6031–6039, 2011.

[50] VARGAS, J.; GULARTE, K.; HEMERLY, E. On-Line Neuro Identification of Uncertain Systems Based on Scaling and Explicit Feedback. *Control Automation and Electrical Systems*, v. 24, n. 6, p. 753–763, 2013.

[51] VARGAS, J. A. R.; GULARTE, K. H. M.; HEMERLY, E. M. An improved on-line neuro-identification scheme. *UKACC International Conference on Control*, p. 1088–1093, 2012.

[52] MALEK, A. *Design and Implementation of Piecewise-Affine Observers for Nonlinear Systems*. Thesis (Master) — Concordia University, 2013.

[53] ADHYARU, D. M. State observer design for nonlinear systems using neural network. *Applied Soft Computing*, v. 12, n. 8, p. 2530–2537, 2012.

[54] LUENBERGER, D. G. Observing the State of a Linear System. *IEEE Transactions on Military Electronics*, v. 8, n. 2, p. 74–80, 1964.

[55] LUENBERGER, D. G. Observers for multivariable systems. *IEEE Transactions on Automatic Control*, v. 11, n. 2, p. 190–197, 1966.

[56] LIEN, C.-H. Robust observer-based control of systems with state perturbations via LMI approach. *IEEE Transactions on Automatic Control*, v. 49, n. 8, p. 1365–1370, 2004.

[57] JIANG, Y.; JI, Q.; BAI, X. An improved neural network observer designed for nonlinear system. *ACSS*, v. 1, n. 6, p. 376–383, 2014.

[58] LI, B. A Non-Gaussian Kalman Filter With Application to the Estimation of Vehicular Speed. *Technometrics*, v. 51, n. 2, p. 162–172, 2009.

[59] LJUNG, L. Asymptotic behavior of the extended Kalman filter as a parameter estimator for linear systems. *IEEE Transactions on Automatic Control*, v. 24, n. 1, p. 36–50, 1979.

[60] HAYKIN, S. *Kalman Filtering and Neural Networks*. [S.l.]: Wiley-Interscience, 2001.

[61] WALCOTT, B.; CORLESS, M.; ZAK, S. Comparative study of nonlinear state observation techniques. *International Journal of Control*, v. 45, n. 6, p. 2109–2132, 1987.

[62] WU, Q. H.; JIANG, L.; WEN, J. Y. Decentralized Adaptive Control of Interconnected Non-Linear Systems Using High Gain Observer. *International Journal of Control*, v. 77, n. 8, p. 703–712, 2004.

[63] MEZOUAR, A.; FELLAH, M. K.; HADJERI, S. Adaptive sliding mode observer for induction motor using two-time-scale approach. *Electric Power Systems Researchs*, v. 77, n. 5-6, p. 604–618, 2007.

[64] BOKER, A. M. *Estimation and Control of Nonlinear Systems Using Extended High-Gain Observeres*. Thesis (Master) — Michigan State University, 2013.

[65] CELLE, F. et al. Synthesis of nonlinear observers: A harmonic-analysis approach. *Mathematical systems theory*, v. 22, n. 1, p. 291–322, 1989.

[66] LEU, Y.-G.; WANG, W.-Y.; LEE, T.-T. Observer-based direct adaptive fuzzy-neural control for nonaffine nonlinear systems. *IEEE Transactions on Neural Networks*, v. 16, n. 4, p. 853–863, 2005.

[67] VARGAS, J. A. R.; HEMERLY, E. M. Robust neural adaptive observer for MIMO nonlinear systems. *Systems, Man, and Cybernetics. IEEE SMC Conference Proceedings*, v. 4, p. 1084–1089, 1999.

[68] VARGAS, J. A. R.; HEMERLY, E. M. Neural adaptive observer for general nonlinear systems. *IEEE, American Control Conference*, v. 1, n. 6, p. 708–712, 2000.

[69] TALEBI F. ABDOLLAHI, R. V. P. K. K. H. A. *Neural Network-Based State Estimation of Nonlinear Systems*. 1. ed. [S.l.]: Springer Science Business Media, LLC, London, 2010.

[70] FU, Z.-J.; XIE, W.-F.; NA, J. Robust adaptive nonlinear observer design via multi-time scales neural network. *Neurocomputing*, v. 190, p. 217–225, 2016.

[71] HUANG, S.; TAN, K. K. Fault Detection and Diagnosis Based on Modeling and Estimation Methods. *IEEE Transactions on Neural Networks*, v. 20, n. 5, p. 872–881, 2009.

[72] QIU, J. et al. Fault Estimation for Nonlinear Dynamic Systems. *Springer-Link. Circuits, Systems, and Signal Processing*, v. 31, n. 2, p. 555–564, 2012.

[73] CAO, C.; HOVAKIMYAN, N. Novel L1 Neural Network Adaptive Control Architecture With Guaranteed Transient Performance. *IEEE Transactions on Neural Networks*, v. 18, n. 4, p. 1160–1171, 2007.

[74] MONTANARI, M. et al. Speed Sensorless Control of Induction Motors Based on a Reduced-Order Adaptive Observer. *IEEE Transactions on Control Systems Technology*, v. 15, n. 6, p. 1049–1064, 2007.

[75] TONG, S. C.; LI, Y. M.; ZHANG, H. G. Adaptive Neural Network Decentralized Backstepping Output-Feedback Control for Nonlinear Large-Scale Systems With Time Delays. *IEEE Transactions on Neural Networks*, v. 22, n. 7, p. 1073–1086, 2011.

[76] DIMASSI, H.; LORIA, A. Adaptive Unknown-Input Observers-Based Synchronization of Chaotic Systems for Telecommunication. *IEEE Transactions on Circuits and Systems I: Regular Papers*, v. 58, n. 4, p. 800–812, 2011.

[77] LIANG, X.; ZHANG, J.; XIA, X. Adaptive Synchronization for Generalized Lorenz Systems. *IEEE Transactions on Automatic Control*, v. 53, n. 7, p. 1740–1746, 2008.

[78] LIU, Y.; TANG, W. K. S. Cryptanalysis of Chaotic Masking Secure Communication Systems Using an Adaptive Observer. *IEEE Transactions on Circuits and Systems II: Express Briefs*, v. 55, n. 11, p. 1183–1187, 2008.

[79] ABDOLLAHI, F.; TALEBI, H. A.; PATEL, R. V. A stable neural network-based observer with application to flexible-joint manipulators. *IEEE Transactions on Neural Networks*, v. 17, n. 1, p. 118–129, 2006.

[80] PAESA, D. et al. Adaptive Observers Applied to Pan Temperature Control of Induction Hobs. *IEEE Transactions on Industry Applications*, v. 45, n. 3, p. 1116–1125, 2009.

[81] POZNYAK, T. et al. Application of the differential neural network observer to the kinetic parameters identification of the anthracene degradation in contaminated model soil. *Journal of Hazardous Materials*, v. 146, n. 3, p. 661–667, 2007.

[82] RESENDIZ, J.; YU, W.; FRIDMAN, L. Two-Stage Neural Observer for Mechanical Systems. *IEEE Transactions on Circuits and Systems II: Express Briefs*, v. 55, n. 10, p. 1076–1080, 2008.

[83] VARGAS, J.; HEMERLY, E. Observação adaptativa neural com convergência assintótica na presença de parâmetros variantes no tempo e distúrbios. *(in Portuguese). Scielo. Sba Controle and Autom.*, v. 19, p. 18–29, 2008.

[84] IOANNOU, P.; SUN, J. *Robust adaptive control.* [S.l.]: Dover Publications, 1995.

[85] KRSTIC, M.; KANELLAKOPOULOS, I.; KOKOTOVIC, P. V. *Nonlinear and Adaptive Control Design.* [S.l.]: Wiley-Interscience, 1995.

[86] MARINO, R.; TOMEI, P. *Nonlinear Control Design, Nonlinear Control Design: Geometric.* [S.l.]: Prentice Hall International, 1995.

[87] CHAIREZ, I. Wavelet Differential Neural Network Observer. *IEEE Transactions on Neural Networks*, v. 20, n. 9, p. 1439–1449, 2009.

[88] CHOI, J. Y.; FARRELL, J. A. Adaptive observer backstepping control using neural networks. *IEEE Transactions on Neural Networks*, v. 12, n. 5, p. 1103–1112, 2001.

[89] STEPANYAN, V.; HOVAKIMYAN, N. Robust Adaptive Observer Design for Uncertain Systems With Bounded Disturbances. *IEEE Transactions on Neural Networks*, v. 18, n. 5, p. 1392–1403, 2007.

[90] GONZALEZ-OLVERA, M. A.; TANG, Y. Black-Box Identification of a Class of Nonlinear Systems by a Recurrent Neurofuzzy Network. *IEEE Transactions on Neural Network*, v. 21, n. 4, p. 672–679, 2010.

[91] KIM, J. H. et al. Adaptive Synchronization of Uncertain Chaotic Systems Based on T–S Fuzzy Model. *IEEE Transactions on Fuzzy Systems*, v. 15, n. 3, p. 359–369, 2007.

[92] VARGAS, J. A. R.; GULARTE, K. H. M.; HEMERLY, E. M. Adaptive observer design based on scaling and neural networks. *IEEE Latin America Transactions*, v. 11, n. 4, p. 989 – 994, 2013.

[93] JOVIC, B. *Synchronization Techniques for Chaotic Communication Systems*. [S.l.]: Springer, 2011.

[94] CUI, R.; ZHANG, X.; CUI, D. Adaptive sliding-mode attitude control for autonomous underwater vehicles with input nonlinearities. *Ocean Engineering*, v. 123, p. 45–54, 2016.

[95] YOO, S. J.; PARK, J. B.; CHOI, Y. H. Adaptive Dynamic Surface Control for Stabilization of Parametric Strict-Feedback Nonlinear Systems With Unknown Time Delays. *IEEE Transactions on Automatic Control*, v. 52, n. 12, p. 2360–2365, 2007.

[96] CHEN, X.; FENG, Y.; SU, C.-Y. Adaptive control for continuous-time systems with actuator and sensor hysteresis. *Automatica*, v. 64, p. 196–207, 2016.

[97] GE, W. C. S. H. S. S. Consensus-based distributed cooperative learning control for a group of discrete-time nonlinear multi-agent systems using neural networks. *Automatica*, v. 50, n. 9, p. 2254–2268, 2014.

[98] HE, W.; DONG, Y.; SUN, C. Adaptive neural impedance control of a robotic manipulator with input saturation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, v. 46, n. 3, p. 334–344, 2016.

[99] LIU, L.; GUO, R. Control problems of Chen–Lee system by adaptive control method. *Nonlinear Dynamics*, v. 87, n. 1, p. 503–510, 2017.

[100] LIU, Y.-J.; TONG, S. Barrier Lyapunov functions for Nussbaum gain adaptive control of full state constrained nonlinear systems. *Automatica*, v. 76, p. 143–152, 2017.

[101] CHU, W.-H.; TUNG, P.-C. Development of an automatic arc welding system using a sliding mode control. *International Journal of Machine Tools and Manufacture*, v. 45, n. 7-8, p. 933–939, 2005.

[102] GU, W. P.; XIONG, Z. Y.; WAN, W. An Automatic Control System Based on Arc Voltage of TIG Welding. *Trans Tech Publications. Advanced Materials Research*, v. 418, p. 1379–1382, 2012.

[103] BERA, M. K.; BANDYOPADHYAY, B.; PAUL, A. K. Robust nonlinear control of GMAW systems-a higher order sliding mode approach. *IEEE International Conference on Industrial Technology (ICIT)*, p. 175–180, 2013.

[104] KARAFI, M. R. et al. Study on automatic control of arc gap in robotic TIG welding. *The International Journal of Advanced Manufacturing Technology*, v. 50, n. 9-12, p. 953–960, 2010.

[105] LEE, C.-Y.; TUNG, P.-C.; CHU, W.-H. Adaptive fuzzy sliding mode control for an automatic arc welding system. *The International Journal of Advanced Manufacturing Technology*, v. 29, n. 5-6, p. 481–489, 2006.

[106] VARGAS, J. A. R.; GRZEIDAK, E.; HEMERLY, E. M. Robust adaptive synchronization of a hyperchaotic finance system. *Nonlinear Dynamics*, v. 80, n. 1-2, p. 239–248, 2015.

[107] AKGUL, A.; HUSSAIN, S.; PEHLIVAN, I. A new three-dimensional chaotic system, its dynamical analysis and electronic circuit applications. *Optik - International Journal for Light and Electron Optics*, v. 127, n. 18, p. 7062–7071, 2016.

[108] YU, H.; CAI, G.; LI, Y. Dynamic analysis and control of a new hyperchaotic finance system. *Nonlinear Dynamics*, v. 67, n. 3, p. 2171–2182, 2012.

[109] ANZEHAEE, M. M.; HAERI, M. Welding current and arc voltage control in a GMAW process using ARMarkov based MPC. *Control Engineering Practice*, v. 19, n. 12, p. 1408–1422, 2011.

[110] BOLLIG, A. et al. Identification and predictive control of laser beam welding using neural networks. *European Control Conference (ECC)*, p. 2457–2462, 2006.

[111] YE, X.; HU, L.; LIU, Y. Nonlinear identification and self-learning CMAC neural network based control system of laser welding process. *9th International Conference on Electronic Measurement Instruments*, p. 3–440–3–445, 2009.

[112] NA, X. et al. Nonlinear Identification of Laser Welding Process. *IEEE Transactions on Control Systems Technolog*, v. 18, n. 4, p. 927–934, 2010.

[113] BINGUL, Z. *Dynamic Modeling of the Gas Metal Arc Welding Process.* Thesis (Phd) — Elect. Eng. Dept., Vanderbilt Univ., Nashville, TN, 2000.

[114] LJUNG, L.; GUNNARSSON, S. Adaptation and tracking in system identification—A survey. *Automatica*, v. 26, n. 1, p. 7–21, 1990.

[115] ZHIYONG, L. et al. An Analysis of Gas Metal Arc Welding Using the Lyapunov Exponent. *Materials and Manufacturing Processes*, v. 28, n. 2, p. 213–219, 2013.

[116] VARGAS, J. A. R.; GRZEIDAK, E.; ALFARO, S. C. A. Identification of unknown nonlinear systems based on multilayer neural networks and Lyapunov theory. *Computational Intelligence (SSCI), IEEE Symposium Series on*, p. 1–7, 2016.

[117] BOYD, S. et al. *Identification of Dynamic Systems - An Introduction with Applications.* Philadelphia, PA: Linear Matrix Inequalities in System and Control Theory, 1994. (Studies in Applied Mathematics, v. 15).

[118] PLANCKAERT, J.-P. et al. Modeling of MIG/MAG welding with experimental validation using an active contour algorithm applied on high speed movies. *Applied Mathematical Modelling*, v. 34, n. 4, p. 1004–1020, 2010.

[119] TEE, K. P.; GE, S. S.; TAY, E. H. Barrier Lyapunov Functions for the control of output-constrained nonlinear systems. *Automatica*, v. 45, n. 4, p. 918–927, 2009.

[120] PETERSEN, K. B.; PEDERSEN, M. S. *The Matrix Cookbook.* [S.l.]: Technical University of Denmark, 2012.

[121] ZHANG, Q.; BENVENISTE, A. Wavelet networks. *IEEE Transactions on Neural Networks*, v. 3, n. 6, p. 889–898, 1992.

[122] WANG, L. *Adaptive Fuzzy Systems and Control: Design and Stability Analysis.* [S.l.]: Pearson Prentice Hall, 1994.

[123] HAN, J.; MORAG, C. The influence of the sigmoid function parameters on the speed of backpropagation learning. *Springer - Natural to Artificial Neural Computation*, p. 195–201, 1995.

[124] ANTUNES, R.; GONZÁLEZ, V. A.; WALSH, K. Identification of Repetitive Processes at Steady- and Unsteady-state: Transfer Function. *Research Gate Conference Paper Perth, Australia*, p. 793–802, 2015.

Appendix

I. TECHNICAL BACKGROUND

I.1 Motivation

In this chapter, technical background about Lyapunov Stability Theory, neural networks, their properties and the notation that will be used throughout this Master's thesis will be introduced. Our aim is to provide basic information about the contents used in this dissertation.

I.2 Mathematical Preliminaries

This section provides some fundamental mathematical concepts that are necessary for the remaining chapters.

I.2.1 Vector Norms

Definition 1. *Let $x \in X \subset \Re^n$ be a $n-dimensionalvector$. The p-norm of f is defined by*

$$\|x\|_p = \left(\sum_i |x_i|^p \right)^{1/p}, \; for \; p \in [1, \infty) \tag{I.1}$$

Thus, by denoting $p = 1, 2, \infty$, the corresponding normed spaces are called L_1, L_2, L_∞, respectively. In this dissertation we usually use the case where $p = 2$:

$$\|x\| = \|x\|_2 = \sqrt{\sum_i |x_i|^2} \tag{I.2}$$

$$\|x\|^2 = x^T x, \; x \in \Re^{1 \times n} \tag{I.3}$$

More information can be obtained in [19, 120]

I.2.2 Matrix Norms

Definition 2. *Let $A, B \in \Re^{n \times n}$ be $n - dimensional$ matrices. Then, the following properties are valid*

$$\|A\| > 0 \tag{I.4}$$

$$\|cA\| = \|c\| \|A\| \, c \in \Re \tag{I.5}$$

$$\|cA\| \ = \|c\|\|A\| \, c \in \Re \tag{I.6}$$

$$\|A+B\| \ \leq \|A\| \ + \|B\| \tag{I.7}$$

The Frobenius norm can be defined as:

$$\|A\|_F^2 \ = tr(A^T A) \tag{I.8}$$

Where $tr(A)$ is the trace function. This function returns the sum of diagonal entries of a square matrix. More information can be obtained in [120]

I.2.3 Lyapunov Stability Theory

We present in this section some concepts about Lyapunov stability theory. The following definitions and theorem were extracted from [84].

I.2.3.1 Concepts of Stability

We consider systems described by ordinary differential equations of the form

$$\dot{x} = f\left(t, x\right), \ x\left(t_0\right) = x_0 \tag{I.9}$$

where $x \in \Re^n, f : \tau \times B\left(r\right)$, $\tau = [t_0, \infty)$, and $B\left(r\right) = \left\{x \in \Re^n | \, \|x\| < r\right\}$. We assume that f is of such nature that for every $x_0 \in B\left(r\right)$ and every $t_0 \in \Re^+$, (I.9) have one and only one solution $x\left(t; t_0; x_0\right)$.

Definition 3. *A state x_e is said to be an* ***equilibrium state*** *of the system described by (I.9) if*

$$f\left(t, x_e\right) \equiv 0 \ for \ all \ \ t \geq t_0 \tag{I.10}$$

Definition 4. *An equilibrium state x_e is called an* ***isolated equilibrium state*** *if there exists a constant $r > 0$ such that $B\left(x_e, r\right) := \left\{x | \, \|x - x_e\| < r\right\}$ contains no equilibrium state of (I.9) other than x_e.*

Definition 5. *The equilibrium state x_e is said to be* ***stable****(in the sense of Lyapunov) if for arbitrary t_0 and $\varepsilon > 0$ there exists a $\delta\left(\varepsilon, t_0\right)$ such that $\|x_0 - x_e\| < \delta$ implies $\|x\left(t; t_0; x_0\right) - x_e\|$ for all $t \geq t_0$.*

Definition 6. *The equilibrium state x_e is said to be* ***uniformly stable (u.s)*** *if it is stable and if $\delta\left(\varepsilon, t_0\right)$ in Definition 5 does not depend on t_0.*

Definition 7. *The equilibrium state x_e is said to be* ***asymptotically stable (a.s)*** *if (i) it is stable, and (ii) there exists a $\delta(t_0)$ such that $\|x_0 - x_e\| < \delta(t_0)$ implies $lim_{n\to\infty} \|x(t;t_0;x_0) - x_e\| = 0$. If condition (ii) is satisfied, then the equilibrium state x_e is said to be* ***attractive****.*

Definition 8. *The set of all $x_0 \in \Re^n$ such that $x(t;t_0;x_0) \to x_e$ as $t \to \infty$ for some $t_0 \geq 0$ is called the* ***region of attraction*** *of the equilibrium state x_e.*

Definition 9. *The equilibrium state x_e is said to be* ***uniformly asymptotically stable (u.a.s)*** *if (i) it is uniformly stable, (ii) for every $\varepsilon > 0$ and any $t_0 \in \Re^+$, there exist a $\delta > 0$, independent of t_0 and ε and $T(\varepsilon) > 0$, independent of t_0, such that $\|x(t;t_0;x_0) - x_e\| < \varepsilon$ for all $t > t_0 + T(\varepsilon)$ whenever $\|x_0 - x_e\| < \delta_0$.*

Definition 10. *The equilibrium state x_e is* ***exponentially stable (e.s)*** *if there exists an $\alpha > 0$, and for every $\varepsilon > 0$ there exists a $\delta(\varepsilon) > 0$ such that*

$$\|x(t;t_0;x_0) - x_e\| < \varepsilon e^{-\alpha(t-t_0)} \text{ for all } t \geq t_0 \tag{I.11}$$

whenever $\|x_0 - x_e\| < \delta(\varepsilon)$.

Definition 11. *The equilibrium state x_e is said to be* ***unstable*** *if it is not stable. When (I.9) have a unique solution for each $x_0 \in \Re^n$ and $t_0 \in \Re^+$, we need the following definitions for the global characterization of solutions.*

Definition 12. *A solution $x(t;t_0;x_0)$ of (I.9) is* ***bounded*** *if there exists a $\beta > 0$ such that $\|x(t;t_0;x_0) - x_e\| < \beta$ for all $t > t_0$, where β may depend on each solution.*

Definition 13. *The solutions of (I.9) are* ***uniformly bounded (u.b)*** *if for any $\alpha > 0$ and $t_0 \in \Re^+$, there exists a $\beta = \beta(\alpha)$, independent of t_0 , such that if $\|x_0\| < \alpha$, then $\|x(t;t_0;x_0) - x_e\| < \beta$ for all $t > t_0$.*

Definition 14. *The solutions of (I.9) are* ***uniformly ultimately bounded (u.u.b)*** *(with bound B) if there exists a $B > 0$ and if corresponding to any $\alpha \geq 0$ and $t_0 \in \Re^+$, there exists a $T = T(\alpha) > 0$ (independent of t_0) such that $\|x_0\| < \alpha$ implies $\|x(t;t_0;x_0)\| < B$ for all $t > t_0 + T$.*

Definition 15. *If $x(t;t_0;x_0)$ is a solution of $\dot{x} = f(t,x)$, then the trajectory $x(t;t_0;x_0)$ is said to be* ***stable (u.s., a.s., u.a.s., e.s., unstable)*** *if the equilibrium point $z_e = 0$ of the differential equation*

$$\dot{z} = f(t, z + x(t;t_0;x_0)) - f(t, x(t;t_0;x_0)) \tag{I.12}$$

is ***stable (u.s., a.s., u.a.s., e.s., unstable, respectively)****.*

I.2.3.2 Lyapunov's Direct Method

The stability properties of the equilibrium state or solution of (I.9) can be studied by using the direct method of Lyapunov (also known as Lyapunov's second method). The objective of this method is to answer questions of stability by using the form of $f(t,x)$ in (I.9) rather than the explicit knowledge of the solutions. We start with the following definitions.

Definition 16. *A continuous function $\varphi : [0,r] \to \Re^+$ (or a continuous function $\varphi : [0,\infty) \to \Re^+$) is said to belong to **class** K, $\varphi \in K$, if*

(i) $\varphi(0) = 0$.

(ii) φ is strictly increasing on $[0,r]$ (or on $[0,\infty)$).

Definition 17. *A continuous function $\varphi : [0,\infty) \to \Re^+$ is said to belong to **class** KR, $\varphi \in KR$, if*

(i) $\varphi(0) = 0$.

(ii) φ is strictly increasing on $[0,\infty)$.

(iii) $lim_{r\to\infty}\varphi(r) = \infty$.

Definition 18. *Two functions $\varphi_1, \varphi_2 \in K$ defined on $[0,r]$ (or on $[0,\infty]$) are said to be of the **same order of magnitude** if there exist positive constants k_1, k_2, such that*

$$k_1\varphi_1(r_1) \leq \varphi_2(r_1) \leq k_2\varphi_2(r_1), \forall r_1 \in [0,r]\,(ou\forall r_1 \in [0,\infty]) \tag{I.13}$$

Definition 19. *A function $V(t,x) : \Re^+ \times B(r) \to \Re$ with $V(t,0) = 0, \forall t \in \Re^+$ is **positive definite** if there exists a continuous function such that $V(t,x) \geq \varphi(x), \forall t \in \Re^+$, $x \in B(r)$ and some $r > 0$. $V(t,x)$ is called **negative-definite** if $-V(t,x)$is positive definite.*

Definition 20. *A function $V(t,x) : \Re^+ \times B(r) \to \Re$ with $V(t,0) = 0, \forall t \in \Re^+$ is said to be **positive(negative) semidefinite** if $V(t,x) \geq 0 (V(t,x) \leq 0), \forall t \in \Re^+$ for all $t \in \Re^+$ and $x \in B(r)$ for some $r > 0$.*

Definition 21. *A function $V(t,x) : \Re^+ \times B(r) \to \Re$, $V(t,0) = 0, \forall t \in \Re^+$ is said to be **decreasing** if there exists $\varphi \in K$ such that $V(t,x) \leq \varphi(x), \forall t \geq 0$ and $\forall x \in B(r)$ for some $r > 0$.*

Definition 22. *A function $V(t,x) : \Re^+ \times \Re^r \to \Re$ with $V(t,0) = 0, \forall t \in \Re^+$ is said to be **radially unbounded** if there exists $\varphi \in KR$ such that for all.*

It is clear from the Definition (22) that if $V(t,x)$ is radially unbounded, it is also positive definite for all $x \in \Re^n$, but the converse is not true.

Let us assume (without loss of generality) that $x_e = 0$ is an equilibrium point of (I.9) and define $\dot{V}$ to be the time derivative of the function $V(t,x)$ along the solution of (I.9), so

$$\dot{V} = \frac{\partial V}{\partial t} + (\nabla V)^T f(t,x) \tag{I.14}$$

where $\nabla V = \left[\frac{\partial V}{\partial x_1}, \frac{\partial V}{\partial x_2}, ..., \frac{\partial V}{\partial x_n}\right]$ is the gradient of V with respect to x. The second method of Lyapunov is summarized by the following theorem.

Theorem I.2.1. *Suppose there exists a positive definite function $V(t,x) : \Re^+ \times B(r) \rightarrow \Re$ for some $r > 0$ with continuous first-order partial derivatives with respect to x, t, and $V(t,0) = 0, \forall t \in \Re^+$. Then, the following statements are true:*

(i) If $\dot{V} \leq 0$, then x_e is ***stable****.*

(ii) If V is decreasing and $\dot{V} \leq 0$, then x_e is ***uniformly stable****.*

(iii) If V is decreasing and $\dot{V} < 0$, then x_e is ***uniformly asymptotically stable****.*

(iv) If V is decreasing and there exist $\varphi_1, \varphi_2, \varphi_3 \in K$ of the same order of magnitude such that

$$\varphi_1(|x|) \leq V(t,x) \leq \varphi_2(|x|), V(t,x) \leq -\varphi_3(|x|) \tag{I.15}$$

for all $x \in B(r)$ and $t \in \Re^+$, then $x_e = 0$ is exponentially stable.

In the above theorem, the state is restricted to be inside the ball $B(r)$ for some $r > 0$. Therefore, the results (i) to (iv) of Theorem I.2.1 are referred to as local results.

Theorem I.2.2. *Assume that (I.9) possesses unique solutions for all $x_0 \in \Re^n$. If there exists a function $V(t,x)$ defined on $|x| \geq R$ and $t \in [0,\infty)$ with continuous first-order partial derivatives with respect to x, t and if there exist $\varphi_1, \varphi_2 \in KR$ such that*

(i) $\varphi_1(|x|) \leq V(t,x) \leq \varphi_2(|x|)$

(ii) $\dot{V}(t,x) \leq 0$ for all $|x| \geq R$ and $t \in [0,\infty)$, then, the solutions of (I.9) are uniformly bounded. If in addition there exists $\varphi_3 \in K$ defined on $[0,\infty)$ and

(iii) $\dot{V}(t,x) \leq \varphi_3(|x|)$ for all $|x| \geq R$ and $t \in [0,\infty)$ then, the solutions of (I.9) are uniformly ultimately bounded.

I.3 Artificial Neural Networks

Initially will be shown some concepts about neural networks biological that will ultimately serve as motivation for artificial neural network which will be discussed next. The concepts and figures used in this section were taken from [1].

I.3.1 Biological Neural Networks

Neurons, or nerve cells, are the building blocks of the nervous system. Neurons have unique features and structures that differentiate them from other cells. The neuron has three distinct regions:

- Cell body (or soma): provides the support functions and structure of the cell, it collects and process information received from other neurons.

- Dendrites: are tube like extensions that branch repeatedly and form a bushy tree around the cell body. They provide the main path on which the neuron receives coming information.
- Axon: the part of the neuron that extends away from the cell body and provides the path over which information travel to other neurons.

The figure I.1 shows a biological neuron.

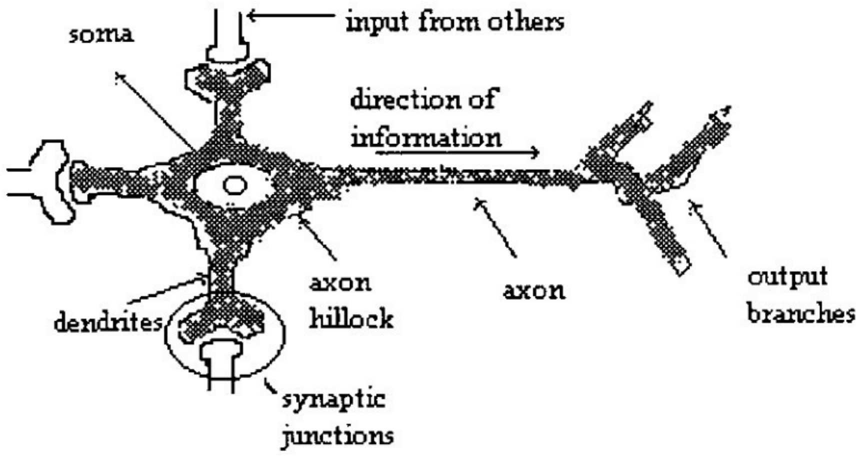


Figure I.1: Biological Neuron Scheme [1]

A nerve impulse is triggered, at the origin of the axon, by the cell body in response to received information. The impulse sweeps along the axon until it reaches the end. The junction point of an axon with a dendrite of another neuron is called a synapse.

I.3.2 Artificial Neural Models

An artificial neural network (ANN) is a massively parallel distributed processor, inspired from biological neural networks, which can store experimental knowledge and makes it available for use. The similarities with the brain are:

- Knowledge is acquired through a learning process.
- Interneuron connectivity named as synaptic weights are used to store this knowledge.

The procedure for the learning process is known as a learning algorithm. Its function is to modify the synaptic weights of the networks in order to attain a prespecified goal. The weights modification provides the traditional method for neural networks design and implementation. The neuron is the fundamental information-processing unit for the operation of a neural network [1, 2]. The figure I.2 shows the model of a neuron.

This model has three basic elements:

- A set of synapses links, each element being characterized by its own weight or strength.

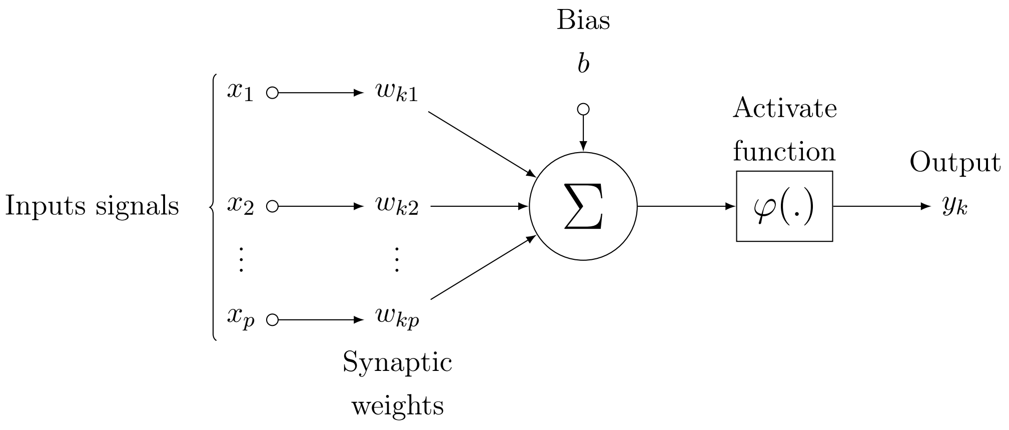


Figure I.2: Nonlinear model of a neuron [2]

- An adder for summing the inputs signal components, multiplied by the respective synapsis weight.
- A nonlinear activation function transforming the adder output into the output of the neuron.

The neuron scheme presented in figure I.2 also includes an externally applied bias or threshold, denoted by b. Bias can increase or decrease the input of the activation function, depending on whether it is positive or negative, respectively.

I.3.3 Linearly Parameterized Neural Networks

Linearly parameterized neural networks (LPNNs) can be expressed mathematically as

$$\rho_{nn}\left(\hat{W}, \zeta\right) = \hat{W}\sigma\left(\zeta\right) \tag{I.16}$$

where $\rho_{nn} : \Re^{L_{nn}} \mapsto \Re^{n}$ is a function, $\hat{W} \in \Re^{n\times L_\rho}$ is a weight matrix, $\zeta \in \Re^{L_\zeta}$ are the inputs of the neural network and $\sigma : \Re^{L_\zeta} \mapsto \Re^{L_\rho}$ is the basis function vector, which can be considered as a nonlinear vector function whose arguments are preprocessed by a scalar function $s(\cdot)$, and $n, L_\rho, L_\zeta, L_{nn}$ are integers strictly positive. Commonly used scalar functions include sigmoid (used in this work), hyperbolic tangent, gaussian, Hardy's, inverse Hardy's multiquadratic [19]. However, here we are only interested in the class of LPNNs for which $\sigma(\cdot)$ is bounded, since in this case we have,

$$\left\|\sigma\left(\zeta\right)\right\| \leq \sigma_0 \tag{I.17}$$

being σ_0 a strictly positive constant.

The class of LPNNs considered in this work includes radial basis function neural networks (RBFs), wavelet networks, high order neural nertworks (HONNs) [19, 20, 121], and also others linearly parameterized approximators as Takagi-Sugeno fuzzy systems [122]. Universal approximation results in [19, 20, 121, 122] indicate that:

Property 1. *Given a constant $\varepsilon_0 > 0$ and a continuous function $f : \Re^{L_\zeta} \mapsto \Re^n$ there exists a weight matrix $\hat{W} = W^*$ and a L_ρ is big enough such that*

$$\left\| f(\zeta) - \hat{W}\sigma(\zeta) \right\|_\infty \leq \varepsilon_0 \tag{I.18}$$

where $\hat{W} \in \Gamma$, $\zeta \in \Omega$, $\Gamma = \left\{ \hat{W} | \left\| \hat{W} \right\| \leq \alpha_{\hat{W}} \right\}$, $\alpha_{\hat{W}}$ is a positive constant, $\hat{W}$ is the estimation of W^*, which is an "optimum" matrix, and ε_0 is an approximation, reconstruction or modeling error.

Property 2. *The Output of LPNNs is continuous with respect to their arguments and satisfies the condition of Lipschitz [84], for all $\zeta \in \Omega(\zeta)$, where Ω is a compact set.*

I.3.4 Neural Network Structures

The way in which the neurons of a neural network are interconnected determines its structure. The commonly used static neural network structures for system identification are multilayer perceptron, fuzzy sytems, radial basis function and wavelet networks.

I.3.4.1 Multilayer Feedforward Neural Network

They distinguish themselves by the presence of one or more hidden layer whose computation nodes are called hidden neurons. Typically the neurons in each layer have as their inputs the output signals of the preceding layer. If each neuron in each layer is connected to every neurone in the adjacent forward layer, then the neural network is named as fully connected, on the opposite case, it is called partly connected. A multilayer perceptron (MLP) has three distinctive characteristics:

- The activation function of each neuron is smooth as opposed to the hard limit used in the single layer perceptron. Usually, this nonlinear function is a sigmoidal.
- The network contains one or more layers of hidden neurons.
- The network exhibits a high degree of connectivity.

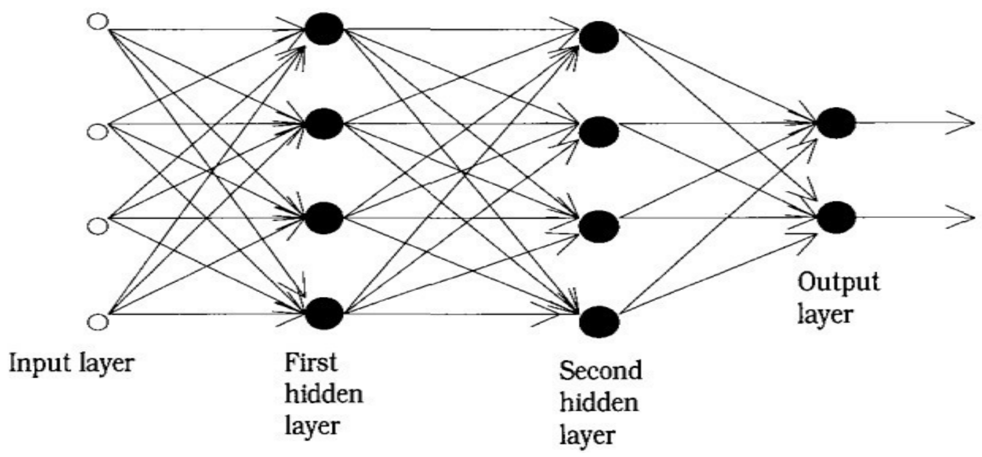


Figure I.3: Multilayer Perceptron [1]

In this dissertation we use systems with one hidden layer.

I.3.5 Sigmoidal functions

In this section we extract some informations from [123]. The activation functions usually employed in multilayer perceptron are the sigmoidals.

Some Sigmoidal functions	
Name	**Formula**
Logistic	$\frac{1}{1+e^{-\gamma x}},\ \gamma > 0$
Hyperbolic Tangent	$\frac{e^x - e^{-x}}{e^x + e^{-x}}$
Arc-tangent	$arctan\ x$

Table I.1: Common Sigmoidal Activation Functions for MLP Networks

In this work, all activation functions are logistic type.

I.4 Systems Identification

Initially will be shown some concepts about systems identification. The concepts and figures used in this section were taken from [3].

I.4.1 Theoretical and Experimental Modeling

In the following figure there are the different kinds of mathematical models.

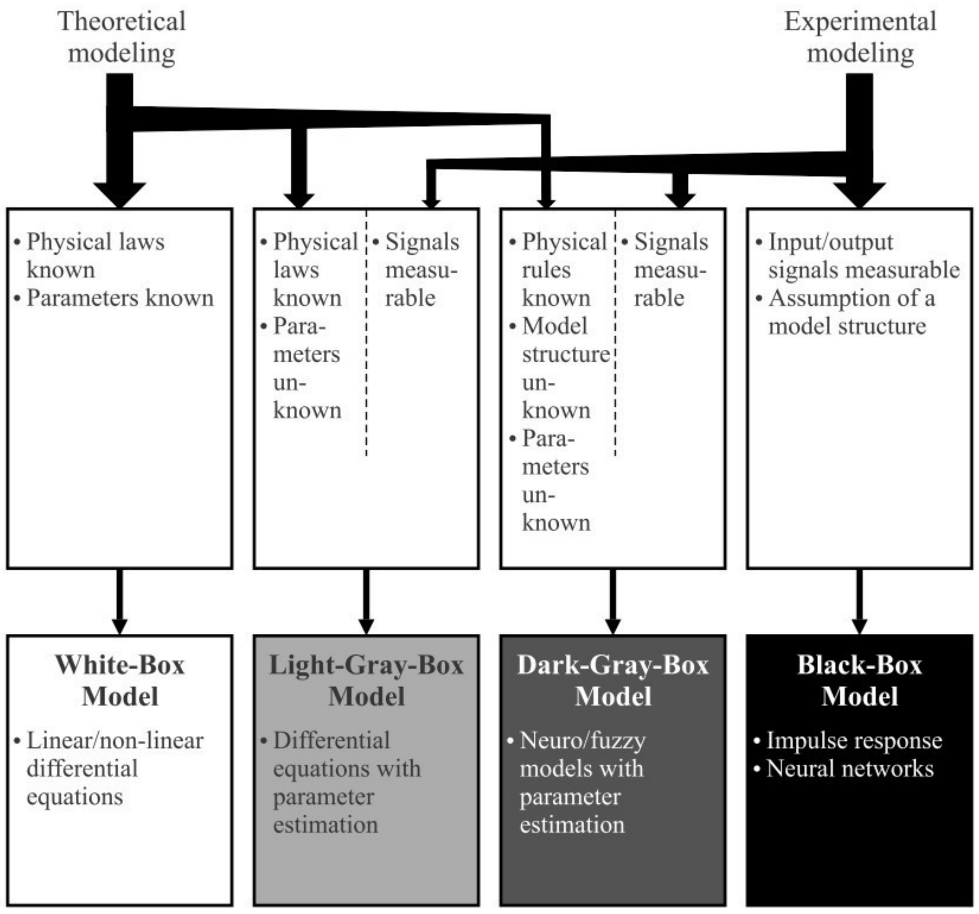


Figure I.4: Different kinds of mathematical models [3]

Despite the fact that the theoretical analysis can in principle deliver more information about the system, provided that the internal behavior is known and can be described mathematically, experimental analysis has found ever increasing attention over the past 50 years. The main reasons are the following:

- Theoretical analysis can become quite complex even for simple systems.
- Mostly, model coefficients derived from the theoretical considerations are not precise enough.
- Not all actions taking place inside the system are known.
- The actions taking place cannot be described mathematically with the required accuracy.
- Some systems are very complex, making the theoretical analysis too time consuming.
- Identified models can be obtained in shorter time with less effort compared to theoretical modeling

The experimental analysis allows the development of mathematical models by measurement of the input and output of systems of arbitrary composition. One major advantage is the fact that the same experimental analysis methods can be applied to diverse and arbitrarily complex systems. By measuring the input and output only, one does however only obtain models governing the input-output behavior of the system, i.e. the models will in general not describe the precise internal structure of the system. These input-output models are approximations and are still sufficient for many areas of application. If the system also allows the measurement of internal states, one can obviously also gather information about the internal structure of the system. With the advent of digital computers starting in the 1960s, the development of capable identification methods has started.

I.4.2 Offline and Online Identification

If digital computers are utilized for the identification, then one differentiates between two types of coupling between process and computer, see figure I.5:

- Offline (indirect coupling)
- Online (direct coupling)

For the offline identification, the measured data are first stored (e.g. data storage) and are later transferred to the computer utilized for data evaluation and are processed there.

The online identification is performed parallelly to the experiment. The computer is coupled with the process and the data points are operated on as they become available. All algorithms used in this work are of online type.

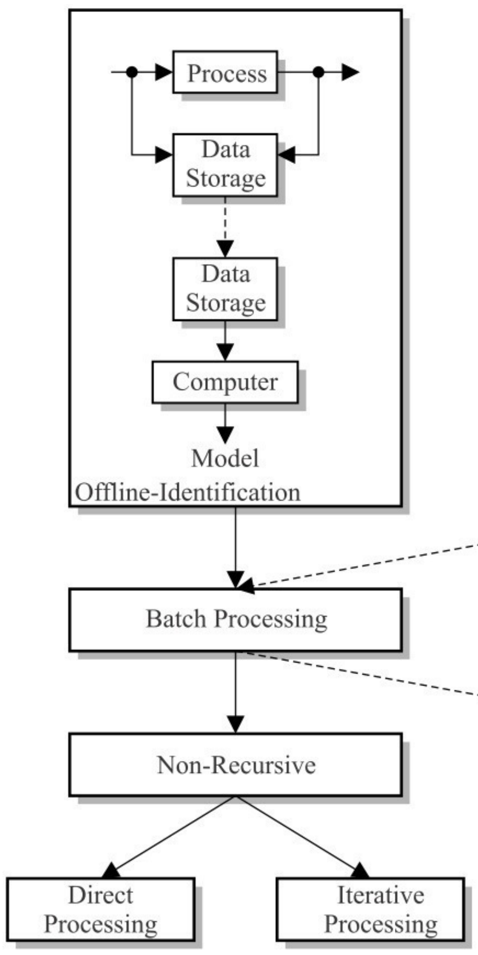


Figure I.5: Different setups for the data processing as part of the identification [3]

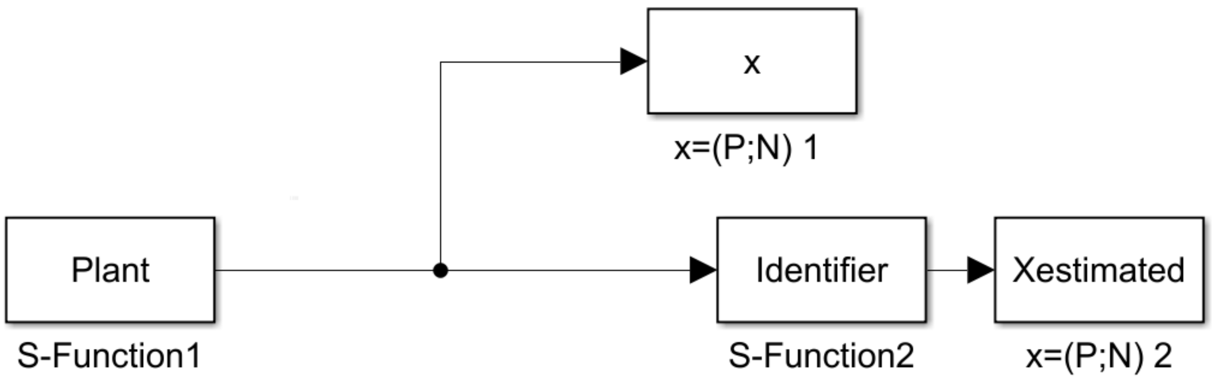
I.5 Transient State, Steady-State, and Unsteady-State Response

The following definition were extracted from [124].

Two parts compose a system response in the time domain, transient, steady-state or unsteady-state. Transient is the immediate system response to an input from an equilibrium state. After the transient state, a system response can assume a steady-state or unsteady-state. In a stable system, the output tends to a constant value when $t \to \infty$. When the system response enters and stays in the threshold around the constant value the system reached the steady-state . The time the stable system takes to reach the steady-state is the settling time, t_s. On the other hand, if the response never reaches a final value or oscillates surpassing the threshold when $t \to \infty$ the system is then at unsteady-state. Consequently, the system outputs at unsteady-state vary with time during the on-time interval even induced by an invariable input.

II. CODES

II.1 Simulink plant used for simulations corresponding to Figures 3.1 - 3.18



II.2 Code for plant model corresponding to Figures 3.1-3.8

```
function [sys,x0,str,ts] = Plant(t,x,u,flag)

%System extract from Akgul, A., Hussain, S. and Pehlivan, I., "A new ...
    three−dimensional
%chaotic system, its dynamical analysis and electronic circuit
%applications", Optik, Volume 127, Issue 18, Pages 7062−7071, 2016.

a=1.8;  %Constants
b=−0.07;
d=1.5;
m=0.12;

switch flag,
   %%%%%%%%%%%%%%%%%%
   % Initialization %
   %%%%%%%%%%%%%%%%%%
case 0,
    sizes = simsizes;
    sizes.NumContStates = 3; %Number of Constant States
    sizes.NumDiscStates = 0; %Number of Discret States
    sizes.NumOutputs = 3;    %Number of Outputs
    sizes.NumInputs = 0;     %Number of Inputs
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
    sys = simsizes(sizes);
    x0=[2 2 2]; %Initial Conditions
    str=[];
    ts=[0 0];
   %%%%%%%%%%%%%%%
   % Directives  %
   %%%%%%%%%%%%%%%
case 1,     %Chaotic System
  sys = [a*(x(1) - x(2));
        -4*a*x(2) + x(1)*x(3) + m*x(1)^3;
        -a*d*x(3) + (x(1)^3)*x(2) + b*x(3)^2] + disturb(x,u,t);
   %%%%%%%%%%%
   % Outputs %
   %%%%%%%%%%%
case 3,
   sys = x;
   %%%%%%%%%
   %  End  %
   %%%%%%%%%
case {2,4,9},
    sys = []; % do nothing
 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end

function disturb = disturb(x,u,t) %disturb
if t>=5
    n=2;
    disturb=[0.5*n*cos(2*t); n*sin(t); n*sin(2*t)];
else
    disturb=[0; 0; 0];
end
```

II.3 Code for identifier corresponding to Figures 3.1, 3.3, 3.5 and 3.7

```
function [sys,x0,str,ts] = Identifier(t,x,u,flag)

%Controller and its parameters
L = 2*[1 0 0; 0 1 0; 0 0 1];
P = 5*[1 0 0; 0 1 0; 0 0 1];
GAMAW=0.5;
GAMA0=1;
G=1;
W01=G*[1 0 0 0 0 0]'; %W zero
W02=G*[0 1 0 0 0 0]';
W03=G*[0 0 1 0 0 0]';
```

```
switch flag,
   %%%%%%%%%%%%%%%%%%%
   % Initialization  %
   %%%%%%%%%%%%%%%%%%%
case 0,

    sizes = simsizes;
    sizes.NumContStates  = 21;  %Number of Constant States
    sizes.NumDiscStates  = 0;   %Number of Discret States
    sizes.NumOutputs     = 4;   %Number of Outputs
    sizes.NumInputs      = 3;   %Number of Inputs
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
    sys = simsizes(sizes);

    x0=zeros(21,1); %
    x0(1)=-2;
    x0(2)=-3;
    x0(3)=-4;
        str=[];
    ts=[0 0];
   %%%%%%%%%%%%%%%
   % Directives  %
   %%%%%%%%%%%%%%%
case 1,
      %Identification Model
      sys = [-L*[x(1)-u(1);x(2)-u(2);x(3)-u(3)] - ...
          GAMAW*GAMA0*[x(1)-u(1);x(2)-u(2);x(3)-u(3)] + ...
          P*[x(4:9)';x(10:15)';x(16:21)']*S(x,u);
         %Learning Law
         -2*GAMAW*(GAMA0*(x(4:9)-W01) + (x(1)-u(1))*S(x,u));
         -2*GAMAW*(GAMA0*(x(10:15)-W02) + (x(2)-u(2))*S(x,u));
         -2*GAMAW*(GAMA0*(x(16:21)-W03) + (x(3)-u(3))*S(x,u))];
   %%%%%%%%%%%
   % Outputs %
   %%%%%%%%%%%
  case 3,
    sys = [x(1:3);
           norm([x(4:9)';x(10:15)';x(16:21)'],'fro')];
case {2,4,9},
    sys = [];

 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function S = S(x,u)     %Regressors
S=[1*(z(u(1)));
   1*(z(u(2)));
   1*(z(u(3)));
   1*(z(u(1))^2);
   1*(z(u(2))^2);
   1*(z(u(3))^2)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function z = z(uu)  %Sigmoidal Function
lambda=0;
alfa=5;
beta=.5;
z=alfa/(exp(-beta*uu)+1)+lambda;
```

II.4 Code for identifier corresponding to Figures 3.2, 3.4, 3.6 and 3.8

```
function [sys,x0,str,ts] = Identifier(t,x,u,flag)

%Controller and its parameters
L = 2*[1 0 0; 0 1 0; 0 0 1];
P = 5*[1 0 0; 0 1 0; 0 0 1];
GAMAW=15;
GAMA0=1;
G=1;
W01=G*[1 0 0 0 0 0]'; %W zero
W02=G*[0 1 0 0 0 0]';
W03=G*[0 0 1 0 0 0]';

switch flag,
   %%%%%%%%%%%%%%%%%%%
   % Initialization  %
   %%%%%%%%%%%%%%%%%%%
case 0,

    sizes = simsizes;
    sizes.NumContStates  = 21;  %Number of Constant States
    sizes.NumDiscStates  = 0;   %Number of Discret States
    sizes.NumOutputs     = 4;   %Number of Outputs
    sizes.NumInputs      = 3;   %Number of Inputs
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
    sys = simsizes(sizes);

    x0=zeros(21,1); %
    x0(1)=-2;
    x0(2)=-3;
    x0(3)=-4;
        str=[];
    ts=[0 0];
   %%%%%%%%%%%%%%%
   % Directives  %
   %%%%%%%%%%%%%%%
case 1,
      %Identification Model
      sys = [-L*[x(1)-u(1);x(2)-u(2);x(3)-u(3)] - ...
          GAMAW*GAMA0*[x(1)-u(1);x(2)-u(2);x(3)-u(3)] + ...
          P*[x(4:9)';x(10:15)';x(16:21)']*S(x,u);
         %Learning Law
         -2*GAMAW*(GAMA0*(x(4:9)-W01) + (x(1)-u(1))*S(x,u));
         -2*GAMAW*(GAMA0*(x(10:15)-W02) + (x(2)-u(2))*S(x,u));
         -2*GAMAW*(GAMA0*(x(16:21)-W03) + (x(3)-u(3))*S(x,u))];
   %%%%%%%%%%%
   % Outputs %
   %%%%%%%%%%%
  case 3,
    sys = [x(1:3);
           norm([x(4:9)';x(10:15)';x(16:21)'],'fro')];
case {2,4,9},
    sys = [];

 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function S = S(x,u)     %Regressors
S=[1*(z(u(1)));
   1*(z(u(2)));
   1*(z(u(3)));
   1*(z(u(1))^2);
   1*(z(u(2))^2);
   1*(z(u(3))^2)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function z = z(uu)  %Sigmoidal Function
lambda=0;
alfa=5;
beta=.5;
z=alfa/(exp(-beta*uu)+1)+lambda;
```

II.5 Code to plot the Figures 3.1, 3.3, 3.5 and 3.7

```
%Shown the graphs of the simulation
clc
fsize=20;

%Figure 1
fig=figure;
plot(t,x(:,1),t, Xestimated(:,1),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
```

```
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{1}(t), \hat{x}_{1}(t)$$','Interpreter','Latex','Fontsize',fsize)

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [3 3];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,3.8,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,3.3,'in action','Fontsize',fsize) % write a text on top of the arrow

set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG31.png');
close(fig)

%Figure 2
fig=figure;
plot(t,x(:,2),t, Xestimated(:,2),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [5 5];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,7.1,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,5.8,'in action','Fontsize',fsize) % write a text on top of the arrow
```

```
ylabel('$$x_{2}(t), \hat{x}_{2}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG33.png');
close(fig)

%Figure 3
fig=figure;
plot(t,x(:,3),t, Xestimated(:,3),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','northeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [25 25];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,28.5,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,26.5,'in action','Fontsize',fsize) % write a text on top of the arrow

ylabel('$$x_{3}(t), \hat{x}_{3}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG35.png');
close(fig)

%Figure 4
fig=figure;
plot(t,Xestimated(:,4),'LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('Estimated Weight Norm','Fontsize',fsize)

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [3 3];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,3.35,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,3.15,'in action','Fontsize',fsize) % write a text on top of the arrow

set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG37.png');
close(fig)
```

II.6 Code to plot the Figures 3.2, 3.4, 3.6 and 3.8

```
%Shown the graphs of the simulation
clc
fsize=20;

%Figure 1
fig=figure;
plot(t,x(:,1),t, Xestimated(:,1),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{1}(t), \hat{x}_{1}(t)$$','Interpreter','Latex','Fontsize',fsize)

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [3 3];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,3.75,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,3.3,'in action','Fontsize',fsize) % write a text on top of the arrow

set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG32.png');
```

```
close(fig)

%Figure 2
fig=figure;
plot(t,x(:,2),t, Xestimated(:,2),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [5 5];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,6.8,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,5.8,'in action','Fontsize',fsize) % write a text on top of the arrow

ylabel('$$x_{2}(t), \hat{x}_{2}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG34.png');
close(fig)

%Figure 3
fig=figure;
plot(t,x(:,3),t, Xestimated(:,3),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','northeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [25 25];
pa.LineWidth  = 2;          % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,28.5,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,26.5,'in action','Fontsize',fsize) % write a text on top of the arrow

ylabel('$$x_{3}(t), \hat{x}_{3}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG36.png');
close(fig)

%Figure 4
fig=figure;
plot(t,Xestimated(:,4),'LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('Estimated Weight Norm','Fontsize',fsize)

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [3 3];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,4.5,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,3.6,'in action','Fontsize',fsize) % write a text on top of the arrow

set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG38.png');
close(fig)
```

II.7 Code for plant model corresponding to Figures 3.9 - 3.18

```
function [sys,x0,str,ts] = Plant(t,x,u,flag)

%System extract from Yu, H., Cai, G. and Li, Y., "Dynamic analysis and
%control of a new hyperchaotic ?nance control system", Nonlinear Dyn,
%Volume 67, Issue 3, Pages 2171-2182, 2012.
```

```
a=0.9; %Constants
b=0.2;
c=1.5;
d=0.2;
k=0.17;

switch flag,
   %%%%%%%%%%%%%%%%%%
   % Initialization %
   %%%%%%%%%%%%%%%%%%
case 0,
    sizes = simsizes;
    sizes.NumContStates = 4; %Number of Constant States
    sizes.NumDiscStates = 0; %Number of Discret States
    sizes.NumOutputs = 4;    %Number of Outputs
    sizes.NumInputs = 0;     %Number of Inputs
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
    sys = simsizes(sizes);
    x0=[1 2 0.5 0.5]; %Initial Conditions
    str=[];
    ts=[0 0];
   %%%%%%%%%%%%%%%
   % Directives  %
   %%%%%%%%%%%%%%%
case 1,     %Hyperchaotic System
    sys= [ x(3)+(x(2)-a)*x(1)+x(4);
           1-b*x(2)-(x(1))^2;
           -x(1)-c*x(3);
           -d*x(1)*x(2)-k*x(4)]+disturb(x,u,t);
   %%%%%%%%%%%
   % Outputs %
   %%%%%%%%%%%
case 3,
   sys = x;
   %%%%%%%%
   % End  %
   %%%%%%%%
case {2,4,9},
    sys = []; % do nothing
 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end

function disturb = disturb(x,u,t) %disturb
if t>=5
    n=4;
    disturb=n*[0.7*cos(6*t);cos(5*t);0.9*cos(4*t);0.8*sin(3*t)];
else
    disturb=[0 ; 0; 0; 0];
end
```

II.8 Code for identifier corresponding to Figures 3.9, 3.11, 3.13, 3.15 and 3.17

```
function [sys,x0,str,ts] = Identifier(t,x,u,flag)

%Controller and its parameters
L = 2*[1 0 0 0; 0 1 0 0;0 0 1 0; 0 0 0 1];
P = 20*[1 0 0 0; 0 1 0 0;0 0 1 0; 0 0 0 1];
GAMAW=0.5;
GAMA0=1;
G=1;
W01=G*[1 0 0 0 0 0 0 0]'; %W zero
W02=G*[0 1 0 0 0 0 0 0]';
W03=G*[0 0 1 0 0 0 0 0]';
W04=G*[0 0 0 1 0 0 0 0]';

switch flag,
   %%%%%%%%%%%%%%%%%%%
   % Initialization  %
   %%%%%%%%%%%%%%%%%%%
case 0,

    sizes = simsizes;
    sizes.NumContStates  = 36;  %Number of Constant States
    sizes.NumDiscStates  = 0;   %Number of Discret States
    sizes.NumOutputs     = 5;   %Number of Outputs
    sizes.NumInputs      = 4;   %Number of Inputs
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
    sys = simsizes(sizes);

    x0=zeros(36,1); %
    x0(1)=-2;
    x0(2)=-2;
    x0(3)=-2;
    x0(4)=-2;
        str=[];
    ts=[0 0];
   %%%%%%%%%%%%%%%
   % Directives  %
   %%%%%%%%%%%%%%%
case 1,
      %Identification Model
      sys = [-L*[x(1)-u(1);x(2)-u(2);x(3)-u(3);x(4)-u(4)] - ...
          GAMAW*GAMA0*[x(1)-u(1);x(2)-u(2);x(3)-u(3);x(4)-u(4)] + ...
          P*[x(5:12)';x(13:20)';x(21:28)';x(29:36)']*S(x,u);
          %Learning Law
          −2*GAMAW*(GAMA0*(x(5:12)−W01) + (x(1)−u(1))*S(x,u));
          −2*GAMAW*(GAMA0*(x(13:20)−W02) + (x(2)−u(2))*S(x,u));
          −2*GAMAW*(GAMA0*(x(21:28)−W03) + (x(3)−u(3))*S(x,u));
          −2*GAMAW*(GAMA0*(x(29:36)−W04) + (x(4)−u(4))*S(x,u))];
   %%%%%%%%%%%
   % Outputs %
   %%%%%%%%%%%
  case 3,
    sys = [x(1:4);
           norm([x(5:12)';x(13:20)';x(21:28)';x(29:36)'],'fro')];
case {2,4,9},
    sys = [];

 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function S = S(x,u)     %Regressors
S=[1*(z(u(1)));
   1*(z(u(2)));
   1*(z(u(3)));
   1*(z(u(4)));
   1*(z(u(1))^2);
   1*(z(u(2))^2);
   1*(z(u(3))^2);
   1*(z(u(4))^2)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function z = z(uu)   %Sigmoidal Function
lambda=0;
alfa=5;
beta=.5;
z=alfa/(exp(−beta*uu)+1)+lambda;
```

II.9 Code for identifier corresponding to Figures 3.10, 3.12, 3.14, 3.16 and 3.18

```
function [sys,x0,str,ts] = Identifier(t,x,u,flag)

%Controller and its parameters
L = 2*[1 0 0 0; 0 1 0 0;0 0 1 0; 0 0 0 1];
P = 20*[1 0 0 0; 0 1 0 0;0 0 1 0; 0 0 0 1];
GAMAW=20;
GAMA0=1;
G=1;
W01=G*[1 0 0 0 0 0 0 0]'; %W zero
W02=G*[0 1 0 0 0 0 0 0]';
```

```
W03=G*[0  0  1  0  0  0  0  0]';
W04=G*[0  0  0  1  0  0  0  0]';

switch flag,
   %%%%%%%%%%%%%%%%%%%
   % Initialization  %
   %%%%%%%%%%%%%%%%%%%
case 0,

    sizes = simsizes;
    sizes.NumContStates  = 36;  %Number of Constant States
    sizes.NumDiscStates  = 0;   %Number of Discret States
    sizes.NumOutputs     = 5;   %Number of Outputs
    sizes.NumInputs      = 4;   %Number of Inputs
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
    sys = simsizes(sizes);

    x0=zeros(36,1); %
    x0(1)=-2;
    x0(2)=-2;
    x0(3)=-2;
    x0(4)=-2;
        str=[];
    ts=[0 0];
   %%%%%%%%%%%%%%%
   % Directives  %
   %%%%%%%%%%%%%%%
case 1,
      %Identification Model
      sys = [-L*[x(1)-u(1);x(2)-u(2);x(3)-u(3);x(4)-u(4)] - ...
          GAMAW*GAMA0*[x(1)-u(1);x(2)-u(2);x(3)-u(3);x(4)-u(4)] + ...
          P*[x(5:12)';x(13:20)';x(21:28)';x(29:36)']*S(x,u);
         %Learning Law
         -2*GAMAW*(GAMA0*(x(5:12)-W01)  +  (x(1)-u(1))*S(x,u));
         -2*GAMAW*(GAMA0*(x(13:20)-W02)  +  (x(2)-u(2))*S(x,u));
         -2*GAMAW*(GAMA0*(x(21:28)-W03)  +  (x(3)-u(3))*S(x,u));
         -2*GAMAW*(GAMA0*(x(29:36)-W04)  +  (x(4)-u(4))*S(x,u))];
   %%%%%%%%%%%
   % Outputs %
   %%%%%%%%%%%
  case 3,
    sys = [x(1:4);
           norm([x(5:12)';x(13:20)';x(21:28)';x(29:36)'],'fro')];
case {2,4,9},
    sys = [];

 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function S = S(x,u)     %Regressors
```

```
S=[1*(z(u(1)));
   1*(z(u(2)));
   1*(z(u(3)));
   1*(z(u(4)));
   1*(z(u(1))^2);
   1*(z(u(2))^2);
   1*(z(u(3))^2);
   1*(z(u(4))^2)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function z = z(uu)  %Sigmoidal Function
lambda=0;
alfa=5;
beta=.5;
z=alfa/(exp(-beta*uu)+1)+lambda;
```

II.10 Code to plot the Figures 3.9, 3.11, 3.13, 3.15 and 3.17

```
%Shown the graphs of the simulation
clc
fsize=20;

%Figure 1
fig=figure;
plot(t,x(:,1),t, Xestimated(:,1),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{1}(t), \hat{x}_{1}(t)$$','Interpreter','Latex','Fontsize',fsize)

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [3 3];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,3.54,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,3.25,'in action','Fontsize',fsize) % write a text on top of the arrow
```

```
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG39.png');
close(fig)

%Figure 2
fig=figure;
plot(t,x(:,2),t, Xestimated(:,2),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [5 5];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,5.65,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,5.3,'in action','Fontsize',fsize) % write a text on top of the arrow

ylabel('$$x_{2}(t), \hat{x}_{2}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG311.png');
close(fig)

%Figure 3
fig=figure;
plot(t,x(:,3),t, Xestimated(:,3),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];             % the location of arrow
pa.Y = [2 2];
pa.LineWidth  = 2;          % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,2.4,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,2.2,'in action','Fontsize',fsize) % write a text on top of the arrow

ylabel('$$x_{3}(t), \hat{x}_{3}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG313.png');
close(fig)

%Figure 4
fig=figure;
plot(t,x(:,4),t, Xestimated(:,4),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;            % associate the arrow the the current axes
pa.X = [5 6.6];             % the location of arrow
pa.Y = [2 2];
pa.LineWidth  = 2;          % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,2.45,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,2.2,'in action','Fontsize',fsize) % write a text on top of the arrow

ylabel('$$x_{4}(t), \hat{x}_{4}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG315.png');
close(fig)

%Figure 5
fig=figure;
plot(t,Xestimated(:,5),'LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
set(0,'DefaultAxesFontSize', 16);
ylim([0 2.5]);
```

```
xlabel('Time (s)','Fontsize',fsize);
ylabel('Estimated Weight Norm','Fontsize',fsize)

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [1 1];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,1.19,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,1.08,'in action','Fontsize',fsize) % write a text on top of the arrow

set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG317.png');
close(fig)
```

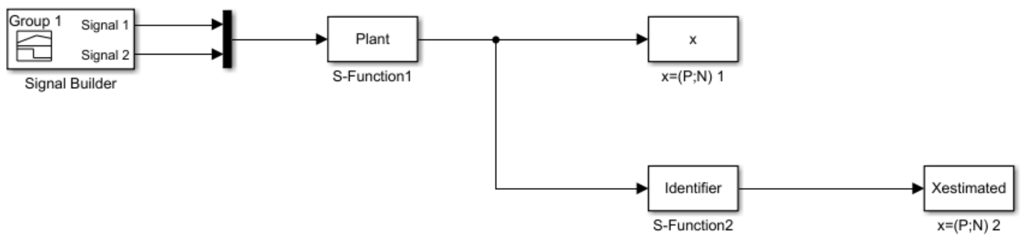
II.11 Code to plot the Figures 3.10, 3.12, 3.14, 3.16 and 3.18

```
%Shown the graphs of the simulation
clc
fsize=20;

%Figure 1
fig=figure;
plot(t,x(:,1),t, Xestimated(:,1),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{1}(t), \hat{x}_{1}(t)$$','Interpreter','Latex','Fontsize',fsize)

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [3 3];
pa.LineWidth  = 2;          % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,3.54,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,3.25,'in action','Fontsize',fsize) % write a text on top of the arrow

set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG310.png');
close(fig)

%Figure 2
fig=figure;
plot(t,x(:,2),t, Xestimated(:,2),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [5 5];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,5.65,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,5.3,'in action','Fontsize',fsize) % write a text on top of the arrow

ylabel('$$x_{2}(t), \hat{x}_{2}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG312.png');
close(fig)

%Figure 3
fig=figure;
plot(t,x(:,3),t, Xestimated(:,3),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
```

```
YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [2 2];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,2.4,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,2.2,'in action','Fontsize',fsize) % write a text on top of the arrow

ylabel('$$x_{3}(t), \hat{x}_{3}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG314.png');
close(fig)

%Figure 4
fig=figure;
plot(t,x(:,4),t, Xestimated(:,4),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [2 2];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,2.45,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,2.2,'in action','Fontsize',fsize) % write a text on top of the arrow

ylabel('$$x_{4}(t), \hat{x}_{4}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG316.png');
close(fig)
```

```
%Figure 5
fig=figure;
plot(t,Xestimated(:,5),'LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('Estimated Weight Norm','Fontsize',fsize)

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [4 4];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,4.65,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,4.3,'in action','Fontsize',fsize) % write a text on top of the arrow

set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG318.png');
close(fig)
```

II.12 Simulink plant used for simulations corresponding to Figures 3.19 - 3.29



II.13 Code for the welding system plant corresponding to Figures 3.19-3.29

```
[frame=single]
ffunction [sys,x0,str,ts] = Plant(t,x,u,flag)

%System extract from Anzehaee, M. M., Haeri, M. "Welding current and arc voltage ...
    control in a
%GMAW process using ARMarkov based MPC", Control Engineering Practice, Volume 19,
%Issue 12, Pages 1408−1422, 2011.

pi=3.14159265;
c1=3.3*10^−10;  %Constants
c2=0.78*10^−10;
re=0.6*10^−3;
Ls=306*10^−6;
tm=50*10^−3;
Ra=0.0237;
Rs=6.8*10^−3;
p=0.43;
V0=15.5;
Ea=400;
lc=0.025;
km=1;

switch flag,
   %%%%%%%%%%%%%%%%%%
   % Initialization %
   %%%%%%%%%%%%%%%%%%
case 0,
    sizes = simsizes;
    sizes.NumContStates = 4;  %Number of Constant States
    sizes.NumDiscStates = 0;  %Number of Discret States
    sizes.NumOutputs    = 6;  %Number of Outputs
    sizes.NumInputs     = 2;  %Number of Inputs
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
    sys = simsizes(sizes);
    x0=[0.01; 0; 0; 0.01]; %Initial Conditions
    str=[];
    ts=[0 0];
   %%%%%%%%%%%%%%%
   % Directives  %   % Welding System
   %%%%%%%%%%%%%%%
case 1,
   sys(1) = x(3)−( c1*x(2)/(pi*re^2) + c2*p*x(1)*x(2)^2/(pi*re^2) );         %% ...
       stick out
   sys(2) = (1/Ls)*(u(2)−(Ra+Rs+p*x(1))*x(2)−V0−Ea*(lc−x(1)));            %% ...
       welding current
   sys(3) = (1/tm)*(km*u(1)−x(3));                                         %% ...
       welding wire speed
   sys(4) = Ra*(1/Ls)*(u(2)−(Ra+Rs+p*x(1))*x(2)−V0−Ea*(lc−x(1))) − Ea*(x(3)−( ...
       c1*x(2)/(pi*re^2) + c2*p*x(1)*x(2)^2/(pi*re^2) )); %% arc voltage
   %%%%%%%%%%
   % Output %
   %%%%%%%%%%
case 3,
   sys = [x;u];
   %%%%%%%%%%%%%
   % Terminate %
   %%%%%%%%%%%%%
case {2,4,9},
    sys = [];

 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end
```

II.14 Code for identifier corresponding to Figures 3.19 - 3.29

```
function [sys,x0,str,ts] = Identifier(t,x,u,flag)

%Identifier and its parameters
L = 200*[20 0 0 0; 0 10 0 0; 0 0 2 0; 0 0 0 3];
P = 2000*[20 0 0 0; 0 10 0 0; 0 0 6 0; 0 0 0 4];
GAMAW=0.001;
GAMA0=0.001;
G=1;
W01=G*[1 0 0 0 0 0 0 0]'; %W zero
W02=G*[0 1 0 0 0 0 0 0]';
W03=G*[0 0 1 0 0 0 0 0]';
W04=G*[0 0 0 1 0 0 0 0]';

switch flag,
   %%%%%%%%%%%%%%%%%%
   % Initialization %
   %%%%%%%%%%%%%%%%%%
case 0,

    sizes = simsizes;
    sizes.NumContStates  = 36;  %Number of Constant States
    sizes.NumDiscStates  = 0;   %Number of Discret States
    sizes.NumOutputs     = 5;   %Number of Outputs
    sizes.NumInputs      = 6;   %Number of Inputs
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
    sys = simsizes(sizes);
    x0=zeros(36,1); %Initial Conditions
    x0(1)=0;
    x0(2)=0;
    x0(3)=0;
        str=[];
    ts=[0 0];
   %%%%%%%%%%%%%%%
   % Directives  %
   %%%%%%%%%%%%%%%
case 1,
        %Identifier Model
      sys = [-L*[x(1)-u(1);x(2)-u(2);x(3)-u(3);x(4)-u(4)] - ...
          GAMAW*GAMA0*[x(1)-u(1);x(2)-u(2);x(3)-u(3);x(4)-u(4)] + ...
          P*[x(5:12)';x(13:20)';x(21:28)';x(29:36)']*S(x,u);
         %Learning Law
         -2*GAMAW*(GAMA0*(x(5:12)-W01) + (x(1)-u(1))*S(x,u));
         -2*GAMAW*(GAMA0*(x(13:20)-W02) + (x(2)-u(2))*S(x,u));
         -2*GAMAW*(GAMA0*(x(21:28)-W03) + (x(3)-u(3))*S(x,u));
         -2*GAMAW*(GAMA0*(x(29:36)-W04) + (x(4)-u(4))*S(x,u))];
   %%%%%%%%%%%
   % Outputs %
   %%%%%%%%%%%
  case 3,
    sys = [x(1:4);
           norm([x(5:12)';x(13:20)';x(21:28)';x(29:36)'],'fro')];
case {2,4,9},
    sys = [];

 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function S = S(x,u)      %Regressors
S=[1*(z(u(1)));
   1*(z(u(2)));
   1*(z(u(3)));
   1*(z(u(4)));
   1*(z(u(1)))*(z(u(2)));
   1*(z(u(2))^2);
   (z(u(5)))
   (z(u(6)))];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function z = z(uu)  %Sigmoidal Function
lamda=0;
alfa=5;
beta=.5;
z=alfa/(exp(-beta*uu)+1)+lamda;
```

II.15 Code to plot the Figures 3.19 - 3.29

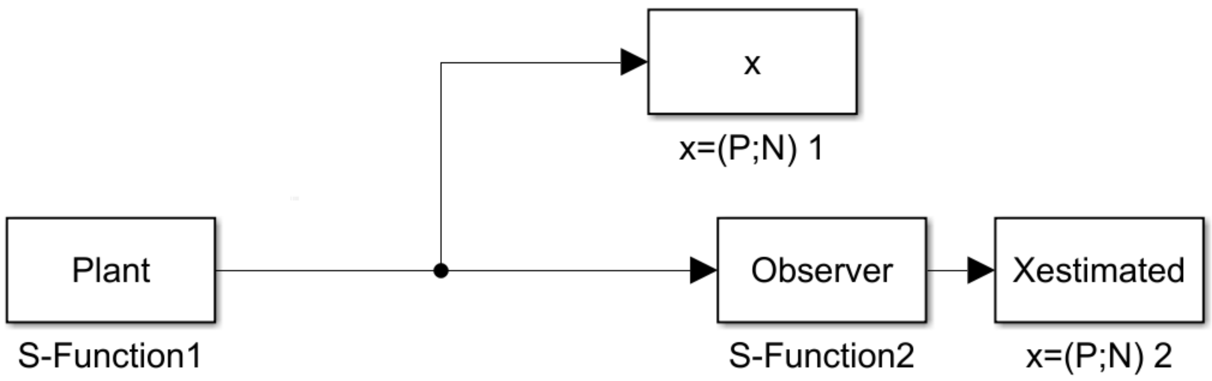
```
%Shown the graphs of the simulation
clc
fsize=20;
```

```
%Figure 1
fig=figure;
plot(t,x(:,5),'LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Input 1','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
ylim([-0.01 0.19]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$u_{1}(t)$$','Interpreter','Latex','Fontsize',fsize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG319.png');
close(fig)

%Figure 2
fig=figure;
plot(t,x(:,6),'LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Input 2','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
ylim([-2 40]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$u_{2}(t)$$','Interpreter','Latex','Fontsize',fsize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG320.png');
close(fig)

%Figure 3
fig=figure;
plot(t,x(:,1),t,Xestimated(:,1),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
ylim([0 0.026]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{1}(t), \hat{x}_{1}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG321.png');
close(fig)

%Figure 4
fig=figure;
plot(t,x(:,2),t,Xestimated(:,2),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
ylim([-600 550]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{2}(t), \hat{x}_{2}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG322.png');
close(fig)

%Figure 5
fig=figure;
plot(t,x(:,3),t,Xestimated(:,3),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
ylim([-0.01 0.19]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{3}(t), \hat{x}_{3}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG323.png');
close(fig)

%Figure 6
fig=figure;
plot(t,x(:,4),t,Xestimated(:,4),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
ylim([-16 14]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{4}(t), \hat{x}_{4}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG324.png');
close(fig)

%Figure 7
fig=figure;
semilogx(t,x(:,1),t,Xestimated(:,1),':','LineWidth',2);
set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','northwest');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlim([0.00007 8]);
ylim([0 0.026]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{1}(t), \hat{x}_{1}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
```

```
saveas(gcf,'FIG325.png');
close(fig)

%Figure 8
fig=figure;
semilogx(t,x(:,2),t,Xestimated(:,2),':','LineWidth',2);
set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','northwest');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlim([0.00007 8]);
ylim([-600 550]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{2}(t), \hat{x}_{2}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG326.png');
close(fig)

%Figure 9
fig=figure;
semilogx(t,x(:,3),t,Xestimated(:,3),':','LineWidth',2);
set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','northwest');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlim([0.00007 8]);
ylim([-0.01 0.19]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{3}(t), \hat{x}_{3}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG327.png');
close(fig)

%Figure 10
fig=figure;
semilogx(t,x(:,4),t,Xestimated(:,4),':','LineWidth',2);
set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','northwest');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlim([0.00007 8]);
ylim([-16 14]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{4}(t), \hat{x}_{4}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG328.png');
```

```
close(fig)

%Figure 11
fig=figure;
plot(t,Xestimated(:,5),'LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
set(0,'DefaultAxesFontSize', 16);
ylim([0 0.013]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('Estimated Weight Norm','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG329.png');
close(fig)
```

II.16 Simulink plant used for simulations corresponding to Figures 4.1 - 4.8



II.17 Code for plant model corresponding to Figures 4.1 - 4.8

```
[frame=single]
function [sys,x0,str,ts] = Plant(t,x,u,flag)

%System extract from Dimassi, H. "Adaptive Unknown—Input Observers—Based ...
    Synchronization
%of Chaotic Systems for Telecommunication", IEEE TRANSACTIONS ON CIRCUITS AND ...
    SYSTEMS,
%Volume 58, Issue 4, Pages 800—812, 2011.
% Rossler System

a=0.398; %Constants
b=2;
```

```
c=4;

switch flag,
   %%%%%%%%%%%%%%%%%%
   % Initialization %
   %%%%%%%%%%%%%%%%%%
case 0,
    sizes = simsizes;
    sizes.NumContStates  = 3;   %Number of Constant States
    sizes.NumDiscStates  = 0;   %Number of Discret States
    sizes.NumOutputs     = 3;   %Number of Outputs
    sizes.NumInputs      = 0;   %Number of Inputs
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
    sys = simsizes(sizes);
    x0=[2 1 2]; %Initial Conditions
    str=[];
    ts=[0 0];
   %%%%%%%%%%%%%%%
   % Derivatives %
   %%%%%%%%%%%%%%%
case 1,
   sys(1) = -(x(2) + x(3));
   sys(2) = x(1) + a*x(2);
   sys(3) = b + x(3)*(x(1)-c) + disturb(x,u,t);
   %%%%%%%%%%
   % Output %
   %%%%%%%%%%
case 3,
   sys = x;
   %%%%%%%%
   % End  %
   %%%%%%%%
case {2,4,9},
    sys = [];

 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end

function disturb = disturb(x,u,t) %disturb

if t>=5
    disturb=0.5*sqrt(x(2)^2 + x(1)^2 + x(3)^2)+ 0.5*(sin(4*t) + cos(2*t));
else
    disturb=0;
end
```

II.18 Code for observer corresponding to Figures 4.1, 4.3, 4.5 and 4.7

```
function [sys,x0,str,ts] = Observer(t,x,u,flag)

%Observer and its parameters
K=3;
W0=1*[1 0 0 1 0 1 0 1 0]'; %W zero
A=[0 −1 −1;1 0.398 0; 0 0 −4];
B=[0;0;1];
T=[1 1];

GAMA0=1;
GAMAW=0.5;
alfa=2*GAMA0*GAMAW;

P12 = 449/500 + alfa/2;
P11 = 4 + P12^2;
P=[P11 P12 1; P12 1 0;1 0 1];
Lsolver1 = [2*P11 0 2*P12 0 2 0 (1 + P11*alfa + 2*P12);
           P12 0 1 0 0 0 (P12*(199/500) − P11 + P12*alfa + 1);
           1 P11 0 P12 1 1 (alfa − P11 − 4);
           0 P12 0 1 0 0 (−1 − P12);
           0 2 0 0 0 2 (alfa − 10 + 1)];
Lsolver2 = rref(Lsolver1);
L11=Lsolver2(1,7);
L12=Lsolver2(2,7);
L21=Lsolver2(3,7);
L22=Lsolver2(4,7);
L31=Lsolver2(5,7);
L32=0;
L = [L11 L12; L21 L22; L31 L32];

switch flag,
   %%%%%%%%%%%%%%%%%%
   % Initialization %
   %%%%%%%%%%%%%%%%%%
case 0,

    sizes = simsizes;
    sizes.NumContStates  = 12;  %Number of Constant States
    sizes.NumDiscStates  = 0;   %Number of Discret States
    sizes.NumOutputs     = 5;   %Number of Outputs
    sizes.NumInputs      = 3;   %Number of Inputs
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
    sys = simsizes(sizes);
    x0=zeros(12,1); %
    x0(1)=0;
    x0(2)=0;
    x0(3)=0;
        str=[];
    ts=[0 0];
   %%%%%%%%%%%%%%%
   % Directives  %
   %%%%%%%%%%%%%%%
case 1,
      %Observer model
      sys = [A*x(1:3)-L*[x(1)-u(1);x(3)-u(3)]+B*K*x(4:12)'*S(x,u);
          %Learning Law
          -2*GAMAW*(GAMA0*(x(4:12)-W0) + K*T*[x(1)-u(1);x(3)-u(3)]*S(x,u))];
   %%%%%%%%%%%
   % Outputs %
   %%%%%%%%%%%
  case 3,
     sys = [x(1:3);
           norm([x(4:12)'],'fro');norm(x(1:3)-u(1:3))];
case {2,4,9},
    sys = [];

 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function S = S(x,u)      %Regressors
S=[1*(z(u(1)));
   1*(z(x(2)));
   1*(z(u(3)));
   1*(z(x(2)))*(z(u(1)));
   1*(z(x(2)))*(z(u(3)));
   1*(z(u(1)))*(z(u(3)));
   1*(z(u(1))^2);
   1*(z(x(2))^2);
   1*(z(u(3))^2)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function z = z(uu)   %Sigmoidal Function
lambda=0;
alfa=5;
beta=.5;
z=alfa/(exp(-beta*uu)+1)+lambda;
```

II.19 Code for observer corresponding to Figures 4.1, 4.3, 4.5 and 4.7

```
function [sys,x0,str,ts] = Observer(t,x,u,flag)
```

```
%Observer and its parameters
K=3;
W0=1*[1 0 0 1 0 1 0 1 0]'; %W zero
A=[0 -1 -1;1 0.398 0; 0 0 -4];
B=[0;0;1];
T=[1 1];

GAMA0=1;
GAMAW=8;
alfa=2*GAMA0*GAMAW;

P12 = 449/500 + alfa/2;
P11 = 4 + P12^2;
P=[P11 P12 1; P12 1 0;1 0 1];
Lsolver1 = [2*P11 0 2*P12 0 2 0 (1 + P11*alfa + 2*P12);
           P12 0 1 0 0 0 (P12*(199/500) - P11 + P12*alfa + 1);
           1 P11 0 P12 1 1 (alfa - P11 - 4);
           0 P12 0 1 0 0 (-1 - P12);
           0 2 0 0 0 2 (alfa - 10 + 1)];
Lsolver2 = rref(Lsolver1);
L11=Lsolver2(1,7);
L12=Lsolver2(2,7);
L21=Lsolver2(3,7);
L22=Lsolver2(4,7);
L31=Lsolver2(5,7);
L32=0;
L = [L11 L12; L21 L22; L31 L32];

switch flag,
   %%%%%%%%%%%%%%%%%%
   % Initialization %
   %%%%%%%%%%%%%%%%%%
case 0,

    sizes = simsizes;
    sizes.NumContStates  = 12;  %Number of Constant States
    sizes.NumDiscStates  = 0;   %Number of Discret States
    sizes.NumOutputs     = 5;   %Number of Outputs
    sizes.NumInputs      = 3;   %Number of Inputs
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
    sys = simsizes(sizes);
    x0=zeros(12,1); %
    x0(1)=0;
    x0(2)=0;
    x0(3)=0;
        str=[];
    ts=[0 0];
   %%%%%%%%%%%%%%%
   % Directives  %
   %%%%%%%%%%%%%%%
case 1,
      %Observer model
      sys = [A*x(1:3)−L*[x(1)−u(1);x(3)−u(3)]+B*K*x(4:12)'*S(x,u);
          %Learning Law
          −2*GAMAW*(GAMA0*(x(4:12)−W0) + K*T*[x(1)−u(1);x(3)−u(3)]*S(x,u))];
   %%%%%%%%%%%
   % Outputs %
   %%%%%%%%%%%
  case 3,
     sys = [x(1:3);
           norm([x(4:12)'],'fro');norm(x(1:3)−u(1:3))];
case {2,4,9},
    sys = [];

 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function S = S(x,u)     %Regressors
S=[1*(z(u(1)));
   1*(z(x(2)));
   1*(z(u(3)));
   1*(z(x(2)))*(z(u(1)));
   1*(z(x(2)))*(z(u(3)));
   1*(z(u(1)))*(z(u(3)));
   1*(z(u(1))^2);
   1*(z(x(2))^2);
   1*(z(u(3))^2)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function z = z(uu)  %Sigmoidal Function
lambda=0;
alfa=5;
beta=.5;
z=alfa/(exp(−beta*uu)+1)+lambda;
```

II.20 Code to plot the Figures 4.1, 4.3, 4.5 and 4.7

```
%Shown the graphs of the simulation
clc
fsize=20;

%Figure 1
fig=figure;
plot(t,x(:,1),t, Xestimated(:,1),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
```

```
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{1}(t), \hat{x}_{1}(t)$$','Interpreter','Latex','Fontsize',fsize)

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [-3 -3];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,-2.3,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,-2.7,'in action','Fontsize',fsize) % write a text on top of the arrow

set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG41.png');
close(fig)

%Figure 2
fig=figure;
plot(t,x(:,2),t, Xestimated(:,2),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [0 0];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,0.68,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,0.3,'in action','Fontsize',fsize) % write a text on top of the arrow

ylabel('$$x_{2}(t), \hat{x}_{2}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG43.png');
```

```
close(fig)

%Figure 3
fig=figure;
plot(t,x(:,3),t, Xestimated(:,3),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [-2 -2];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,-1.12,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,-1.6,'in action','Fontsize',fsize) % write a text on top of the arrow

ylabel('$$x_{3}(t), \hat{x}_{3}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG45.png');
close(fig)

%Figure 4
fig=figure;
plot(t,Xestimated(:,4),'LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('Estimated Weight Norm','Fontsize',fsize)

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [3 3];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,3.33,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,3.15,'in action','Fontsize',fsize) % write a text on top of the arrow

set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG47.png');
close(fig)
```

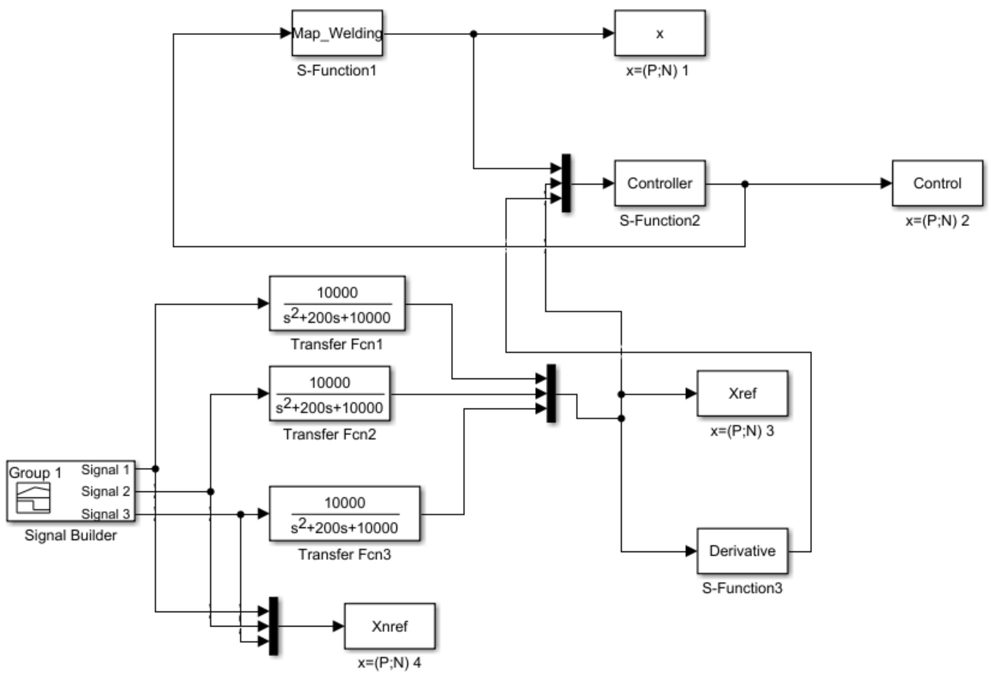
II.21 Code to plot the Figures 4.2, 4.4, 4.6 and 4.8

```
%Shown the graphs of the simulation
clc
fsize=20;

%Figure 1
fig=figure;
plot(t,x(:,1),t, Xestimated(:,1),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{1}(t), \hat{x}_{1}(t)$$','Interpreter','Latex','Fontsize',fsize)

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [-3 -3];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,-2.3,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,-2.7,'in action','Fontsize',fsize) % write a text on top of the arrow

set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG42.png');
close(fig)

%Figure 2
```

```
fig=figure;
plot(t,x(:,2),t, Xestimated(:,2),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [0 0];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,1.2,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,0.5,'in action','Fontsize',fsize) % write a text on top of the arrow

ylabel('$$x_{2}(t), \hat{x}_{2}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG44.png');
close(fig)

%Figure 3
fig=figure;
plot(t,x(:,3),t, Xestimated(:,3),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('Actual','Estimated','Location','southeast');
set(h,'FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [0 0];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;
```

```
text(5.05,0.7,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,0.3,'in action','Fontsize',fsize) % write a text on top of the arrow

ylabel('$$x_{3}(t), \hat{x}_{3}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG46.png');
close(fig)

%Figure 4
fig=figure;
plot(t,Xestimated(:,4),'LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('Estimated Weight Norm','Fontsize',fsize)

YL = get(gca, 'ylim'); %plot the vertical line
YR = YL(2) - YL(1);
YL = [YL(1) - 1000 * YR, YL(2) + 1000 * YR];
line([5, 5], YL, 'YLimInclude', 'off', 'Color','k','LineWidth',2);

pa = annotation('arrow');  % store the arrow information in pa
pa.Parent = gca;           % associate the arrow the the current axes
pa.X = [5 6.6];            % the location of arrow
pa.Y = [4 4];
pa.LineWidth  = 2;         % make the arrow bolder for the figure
pa.HeadWidth  = 20;
pa.HeadLength = 20;

text(5.05,5.2,'disturbance','Fontsize',fsize) % write a text on top of the arrow
text(5.2,4.5,'in action','Fontsize',fsize) % write a text on top of the arrow

set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG48.png');
close(fig)
```

II.22 Simulink plant used for simulations corresponding to Figures 5.2 - 5.14



II.23 Code for the welding system plant corresponding to Figures 5.2-5.14

```
[frame=single]
function [sys,x0,str,ts] = Map_Welding(t,x,u,flag)

%System extract from Anzehaee, M. M., Haeri, M. "Welding current and arc voltage ...
    control in a
%GMAW process using ARMarkov based MPC", Control Engineering Practice, Volume 19,
%Issue 12, Pages 1408—1422, 2011.

pi=3.14159265;
c1=3.3*10^—10;  %Constants
c2=0.78*10^—10;
re=0.6*10^—3;
Ls=306*10^—6;
tm=50*10^—3;
Ra=0.0237;
Rs=6.8*10^—3;
p=0.43;
V0=15.5;
Ea=400;
lc=0.025;
```

```
km=1;

switch flag,
   %%%%%%%%%%%%%%%%%%
   % Initialization %
   %%%%%%%%%%%%%%%%%%
case 0,
    sizes = simsizes;
    sizes.NumContStates = 3;  %Number of Constant States
    sizes.NumDiscStates = 0;  %Number of Discret States
    sizes.NumOutputs    = 3;  %Number of Outputs
    sizes.NumInputs     = 3;  %Number of Inputs
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
    sys = simsizes(sizes);
    x0=[0.01; 0; 0]; %Initial Conditions
    str=[];
    ts=[0 0];
   %%%%%%%%%%%%%%%
   % Directives  %   % Welding System
   %%%%%%%%%%%%%%%
case 1,
   sys(1) = x(3)-( c1*x(2)/(pi*re^2) + c2*p*x(1)*x(2)^2/(pi*re^2) )+u(1);   %% ...
       stick out
   sys(2) =(1/Ls)*(u(2)-(Ra+Rs+p*x(1))*x(2)-V0-Ea*(lc-x(1)));            %% ...
       welding current
   sys(3) =(1/tm)*(km*u(3)-x(3));                                         %% ...
       welding wire speed
   %%%%%%%%%%
   % Output %
   %%%%%%%%%%
case 3,
   sys = [x];
   %%%%%%%%%%%%%
   % Terminate %
   %%%%%%%%%%%%%
case {2,4,9},
    sys = [];

 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end
```

II.24 Code for controller corresponding to Figures 5.2 - 5.14

```
function [sys,x0,str,ts] = Controller(t,x,u,flag)

%Controller and its parameters
```

```
L = 10*[50 0 0; 0 5 0; 0 0 5];
P = 20*[50 0 0; 0 5 0; 0 0 5];
GAMAW=0.005;
GAMA0=0.005;
G=1;
W01=G*[1 0 0 0 0]';
W02=G*[0 1 0 0 0]';
W03=G*[0 0 1 0 0]';

switch flag,
   %%%%%%%%%%%%%%%%%%
   % Initialization %
   %%%%%%%%%%%%%%%%%%
case 0,

    sizes = simsizes;
    sizes.NumContStates   = 15;  %Number of Constant States
    sizes.NumDiscStates   = 0;   %Number of Discret States
    sizes.NumOutputs      = 3;   %Number of Outputs
    sizes.NumInputs       = 9;   %Number of Inputs
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
    sys = simsizes(sizes);
    x0=zeros(15,1); %Initial Conditions
    x0(1)=0;
    x0(2)=0;
    x0(3)=0;
        str=[];
    ts=[0 0];
   %%%%%%%%%%%%%%%
   % Directives  %
   %%%%%%%%%%%%%%%
case 1,
      sys = [-2*GAMAW*(GAMA0*(x(1:5)-W01) + (u(4) - u(1))*S(x,u));
             -2*GAMAW*(GAMA0*(x(6:10)-W02) + (u(5) - u(2))*S(x,u));
             -2*GAMAW*(GAMA0*(x(11:15)-W03) + (u(6) - u(3))*S(x,u))];
   %%%%%%%%%%%
   % Output  %
   %%%%%%%%%%%
  case 3,
    sys = L*[u(4)-u(1);u(5)-u(2);u(6)-u(3)] + ...
        GAMAW*GAMA0*[u(4)-u(1);u(5)-u(2);u(6)-u(3)] - ...
        P*[x(1:5)';x(6:10)';x(11:15)']*S(x,u)+u(7:9);
case {2,4,9},
    sys = [];

 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function S = S(x,u)      %Regressors
S=[1*(z(u(1)));
```

```
   1*(z(u(2)));
   1*(z(u(3)));
   1*(z(u(1)))*(z(u(2)));
   1*(z(u(1)))*(z(u(2)))^2];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function z = z(uu)  %Sigmoidal Function
lamda=0;
alfa=10;
beta=.5;
z=alfa/(exp(-beta*uu)+1)+lamda;
```

II.25 Code for derivative

```
function [sys,x0,str,ts] = Derivative(t,x,u,flag)

Wn = 100;
eta = 1;

switch flag,
   %%%%%%%%%%%%%%%%%%
   % Initialization %
   %%%%%%%%%%%%%%%%%%
case 0,

    sizes = simsizes;
    sizes.NumContStates  = 6;   %Number of Constant States
    sizes.NumDiscStates  = 0;   %Number of Discret States
    sizes.NumOutputs     = 3;   %Number of Outputs
    sizes.NumInputs      = 3;   %Number of Inputs
    sizes.DirFeedthrough = 1;
    sizes.NumSampleTimes = 1;
    sys = simsizes(sizes);
    x0=zeros(6,1); %Initial Conditions
        str=[];
    ts=[0 0];
   %%%%%%%%%%%%%%%
   % Directives  %
   %%%%%%%%%%%%%%%
case 1,
      sys = [x(2);
             -x(1)*Wn^2 - 2*x(2)*eta*Wn + u(1)*Wn^2;
             x(4);
             -x(3)*Wn^2 - 2*x(4)*eta*Wn + u(2)*Wn^2;
             x(6);
             -x(5)*Wn^2 - 2*x(6)*eta*Wn + u(3)*Wn^2;];
   %%%%%%%%%%%
   % Output  %
   %%%%%%%%%%%
```

```
  case 3,
    sys = [x(2); x(4); x(6)];
case {2,4,9},
    sys = [];

 otherwise
   error(['unhandled flag = ',num2str(flag)]);
end
```

II.26 Code to plot the Figures 5.2 - 5.14

```
%Shown the graphs of the simulation
clc
fsize=20;

%Figure 1
fig=figure;
plot(t,Xnref(:,1),t,Xref(:,1),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('$x_{nref_{1}}(t)$', '$x_{ref_{1}}(t)$','Location','southeast');
set(h,'Interpreter','Latex','FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
ylim([0 0.025]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{nref_{1}}(t), x_{ref_{1}}(t)$$','Interpreter','Latex','Fontsize',fsize)
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG51.png');
close(fig)

%Figure 2
fig=figure;
plot(t,Xnref(:,2),t,Xref(:,2),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('$x_{nref_{2}}(t)$', '$x_{ref_{2}}(t)$','Location','southeast');
set(h,'Interpreter','Latex','FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
ylim([0 450]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{nref_{2}}(t), x_{ref_{2}}(t)$$','Interpreter','Latex','Fontsize',fsize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG52.png');
close(fig)

%Figure 3
fig=figure;
plot(t,Xnref(:,3),t,Xref(:,3),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
```

```
grid on
grid minor
h=legend('$x_{nref_{3}}(t)$', '$x_{ref_{3}}(t)$','Location','southeast');
set(h,'Interpreter','Latex','FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{nref_{3}}(t), x_{ref_{3}}(t)$$','Interpreter','Latex', ...
    'Fontsize',fsize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG53.png');
close(fig)

%Figure 4
fig=figure;
plot(t,Xref(:,1),t,x(:,1),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('$x_{ref_{1}}(t)$','$x_{1}(t)$','Location','southeast');
set(h,'Interpreter','Latex','FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{ref_{1}}(t), x_{1}(t)$$','Interpreter','Latex','Fontsize',fsize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG54.png');
close(fig)

%Figure 5
fig=figure;
plot(t,Xref(:,2),t,x(:,2),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('$x_{ref_{2}}(t)$','$x_{2}(t)$','Location','southeast');
set(h,'Interpreter','Latex','FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{ref_{2}}(t), x_{2}(t)$$','Interpreter','Latex','Fontsize',fsize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG55.png');
close(fig)

%Figure 6
fig=figure;
plot(t,Xref(:,3),t,x(:,3),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('$x_{ref_{3}}(t)$','$x_{3}(t)$','Location','southeast');
set(h,'Interpreter','Latex','FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{ref_{3}}(t), x_{3}(t)$$','Interpreter','Latex','Fontsize',fsize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG56.png');
```

```
close(fig)

%Figure 7
fig=figure;
semilogx(t,Xref(:,1),t,x(:,1),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('$x_{ref_{1}}(t)$','$x_{1}(t)$','Location','southeast');
set(h,'Interpreter','Latex','FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlim([0.001 8]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{ref_{1}}(t), x_{1}(t)$$','Interpreter','Latex','Fontsize',fsize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG57.png');
close(fig)

%Figure 8
fig=figure;
semilogx(t,Xref(:,2),t,x(:,2),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('$x_{ref_{2}}(t)$','$x_{2}(t)$','Location','southeast');
set(h,'Interpreter','Latex','FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlim([0.001 8]);
ylim([-50 450]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{ref_{2}}(t), x_{2}(t)$$','Interpreter','Latex','Fontsize',fsize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG58.png');
close(fig)

%Figure 9
fig=figure;
semilogx(t,Xref(:,3),t,x(:,3),':','LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('$x_{ref_{3}}(t)$','$x_{3}(t)$','Location','southeast');
set(h,'Interpreter','Latex','FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlim([0.001 8]);
ylim([-0.02 0.18]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('$$x_{ref_{3}}(t), x_{3}(t)$$','Interpreter','Latex','Fontsize',fsize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG59.png');
close(fig)

%Figure 10
fig=figure;
plot(t,Control(:,1),'LineWidth',2);set(0,'DefaultAxesFontSize',16);
```

```
grid on
grid minor
h=legend('$u_{1}(t)$','Location','southeast');
set(h,'Interpreter','Latex','FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('Control Input 1','Fontsize',fsize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG510.png');
close(fig)

%Figure 11
fig=figure;
plot(t,Control(:,1),'LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('$u_{1}(t)$','Location','northeast');
set(h,'Interpreter','Latex','FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
ylabel('Control Input 1','Fontsize',fsize);
xlim([0.5 8]);
ylim([-0.25 0.2]);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG511.png');
close(fig)

%Figure 12
fig=figure;
plot(t,Control(:,2),'LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('$u_{2}(t)$','Location','southeast');
set(h,'Interpreter','Latex','FontSize',fsize);
set(0,'DefaultAxesFontSize', 16);
ylim([0 35]);
xlabel('Time (s)','Fontsize',fsize);
ylabel('Control Input 2','Fontsize',fsize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG512.png');
close(fig)

%Figure 13
fig=figure;
plot(t,Control(:,3),'LineWidth',2);set(0,'DefaultAxesFontSize',16);
grid on
grid minor
h=legend('$u_{3}(t)$','Location','southeast');
set(h,'Interpreter','Latex','FontSize',fsize);
ylim([0 0.22]);
set(0,'DefaultAxesFontSize', 16);
xlabel('Time (s)','Fontsize',fsize);
```

```
ylabel('Control Input 3','Fontsize',fsize);
set(gcf,'units','normalized','outerposition',[0 0 1 1]);
saveas(gcf,'FIG513.png');
close(fig)
```