Universität der Bundeswehr München
Fakultät für Luft- und Raumfahrttechnik
Institut für Steuer- und Regelungstechnik

Robust Controller Optimization: Application to a Parallel Hybrid Electric Vehicle (PHEV)

Dipl. -Ing. Nabil Lachhab

Vollständiger Abdruck der bei der Fakultät für Luft- und Raumfahrttechnik der Universität der Bundeswehr München zur Erlangung des akademischen Grades eines

Doktor-Ingenieur (Dr.-Ing.)

genehmigten Dissertation.

Gutachter/Gutachterin:

1. Univ. Prof. Dr.-Ing. habil. Ferdinand Svaricek
2. Univ. Prof. Dr.rer.nat. Claus Hillermeier

Die Dissertation wurde am 06.11.2015 bei der Universität der Bundeswehr München eingereicht und durch die Fakultät für Luft- und Raumfahrttechnik am 13.05.2016 angenommen. Die mündliche Prüfung fand am 01.06.2016 statt.

Contents

Nomenclature

Abbreviations

ANN	Artificial Neural Network
$FOPID$	Fractional order PID controller
HEV	Hybrid Electrical Vehicle
IPM	Interior Point Methods
MPC	Model Predictive Control
NN	Neural Network
RNN	Recurrent Neural Network
$PHEV$	Parallel Hybrid Electrical Vehicle
RCP	Rapid Control Prototyping

General Notation

$\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}^T$	System matrices
$\mathbf{A}_0$, $\mathbf{B}_0$, $\mathbf{C}_0^T$	Nominal system matrices
S	Sensitivity transfer function
T	Complementary sensitivity transfer function
T_0	Nominal complementary sensitivity transfer function
T_d	Desired complementary sensitivity transfer function
L_d	Desired open-loop response
G	Uncertain LTI plant
G_0	Nominal plant

FOPID-H_∞-controllers

W_N	Noise weighting transfer function
W_S	Sensitivity weighting transfer function
T^A	Approximated complementary sensitivity transfer function
T_0^A	Approximated nominal complementary sensitivity transfer function
L	Open-loop transfer function
L_d^A	Approximated desired open-loop response
Φ_m	Phase margin
$\Delta\phi_m$	Variation of the phase margin
ω_c	Crossover frequency
Φ_m^d	Desired phase margin
ω_c^d	Desired crossover frequency
α	fractional order for the $PI^\alpha D^\beta$ controller

β	fractional order for the $PI^{\alpha}D^{\beta}$ controller
n	fractional order for the $(PID)^n$ controller
v	fractional order for the $(I)^v$ controller
K_S	Static controller
N	Noise signal
N_W	Weighted noise signal
E	Error signal
E_W	Weighted error signal
Y	Output signal
R	Reference signal
G	Uncertain LTI plant
G_0	Nominal plant
G_P	Augmented LTI plant
V	Uncertain parameter
V_{min}, V_0, V_{max}	Minimal, nominal and maximal value of the uncertain parameter V

Optimization of PID Controllers using MPC and LMIs

T_{min}	Closed-loop transfer function for V_{min}
T_{max}	Closed-loop transfer function for V_{max}
τ_{min}	Time constant of the step response of $T_{min}(s)$
τ_{max}	Time constant of the step response of $T_{max}(s)$
ω_b	Closed-loop bandwidth
r	Reference signal
y	Output signal
u	Input signal
N	Prediction horizing
$\mathbf{u}$	Input vector
$\mathbf{y}$	Output vector
$\mathbf{y}_d$	Desired output vector
$\mathbf{\Omega}$	MPC transition matrix
$\mathbf{E}_u$	Constraint input Matrix
$\mathbf{E}_y$	Constraint input Matrix
$\mathbf{f}_u$	Constraint input Matrix
$\mathbf{f}_y$	Constraint input Matrix
$\tilde{\mathbf{U}}$	Input constraint set
$\tilde{\mathbf{Y}}$	Output constraint set
U_{max}	Maximal input value
Δy_{max}	Maximal output difference value

Δy_{min}	Minimal output difference value
x_d, u_d, y_d	Desired model sate, input and output
a_d, b_d, c_d	Desired model parameters
u_{opt}	Optimal input trajectory
F_I	Discrete-time integrator
F_D	Discrete-time Derivative
$\mathbf{r}$	Reference vector
$\mathbf{e}$	Error vector
$\mathbf{\Omega}_I$	MPC integral transition matrix
$\mathbf{\Omega}_D$	MPC derivative transition matrix
$\mathbf{e}_I$	Integral error vector
$\mathbf{e}_D$	Derivative error vector

Artificial Neural Network Controller

w_i	Neural network weight
$f_1(\cdot)$ to $f_n(\cdot)$	Neural network hidden layer activation functions
$h_1(\cdot)$ to $h_m(\cdot)$	Neural network output layer activation functions
φ_1 to φ_r	Neural network inputs
z_1 to z_q	Neural network outputs
b_{I1} to b_{In}	Neural network input bias
b_{H1} to b_{Hm}	Neural network output bias
w_{lj}	Neural network input weights
v_{ji}	Neural network output weights
$\mathbf{f}(\cdot)$	Neural network field input layer function
$\mathbf{h}(\cdot)$	Neural network field output layer function
$\mathbf{W}$	Neural network input weight matrix
$\mathbf{V}$	Neural network output weight matrix
$\mathbf{z}$	Neural network output vector
$\mathbf{b}_I$	Neural network input bias vector
$\mathbf{b}_H$	Neural network output bias vector
$g(\cdot)$	Nonlinear discrete-time function
$z_{fi}(\cdot)$	Neural network internal states
w_{fij}	Neural network internal states weights
$\mathbf{W}_f$	Neural network states weight matrix
$\mathbf{A}_W$, $\mathbf{B}_W$ and $\mathbf{C}_W^T$	Neural network state-space matrices
$\mathbf{A}_{WP}$, $\mathbf{B}_{WP}$ and $\mathbf{C}_{WP}^T$	Neural network plant state-space matrices
$\mathbf{A}_{WC}$, $\mathbf{B}_{WC}$ and $\mathbf{C}_{WC}^T$	Neural network controller state-space matrices
$\mathbf{f}_P(\cdot)$	Neural network plant field input layer function
$\mathbf{h}_P(\cdot)$	Neural network plant field output layer function
$\mathbf{f}_C(\cdot)$	Neural network controller field input layer function

$\mathbf{h}_C(\cdot)$	Neural network controller field output layer function
$\mathbf{z}_{fP}$	Neural network plant states vector
$\mathbf{z}_{fC}$	Neural network controller states vector
$\mathbf{W}_C$	Neural network controller state-space matrix
$\mathbf{x}_K$	Controller states
$\mathbf{A}_K$, $\mathbf{b}_K$, $\mathbf{c}_K^T$, $\mathbf{d}_K$	Controller state-space matrices
$\mathbf{x}_P$	Plant states

Parallel Hybrid Electrical Vehicle

$\dot{\varphi}_{ICE}$	Angular velocity of the ICE [rpm]
$\dot{\varphi}_{EM}$	Angular velocity of the EM [rpm]
$\dot{\varphi}_{EM,Des}$	Desired angular velocity of the EM [rpm]
$M_{EM,Des}$	Desired EM torque [Nm]
$M_{EM,Act}$	Actual EM torque [Nm]
$M_{CL,Des}$	Desired clutch torque [Nm]
$M_{CL,Act}$	Actual clutch torque [Nm]
G_{ICE}	Transfer function $M_{CL,Act}$ to $\dot{\varphi}_{ICE}$
G_{EM}	Transfer function $M_{EM,Des}$ to $M_{EM,Act}$
G_{ROT}	Transfer function $M_{EM,Act}$ to $\dot{\varphi}_{EM}$
G_{CL}	Transfer function $M_{CL,Des}$ to $M_{CL,Act}$
$G_{CL,0}$	Nominal transfer function $M_{CL,Des}$ to $M_{CL,Act}$
PD_{f1}	Discrete-time PD controller with filtered derivative
PD_{f2}	Discrete-time PD controller with filtered derivative
$L_{CL}(z)$	Discrete-time clutch open-loop function
$L_{CL,0}(z)$	Nominal discrete-time clutch open-loop function
$T_{CL}(z)$	Discrete-time clutch closed-loop function
$\tilde{\mathbf{M}}_{Des}$	MPC desired torque input vector
$\mathbf{n}_{EM}$	MPC EM angular velocity vector
$\mathbf{n}_{ICE}$	MPC ICE angular velocity vector
$\mathbf{\Omega}_{ICE}$	MPC ICE transition matrix
$\mathbf{\Omega}_{EM}$	MPC EM transition matrix
$\tilde{\mathbf{M}}_{ICE,Des}$	MPC desired torque input vector (controller output ICE)
$\tilde{\mathbf{M}}_{EM,Des}$	MPC desired torque input vector (controller output EM)

Acknowledgments

This thesis is the conclusion of my work at the Institute of Control Engineering, Department of Aerospace Engineering at the University of the Federal Armed Forces under the supervision of Prof. Dr.-Ing. Ferdinand Svaricek. It was a coorperation between the university and the automotive engineering company IAV. This thesis was one of the most challenging task, which was requiring a lot of patience, determination, diligence and not least the support many people.

I am greatly thankful to Prof. Dr.-Ing. Ferdinand Svaricek who gave me the possibility to work at the Institute of Control Engineering. Moreover, i want to thank him for his support during my whole stay at the University of the Federal Armed Forces. I want also to thank the company IAV, which was willing for the financial support of my work. Especially, i am greatful to Heiko Rabba, Dr.-Ing Frank Wobbe, Peter Michau, Tomas Binnewies and Mathias Schultalbert. I want also to thank Prof. Dr. rer. nat. habil. Claus Hillermeier for willing to review my work and the chair of the doctoral committee. I am also deeply grateful to Dr. rer. nat. José-Luis Marqués-López for his review of the first draft of this work and for his support during my work.

I want to thank all my collegues at the Institute of Control Engineering Christoph Hartung, Hermann Lex, Dr.Ing Gunther Reissig, Elisabeth Loessl. Special thanks to my room colleague Tobias Fueger for his support and guidance, with whom i had a very enjoyable time.

Finally, i am very thankful to my parents Miloudi and Amina Lachhab for their love and support during my whole academic journey. I want also to thank my brothers Nourdinne, Karim and Abdellatif and my sister Amal. Last but not least, i am deeply thankful to Snjezana Plön for her love and support during this challeging phase of my life. Without her encouregement and motivation, it would not be possible to succeed this work.

Abstract

This thesis deals with the design and optimization of robust low-order/fixed-structure controllers. Thereby, two classes are considered, namely integer and fractional order controllers. Three approaches are proposed to tune the parameters of these controllers. Moreover, the obtained controllers are validated in simulation as well as in real environment.

In what concerns fractional order controllers, a method is proposed to optimize the parameters of $PI^{\alpha}D^{\beta}$ and $(PID)^{n}$ controllers. Thereby, robustness specifications have to be achieved. These are expressed in terms of a desired phase margin, desired crossover frequency and robust performance in case of static gain variations. For this purpose, the H_{∞} norm is used to formulate the control problem. The resulting optimization problem is solved iteratively using our proposed method. Moreover, a Matlab Toolbox has been developed dedicated to the controller optimization, namely Fractional Order FOPID Controller (FOPID)-Toolbox.

The second method is dedicated to the optimization of robust PID controllers. Based on Model Predictive Control (MPC), the optimization problem is formulated. The resulting problem is transformed into a matrix inequality problem. Specifically, it is a Bilinear Matrix Inequality (BMI) problem. Using a proposed method, this problem is transformed into a Linear Matrix Inequality (LMI), which is solved in the controller parameters using well known LMI solvers. Simulation examples are presented to show the effectiveness of this approach.

The third method is concerned with the optimization of linear controllers based on Recurrent Neural Networks (RNN). Using a novel procedure, the control problem can be formulated as a closed-loop RNN. The plant as well as the controller are structured as a recurrent network. The weights of the network which represent the plant are set fixed. The controller network weights are free parameters, which will be updated during the training stage. Thereby, the goal is to provide the robustness requirements given in terms of the closed-loop step response. Simulation results are also presented to show the effectiveness of this approach.

To test the approaches proposed in this work, a test Parallel Hybrid Electrical Vehicle (PHEV) is considered. Based on measured data, a MIMO model is first identified. Then, this model is used to compute the robust controller parameters. After testing the two model-based methods in simulation, the neural network linear controller provides the best performance. This controller is implemented on the real plant using rapid prototyping methods. Experimental results show the achieved performance with this controller.

Kurzfassung

Der Entwurf und die Optimierung robuster Regelung ist eine wichtige Aufgabestellung im Bereich der Regelungstechnik. Dabei ist das Ziel die Reglerparameter so zu optimieren, dass der geschlossene Regelkreis bei einer Änderung der Streckenparameter robust ist. In dieser Arbeit werden drei verschiedene Verfahren zur Regleroptimierung vorgestellt und untersucht.

Die erste Methode behandelt die Optimierung der fraktionalen $PI^{\alpha}D^{\beta}$ und $(PID)^{n}$ Reglern. Das Ziel ist die Optimierung der Reglerparameter, sodass die gewünschte Robustheit erreicht werden kann. Die vorgegebene Dämpfung des geschlossenen Regelkreises bleibt bei einer Änderung der Streckenverstärkung konstant. Das Verfahren beruht auf dem kürzlich entwickelten H_{∞}-Verfahren zur Optimierung von Reglern mit vorgegebener Struktur. Dieses Verfahren ist inzwischen in der Matlab Toolbox "Robust Control" enthalten.

Das zweite Verfahren betrachtet die Optimierung von klassischen PID-Reglern mit Hilfe von linearen Matrizenungleichungen und modellbasierter prädiktiver Regelung. Auch bei dieser Methode muss die Robustheit bei einer Variation der Streckenverstärkung erzielt werden. Zuerst wird das Optimierungsproblem durch das Konzept der Prädiktion formuliert. Das resultierende Problem ist nichtkonvex und stellt eine Herausforderung dar. Durch eine geschickte Transformation der resultierenden nichtlinearen Ungleichungen wird das Optimierungsproblem in ein konvexes Problem *Linear Matrix Inequality (LMI)* überführt. Anschliessend wird das LMI-Problem mit Hilfe der Yalmip-Toolbox aufgestellt und mittels der SeDuMi-Toolbox gelöst.

Als drittes Verfahren wird die Optimierung neuronaler Netze betrachtet. Zwei neuronale Netze werden zusammengeführt um die Dynamik des geschlossenen Regelkreises darzustellen. Ein neuronales Netz beschreibt den gesuchten Regler mit einer vordefinierten Reglerstruktur. Die Strecke, dessen Verstärkung mit einer Unsichertheit behaftet ist, wird durch ein zweites Netz abgebildet. Die Gewichtungen dieses Netz sind konstant und stellen die Streckenparameter dar. Mit Hilfe eines Optimierungsverfahren, das Bestandteil der Neural Network Matlab Toolbox ist, werden die Reglerparameter so optimiert, dass bei einer Variation der Streckenverstärkung die Sprungantwort des geschlossenen Regelkreises keinen Überschwinger aufweist.

Um die Regelgüte der verschiedenen Methoden beurteilen zu können, werden sowohl Simulation- als auch Versuchsergebnisse präsentiert. Dabei wird das Model des Antriebsstrangs eines Parallelhybridfahrzeugs betrachtet. Basierend auf einer linearen Identifikationsmethode wird die Dynamik der Strecke approximiert. In Simulation zeigen neuronale Netze die besten Ergebnisse. Die geforderte Robustheit ist erfüllt. Aus diesem Grund wird der neuronale Netz-Regler mit Hilfe des Prototyping- und Schnittstellenmoduls ES910 der Firma ETAS am Testfahrzeug implementiert und getestet. Die experimentellen Ergebnisse zeigen, dass die Robustheit auch in der Praxis erhalten bleibt.

1 Introduction

1.1 State of the Art (PHEV)

Nowadays, many automotive manufacturers are working on the possibility of replacing the Internal Combustion Engine (ICE) by alternative engines. The strict regulation in spite of emission reduction and the increasing price of oil has been motivating engineers to replace this kind of energy. Electrical Machines (EM) offer an alternative to conventional combustion engines, which produce zero emission. Due to the low kilometer range of the battery, the complete replacement of the ICE by the EM is not always possible. Hybrid Electrical Vehicles (HEV) constitute a midterm solution to this problem. The key idea is to partially replace the ICE instead of a fully replacement. This can be realized by combining the combustion engine with an electrical machine. The vehicle is driven electrically and the ICE starts, when necessary.

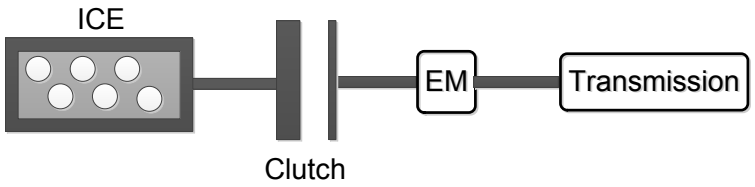

Figure 1.1: Simplified representation of the drivetrain configuration of a PHEV

Figure 1.1 shows a representation of a Parallel Hybrid Electrical Vehicle (PHEV). In this configuration, the EM and the ICE are both placed on the same shaft. A separation clutch in between decouples the ICE from the EM. Using this clutch, it is possible to drive the vehicle pure electrically. In this latter case, the separation clutch is open. The driver desired torque is provided by the EM. Moreover, in case that this torque is higher than the maximal torque of the EM, the clutch has to be closed and the ICE has to be started. In this case, the EM is working as a generator to charge the battery.

Another drive scenario can be achieved by the configuration shown in Figure 1.1 namely, the boosting. In this case, the separation clutch is closed. The total available torque is the sum of the EM torque and the ICE torque. Generally, depending on the drive scenario, it is required to decouple the EM from the ICE by opening the clutch or to couple both sources of torque by closing the separation clutch.

Due to its direct impact on the driver comfort, this work considers the engagement of the separation clutch. This has to be achieved without any deterioration of the driver comfort. Moreover, the angular velocity of the ICE has to be synchronized with the angular velocity of the EM. This has to occur without causing any oscillation in the

drivetrain or a noticeable jerk. In the literature, this problem have been extensively discussed. Mostly, the proposed approaches have been validated only through simulation. Below, the well known contributions in this field are briefly summarized.

For example, a PHEV called P12 is discussed in (Beck et al. 2005) and (Beck et al. 2007). This configuration consists of one conventional combustion engine and two electrical machines. The first EM is placed between the clutch and the ICE. It is used as a starter for the ICE. The second EM is used to drive the vehicle electrically. The authors propose a method based on MPC to synchronize the angular velocity of the ICE with the angular velocity of the EM. The controller is designed to achieve a fast engagement of the clutch without abuse in the driver comfort. Actually, the authors consider the case that the ICE already rotates at idle speed. The task of the MPC controller is to bring the velocity of the ICE from the idle speed up to the actual speed of the EM. Simulation results are presented to conclude the work. In (Beck et al. 2005), robustness in the presence of uncertainty in the mass of the vehicle is considered.

The application of an hybrid optimal strategy to control a dry clutch for an Automated Manual Transmission (AMT) is considered in (Van Der Heijden et al. 2007). Based on a non-linear friction model of the clutch, the authors design an explicit MPC controller and a Piecewise Linear Quadratic (PWLQ) controller. Moreover, due to the difficulties related to the implementation of MPC controllers, the authors use the explicit MPC. It consists of solving the MPC optimization problem parametrically, which results in a large set of explicit piecewise affine control laws. Simulation results show that, although the controller provides a fast engagement of the clutch, the robustness is not achieved. Moreover, the authors mention that variations in the model and disturbances lead to infeasibility of the controller. To compare the performance of the explicit MPC, the author design a PWQC. The related simulation results are more promising.

In (Gasper et al. 2009) and (Gasper et al. 2011), flatness based control is used for the launch clutch of a PHEV with an AMT. In (Dolcini et al. 2005), an observer-based controller method is presented to control the clutch engagement in an AMT. Moreover, the controller is based on solving an optimal problem. Uncertainty in the clutch model regarding the friction parameter is taking into consideration. Using a Linear Time Varying (LTV) observer, this parameter is estimated. Simulation results are presented to conclude this work.

In (Jarczyk et al. 2009), (Alt et al. 2010) and (Alt et al. 2012) a strategy is presented to synchronize the angular velocity of the ICE with the angular velocity of the EM in a PHEV. The structure of the strategy is based on a decoupling network and two optimized controllers. The controller design is based on a linearization of the nonlinear model of the plant. Feedforward strategies are explored to ameliorate the overall performance. Nonlinear simulation results are shown to demonstrate the performance and robustness of the controller. The application of flatness based control is also considered.

In (Amari et al. 2009) and (Amari et al. 2008), the design of a hybrid MPC controller

for the clutch engagement in an automated manual transmission is presented. Moreover, experimental results and a comparison with a PI based control are given. The authors show that the MPC based approach provides a better performance. By means of the performance with respect to the start-up of the vehicle, the response of the MPC controller is faster then using a PI controller.

Besides the publications cited above, there exists some patents dedicated to the synchronization problem in PHEV using the separation clutch and to the clutch engagement in automated manual transmission. Many automobile manufacturers have been working on solving the synchronization problem from different point of view. In (Schnitzer et al. 2013), a method to synchronize the angular velocity of the ICE with the angular velocity of the EM is proposed. First, the separation clutch is partially closed to pull up the ICE up to a predefined start velocity (about 300 rpm). Then, the clutch is reopened again. At this velocity, the ignition can be occurred and the ICE can be started. This means that the ICE can now provide itself a torque. Taking the actual velocity of the EM as a target velocity for the ICE, a velocity controller can be used to reach this velocity fast and without overshoot. When both velocities are approximately equal, the separation clutch can be closed. In (Motosugi et al. 2007), a PHEV with two clutches is considered. The first clutch is placed between the ICE and the EM. The second clutch is placed between the EM the automatic transmission aimed to replace the standard torque converter. The goal is to use the second clutch to control the transmission of the driving torque from both energy sources to the wheels. This task is achieved based on a speed controller of the input and output clutch angular velocities. Internal, a target speed based on the driver operations and drive conditions is generated. In (Shimabukuro et al. 2003), the control of start-stop operations in hybrid vehicles is considered. In this case, saving fuel consumption is achieved when the vehicle is temporary stopped and the engine is not running. Thus, the electric machine plays the role of a starter. In (Deguchi et al. 2000), another electric hybrid vehicle structure is discussed. It consists of two electrical machines, one is directly connected to the ICE and the other EM is placed on the main shaft separated from the ICE using the clutch. The engagement and disengagement of the separation clutch allows a pure electrical as well as a hybrid drive. The main advantage of this configuration is that the first EM can be used to pull the ICE speed up to the EM speed without any deterioration of the driver comfort as this is achieved with an opened clutch. When both velocities are equal, the clutch can be closed.

The main difficulty in using the separation clutch in hybrid electrical vehicle is that its characteristic depends on the operating conditions in terms of temperature and ageing effects. The dynamic of the clutch is not constant and should be taken in consideration. In (Eisenwerth et al. 2013), the adaptation of the characteristic of the clutch torque is considered. Due to the use of a feedforward strategy to synchronize the ICE and EM velocities using the separation clutch, an exact knowledge of the transmitted clutch torque is mandatory. In the patent cited above, the authors propose a continuously adaptation using a so-called phantom start.

Another realization of the synchronization task is to use the separation clutch to pull the angular velocity of the ICE up to the actual angular velocity of the EM. This can be realized by adjusting the position of the clutch such that the ICE velocity approaches the EM velocity. When both angular velocities are equal, the clutch is completely closed. Under the assumption that the dynamic of the clutch is well known, the synchronization of both angular velocities can be performed using a feedforward strategy.

Using the separation clutch to pull up the ICE during the synchronization phase is based on the knowledge of the torque position characteristic curve. Due to the absence of a sensor to measure the transmitted torque, this curve has to be permanently adapted. This fact requires a high application effort. One of the main contribution of this work is the design of an overall robust control strategy to solve the synchronization task in PHEVs as well as the experimental validation on a test vehicle. Low and structured controllers (PID, Lead-Lag, etc.) are used for this task. The main benefit of such controllers is that the implementation does not require a high computing power. In (Apkarian and Noll 2006) and (Gahinet and Apkarian 2011), a method is proposed to optimize this class of controllers. It is based on nonsmooth optimization techniques to solve the H_∞ problem under structural constraints. In (Khatibi et al. 2008), an approach is presented to design low order controllers. The authors used a finite set of complex values to represent the system dynamic and then solved the related optimization problem using convex optimization. Genetic algorithms are used in (Farag and Werner 2006) and (Popov et al. 2005) to optimize low order controllers.

In the scope of designing a controller for the synchronization task in a PHEV, two novel approaches to optimize structured controllers are developed in this work. The first method is based on MPC. The control optimization problem is first formulated and then transformed into a convex optimization problem. The controller parameters are decision variables and can be efficiently optimized. Afterwards, the optimization problem is solved offline using the LMI solver SEDUMI (Sturm 1999) in combination with the modeling Toolbox YALMIP (Lofberg 2004).

The second method uses Neural Network (NN) strucutres to represent the closed-loop system. After defining the objective function, the closed-loop network is trained to optimize the controller weights. NNs have been mainly presented in the literature to identify the dynamic of linear and nonlinear plants. Several types of NNs have been proposed, see (Abbas and Werner 2008) and (Gil et al. 2006a). There exists two kinds of neural networks, feedback or recurrent and feedforward networks. Recurrent Neural Networks (RNN) have been successfully used in (Lachhab et al. 2008) to identify nonlinear plants operating in closed-loop. It is based on building a structured RNN, which represents the closed-loop system. The controller parameters are fixed known weights and the required model parameters are the unknown weights of the RNN. This idea is used in this work to solve the structured controller problem. Unlike in (Lachhab et al. 2008), the plant model was already identified separately and therefore known. The controller parameters are the unknown weights which have to be optimized. The overall RNN consists of two

networks. A network with known weights, which represents the plant. Another network with unknown weights, which represents the controller. The controller parameters are adapted during the training of the RNN. The adaption or the optimization of the RNN weights can be performed using well known algorithms as Steepest-Descent, Newton or Levenberg-Marquadt algorithms.

1.2 Contributions of the Work

The main objective of this work is to explore new methods and approaches to design robust low- and structured order controllers. Thereby, integer and fractional order controllers are considered. Moreover, an application to automotive control problems is presented. Due to its today importance and the role it plays, electric vehicles constitute an attractive plant to be studied. Thanks to the corporation with the automotive engineering company IAV, it was possible to implement and test the controller on a hybrid vehicle. During the work, four main approaches were developed. The main contributions of this work are summarized as follows

- **Fractional controllers $(PID)^n$ and $PI^\alpha D^\beta$.** It deals with the design and optimization of robust fractional order controllers based on a frequency domain approach. Specifically, we propose a systematic method based on the recently developed nonsmooth techniques (Apkarian and Noll 2006) to optimize the parameters of the fractional controllers $(PID)^n$ and $PI^\alpha D^\beta$.

  In the fractional control community, a very prominent work in the field of robust control design for uncertain systems with static gain variations is given by the fractional CRONE Toolbox developed by the CRONE Group with the first generation presented in (Oustaloup et al. 1993). However, the considered controller turns out to be an approximation of the desired open-loop response divided by the nominal transfer function of the plant. Its structure is given through a transfer function in polynomial form with a given order n. The control problem consists in identifying the coefficients of this function. Contrary to the CRONE Group, the fractional controller in this work has a fixed structure which consists of $(PID)^n$ and $PI^\alpha D^\beta$. We are optimizing the classical PID parameters K_P, K_I and K_D and the fractional orders α, β and n. Robustness requirements are taken into consideration by specifying a desired open-loop frequency response. This is achieved through a desired phase margin, desired crossover frequency and a flat phase around the crossover frequency.

  To make the design and tuning of fractional controllers easy, we implemented the whole algorithm and provided it as a Matlab Fractional Order PID Toolbox (FOPID). The user specifies to our FOPID-Toolbox the considered uncertain plant, the desired phase margin and crossover frequency. Afterwards, the optimization is started and the optimal parameters are computed iteratively. One of the main

benefits of using our FOPID-Toolbox is that time-delay systems are also considered, which is achieved through the approximation of a time-delay fractional open-loop function. Simulations results are presented, which show the effectiveness of our approach. Moreover, the comparison to other methods show that the controller computed using our Toolbox outperform all other controllers.

- **Optimization of PID controllers based on MPC and LMIs.** Here, we considered the design and optimization of PID controllers using two modern methods, namely Model Predictive Control (MPC) and Linear Matrix Inequalities (LMIs).

  The design of PID controllers has been extensively considered in the literature. For example, in case that the order of the plant is low and the requirements, which have to be satisfied are not too severe, heuristic methods can be used. A very prominent method is the Ziegler and Nichols tuning method. In case that a desired open-loop or closed-loop response is provided, heuristic methods fail. This is also the case for constraints on the input, output or states. Modern algorithms are required to solve such problems.

  In the scope of this work, we developed a method to optimize robust PID controllers under specified signal constraints. Our proposed approach is based on MPC to formulate the related control problem including robustness requirements. Standard approaches, which deal with MPC problems consists of solving iteratively the optimization problem at each sampling time. This is a hard constraint imposed on the hardware as the convergence has to be achieved within the specified sampling time. For this reason, MPC has been reserved to system with slow dynamics.

  Generally, it exists two main approaches to solve MPC. The first approach consists of solving a simplified version of the optimization problem in real-time, see (Wang and Boyd 2008). The second approach is based on the implementation of look-up tables after solving the MPC problem offline, (Bemporad et al. 2002). In this work, the second method is considered but without the need to use look-up tables. This has been achieved by approximating the optimal MPC trajectory using a fixed structure controller, namely PID. The resulting optimization turns out to be nonlinear, which can not be solved efficiently without further manipulation. For this reason, we developed a technique to formulate this optimization as a convex problem. Thereby, linear matrix inequalities are used. Due to the availability of modern solvers, which can be embedded in Matlab LMIs can be solved efficiently. Combining MPC with LMIs, we come out with a method to optimize robust PID controllers under input, output or states constraints. Moreover, our proposed approach is applied on SISO as well as on MIMO systems.

- **Optimization of low order linear controllers based on NN.** Here, a novel approach is presented to optimize linear robust controllers using recurrent neural

networks. We have shown that these networks are not only a powerful mathematical tool to solve nonlinear problems but can also be used to formulate practical control problems.

The main idea of this work is to represent closed-loop control problems as a two neural networks structure. The first network is dedicated to the plant, which can be linear as well as nonlinear. The second network represents the controller and can be structured accordingly. In the literature, this idea was first considered in (Lachhab et al. 2008). It was successfully applied to identify an unstable plant operating in closed-loop with a known controller. In this work, we extend this approach to controller design problems. It means that we are also training a neural network structure, but this time the plant is known and the controller has to be optimized in closed-loop satisfying a desired closed-loop response. Moreover, robustness in terms of static gain variations is considered.

The recurrent network developed in this work have been successfully used to optimize robust controllers for SISO as well as for MIMO systems. In both cases, the neural network controller achieves the desired performance. Moreover, stability of the closed-loop system is easy to check as this controller has a linear state-space representation. In this case, linear stability conditions can be applied.

- **Modeling and control of the synchronization task.** Here, the application of MPC and neural network based approaches to design a Single-Input Multiple-Output robust controller for the synchronization task in a PHEV is considered.

  In the automotive industry, it is common to use feedforward strategies to achieve a desired performance. The main disadvantage of this approach is that changes in the dynamics of the system are not taken into consideration. This requires a real-time identification of the varying parameters at each operating point. This is also associated with a high effort as mostly the considered dynamic is nonlinear. Here, we show that model based controllers can be successfully applied to solve the synchronization problem in a PHEV. Moreover, robustness is taking into consideration during the controller design. The benefit of using robust controllers is that the desired performance is guaranteed within the specified parameter range. Contrary to feedforward techniques, a real-time identification of the uncertain parameters is not required.

  As it is presented in (Alt et al. 2012) and (Alt et al. 2010), the physical modeling of this plant results in nonlinear differential equations. Moreover, due to the dynamic of the separation clutch parameter uncertainty has to be taken into consideration. This is due to the fact that temperature affects the transmitted torque by the clutch. In this work, we have proposed a method to handle this issue. It is based on identifying a MIMO system with static gain variations. The structure used to model

the dynamic of the synchronization task consists of two inputs, the desired clutch and EM torques, and two outputs the ICE and EM angular velocities. Thereby, the clutch is modelled as an uncertain LTI system. Based on this model, a robust PD as well as a state-space neural network controller are optimized. Simulation results are presented to compare the performance of these two controllers. Moreover, the neural network controller is implemented on the hybrid electric test vehicle. Experimental results are presented at the end of this thesis.

1.3 Thesis Overview

The present thesis is organized in the following sections:

- Section 2 is dedicated to the design and optimization of fractional order PID controllers. First, the robust control problem is defined. Based on the robustness specifications, a desired open-loop frequency response is formulated. An approach is proposed to optimize the parameters of the fractional controllers. At the end of this section, examples are given to shown the effectiveness of the proposed approach in terms of satisfaction of the design requirements.

- In Section 3, the problem of optimizing PID controllers with input, output or states constraints is treated. Based on MPC, the open-loop as well as the closed-loop problem are formulated. The resulting optimization problem turns out to be a bilinear matrix inequality, which is very hard to solve. Using our proposed approach, the BMI problem can be transformed into a set of linear matrix inequalities. Thus, this problem can be solved efficiently using LMI solvers. At the end of this section, an example is given to show the effectiveness of our approach.

- Section 4 discusses neural network controllers. A novel network structure is presented. It is based on representing the closed-loop system with two recurrent neural networks. One network is used to represent the plant. Another network is used to express the controller structure. The result is a closed-loop network, which is trained in the controller parameters. An example is given to show the effectiveness of the proposed approach. Thereby, a robust controller is considered to ensure robustness for uncertain plants.

- In Section 5, the platform used to test the controllers is presented. The modeling of this plant, which is a parallel hybrid electrical vehicle is discussed. Based on measured real data, a multiple-input multiple-output model is identified and validated. Based on this MIMO model, the controller methods presented in Section 3 and 4 are applied. At the end this section a comparison between the two methods is given. Moreover, the implementation of the neural network controller is presented. Thereby, the task consists of synchronizing the angular velocity of the ICE with the angular velocity of the EM.

- Section 6 presents the conclusion of this work. Moreover, an outlook to future works is provided.

The mathematical definitions which are essential for understanding this work are briefly presented and discussed in Appendix B.

2 FOPID-H_∞-controllers

In this chapter, the development of an overall strategy to optimize the parameters of the fractional controllers $(PID)^n$ and $PI^\alpha D^\beta$ is considered. Moreover, the whole approach is put into a Fractional Order PID (FOPID) Matlab-Toolbox. The aim is to make the design and the computation of the fractional controller parameters easy. Our approach is based on the work (Apkarian and Noll 2006) to solve nonsmooth nonconvex problems. Actually, this approach as well as the algorithm presented in this section does not necessary guaranty that the global minimum is reached. For this reason, the gaol of the FOPID-Toolbox is to give the user the possibility to manually tune the parameters, if necessary. In all presented examples in this chapter, it was not necessary to further tune the controller parameters by hand after a minimization step.

2.1 State of the Art (FOPID)

There is no doubt that the PID controller is the most frequently controller type used in control-loops. As it is mentioned in (Åström and Hägglund 2001), more than 90% of all the control-loops are $PIDs$. The design and optimization of this controller is a well studied and still an active field of research. Generally, there exists two methods to tune the parameters of PID controllers. The first method is based on heuristic approaches (Ziegler and Nichols method). The second method is based on modern optimization approaches such as nonsmooth algorithms, see (Apkarian and Noll 2006), or linear matrix inequalities, see (Khatibi et al. 2008).

Recently, a novel approach based on intelligent PID controllers has been presented, see (Fliess et al. 2008). It is a model-free control strategy. This means that a model of the plant to be controlled is not necessary. The main idea is to approximate the dynamic of the system locally as a function of the input signal and the n^{th}-order derivative of the output signal. Moreover, the authors present several examples to show the benefit of model-free PID controllers. The difficulties related to the computation of the n^{th}-derivative of the output signal is discussed in (Fliess et al. 2011). The authors present a method to compute the derivative based on linear parameter identification.

A generalization of the PID controller structure is given by the fractional $PI^\alpha D^\beta$ controller. It was first introduced in (Podlubny 1999). The main advantage of using $PI^\alpha D^\beta$ controllers is the additional degree of freedom given by the fractional orders α and β.

Due to this fact, the fractional $PI^\alpha D^\beta$ controller, when well optimized, outperforms the classical PID controller. The fractional orders makes it possible to impose more constraints on the control-loop without completely leaving the PID controller framework. Instead of the three parameters, K_P K_I and K_D for the classical PID controller, the designer has now two additional parameters, namely α and β.

The tuning of fractional $PI^{\alpha}D^{\beta}$ controllers is discussed in (Monje et al. 2004). An optimization method is presented to tune the parameters of this controller. The authors define a set of six requirements to be achieved by the fractional controller. This set consists of a zero steady-state error, phase margin Φ_m and gain margin G_m specifications, robustness to static gain variations, robustness to high frequency noise and good output disturbance rejection. Five of these six requirements can be achieved using the five parameters of the $PI^{\alpha}D^{\beta}$ controller. First, the set of requirements is transformed into a set of nonlinear equations and then solved numerically using the Matlab optimization function *fmincon*. Simulation results are presented to conclude the work. The design of a fractional PD^{β} for a class of second order systems is considered in (Li et al. 2010). Another fractional order controller structure is presented in (Luo and Chen 2009), namely the fractional controller $[PD]^{\beta}$. A comparison between this controller, the fractional controller PD^{β} and the classical PD controller using simulation and experimental results is given. The application of evolutionary search algorithms to tune $PI^{\alpha}D^{\beta}$ controllers is discussed in (Padhee et al. 2011).

Another class of fractional controllers is proposed by (Tenoutit et al. 2011) and (Tenoutit et al. 2011a), namely $(PI)^n$ and $(PID)^n$. The synthesis of these controllers ensuring robustness to system gain variations is introduced and solved. Time domain constraints are taken into account using the equality of moments between the closed-loop system and a reference fractional model. The Lyapunov condition is used to guaranty stability of the system. Numerical examples are presented to show the performance of these fractional controllers.

In order to automate the design of fractional robust controllers, a French group called CRONE develops a computer software dedicated to this task. CRONE is a French acronym, which means fractional order robust control. The design and optimization of the controller are performed using the Matlab CRONE Toolbox. There exists three generations of this Toolbox. The first generation is presented in (Oustaloup et al. 1993). It is based on the fact that the phase margin variation $\Delta\Phi_m$ consists of the phase margin variation of the plant $\Delta\Phi_P$ and controller $\Delta\Phi_C$. The idea is to reduce the phase margin variation $\Delta\Phi_m$ through the design of a fractional controller with a constant phase around the crossover frequency ω_c. The second CRONE control generation is introduced in (Oustaloup et al. 1993). It is based on designing fractional controllers which provide a constant phase margin Φ_m around the crossover frequency ω_c of the open-loop system. These requirements are transformed into a desired open-loop transfer function in terms of a fractional integrator. The control problem is formulated as an optimization problem which is solved using the simplex algorithm. The third CRONE control generation is presented in (Lanusse et al. 1993). It generalizes the first and second generations by using complex fractional orders. A rview about existing fractional order PID-Toolboxes is given in (Shah and Agashe 2016).

Despite the publications cited above, the number of contributions related to the field of fractional controllers and especially fractional PID controllers is still very modest in

comparison with classical PID controllers. Therefore, there is a need to explore and develop new tuning and optimization methods. The goal of this work is to develop a systematic tool to design fractional controllers in the form $PI^\alpha D^\beta$ and $(PID)^n$. The approach is based on the recently developed nonsmooth optimization techniques (Apkarian and Noll 2006) to solve the low order H_∞ problem. The requirements to be satisfied by the controller are expressed in terms of a desired open-loop response. The optimization problem consists of finding a fractional controller $PI^\alpha D^\beta$ or $(PID)^n$ which best fits this desired response.

2.2 Robust Performance Specifications

In this section, the robustness requirements to be achieved by the controller are presented. These requirements are given in terms of a desired phase margin φ_m^d, a desired crossover frequency ω_c^d and a flat phase around ω_c^d. A method is proposed to transform these specifications into a desired open-loop response. First, an integrator is used to represent all these requirements. As it will be shown, the integrator fails to satisfy all the requirements, simultaneously. For this reason, fractional integrators provide a better way to incorporate all these specifications. Moreover, a method is given to compute the parameters of the fractional integrator. The resulting transfer function represents the desired open-loop response. The time response of the related closed-loop transfer function is explored. The impact of the desired phase margin on the overshoot of the step response is studied. Additionally, the relation between the overshoot and the phase margin is parametrized.

Before presenting the main results of this section, the class of plants considered is defined. This is given by the following uncertain linear single-input single-output (SISO) system

$$G(s) = V \cdot \underbrace{\frac{1}{s^n + a_{n-1}s^{n-1} + \cdots + a_0}}_{G_0(s)} = \frac{Y(s)}{U(s)}. \tag{2.1}$$

The parameter V represents the uncertainty of the system $G(s)$. $G_0(s)$ is the nominal transfer function. Moreover, the range of variations of the parameter V is given by the upper and lower values V_{min} and V_{max}, respectively.

2.2.1 Robust Performance Definition

After defining the class of uncertain plants, the requirements to be satisfied by the controller have to be defined. Following the approach in (Monje et al. 2008) and (Chen et al. 2003), phase margin φ_m and crossover frequency ω_c are used to specify the open-loop performance. Moreover, robustness is taken into consideration by requiring that the variation of the phase margin $\Delta\varphi_m$ around the crossover frequency ω_c to be constant. It means that the phase plot of the open-loop transfer function around ω_c is flat.

Now consider the closed-loop structure shown in Figure 2.1. G is a SISO system with the control input U and the measured output Y. The reference is denoted by R. $K(s)$ is a given LTI controller.

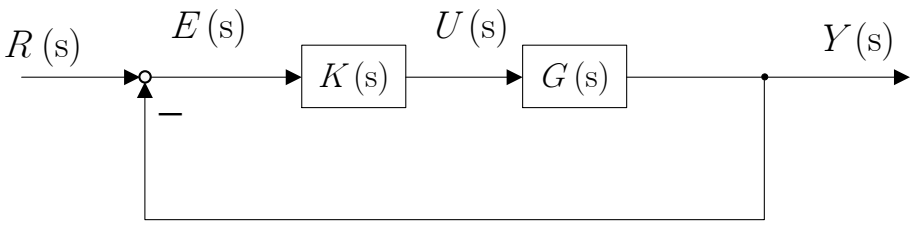

Figure 2.1: Classical feedback control structure

The open-loop transfer function from the error E to the output Y is defined as follows

$$L(s) = K(s)G(s) = \frac{Y(s)}{E(s)}. \tag{2.2}$$

Using the definition of the phase margin φ_m, the crossover frequency ω_c and the robustness objective given by a flat phase around ω_c, the following specifications

$$|L(j\omega_c)|_{dB} = 0\,dB, \tag{2.3}$$

$$\arg(L(j\omega_c)) = -\pi + \varphi_m \tag{2.4}$$

and

$$\frac{d(\arg(L(j\omega))}{d\omega}|_{\omega=\omega_c} = 0 \tag{2.5}$$

are used to express the robust controller requirements, see (Monje et al. 2008) and (Chen et al. 2003).

Equation (2.3) represents the definition of the crossover frequency ω_c. It is a measure how fast the closed-loop response of the system is, see (Gahinet and Apkarian 2011). It is also directly related to the closed-loop bandwidth. Equation (2.4) is the definition of the phase margin. It is a measure of system instability, see (Skogestad and Postlethwaite 2007, p. 35-36). The requirement (2.5) is directly related to the robustness of the closed-loop system. In fact, this specification ensures that robustness in case of static gain variations is achieved.

The specifications (2.3) to (2.5) have to be transformed into a desired open-loop frequency response. To achieve this task, one possibility is to use the integrator

$$L_d(s) = \frac{F}{s}. \tag{2.6}$$

Thereby, the parameter F denotes the integrator gain. Through the appropriate value of F, the integrator (2.6) satisfies the specifications (2.3) to (2.5). To see this fact, substitute the desired open-loop function $L_d(s)$ in Equation (2.3) to (2.5). The following relations

$$|L_d(j\omega_c)|_{dB} = \frac{F}{\omega_c} = 0\,dB, \ \rightarrow F = \omega_c, \tag{2.7}$$

$$\arg(L_d(j\omega_c)) = \arg\left(\frac{F}{j\omega}\right) = arg\,(-j) \equiv -\pi + \varphi_m \ \rightarrow \varphi_m = \frac{\pi}{2} \tag{2.8}$$

and

$$\frac{d(\arg(L_d(j\omega))}{d\omega}|_{\omega=\omega_c} = \frac{d(\arg(-j)}{d\omega}|_{\omega=\omega_c} = 0 \tag{2.9}$$

are obtained. Selecting $F = \omega_c$, all of the three control objectives are covered using the integrator (2.6). The representation of the open-loop transfer function (2.6) can be used to formulate the robust control problem. For example, one can define a desired crossover frequency ω_c^d. Then, set the integrator gain F to ω_c^d. Afterwards, find a controller $K(s)$ such that the resulting open-loop transfer function $L(s) = K(s)G(s)$ is equal to the desired open-loop function $L_d(s)$. In case that such a controller exists, the robustness requirements (2.3) to (2.5) are achieved. The obtained controller is said to be robust.

Unfortunately, the use of the integer order integrator (2.6) does not allow to specify a desired phase margin. Moreover, the phase margin is an outcome of the integrator structure. It is always equal to 90°. To take into account any desired phase margin, other representations are required. In this context, fractional order integrators provide an additional degree of freedom given by the fractional order. This can be used to represent any desired phase margin. An introduction to fractional operators including derivatives and integrals is given in Appendix B.3.

The following transfer function

$$L_d(s) = \frac{F}{s^v} \tag{2.10}$$

is introduced to denote a fractional integrator of order v, which varies from 1 to 2. For $v = 1$, the fractional integrator turns out to be the classical integrator (2.6). For $v = 2$, the transfer function defines a double integrator. In (Bode 1945), Bode introduced the function (2.10) which is known as the Bode ideal transfer function, see (Monje et al. 2008).

Due to the constant phase around the crossover frequency, this function provides an overshoot free step response in case of static gain variations. This characteristic is known as the iso-damping property of the time response, see (Monje et al. 2008).

Given now a desired phase margin φ_m^d and a desired crossover frequency ω_c^d, the parameters F and v of the fractional integrator (2.10) can be computed as follows

$$|L_d(j\omega_c^d)|_{dB} = \frac{F}{(\omega_c^d)^v} = 0\,dB \rightarrow F = \left(\omega_c^d\right)^v \tag{2.11}$$

and

$$\arg\left(L_d\left(j\omega_c^d\right)\right) = \arg\left(\frac{F}{(j\omega)^v}\right) = -v\frac{\pi}{2} \equiv -\pi + \varphi_m \;\rightarrow v = \frac{2}{\pi}(\pi - \varphi_m^d). \tag{2.12}$$

Equation (2.12) shows that the order v depends linearly on the desired phase margin φ_m^d. It can be used to represent the desired phase margin. Using the fractional order v, the phase margin is not restricted to 90° any more.

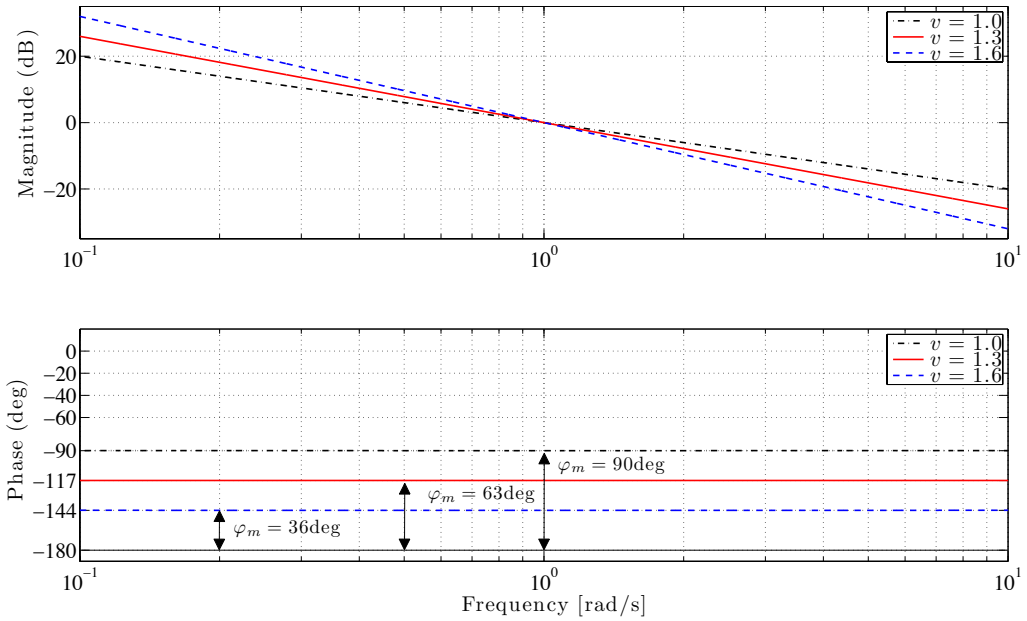

Figure 2.2: Bode plot (magnitude and phase) of the fractional integrator $\frac{1}{s^v}$ for $v = 1.0$ (dashdot black), $v = 1.3$ (red) and $v = 1.6$ (dashed blue) with the related phase margins 90°, 63° and 36°, respectively

Figure 2.2 shows an example of the bode plot of a desired open-loop response given by the fractional integrator $\frac{1}{s^v}$. Thereby, the fractional order v is considered to take the values 1.0, 1.3 and 1.6. The phase plot shows that for $v = 1.0$, the phase margin is equal

to 90°. This case corresponds to the frequency response of the classical integrator (2.6) with $F = 1$. Increasing the order v leads to a smaller phase margin. The phase margin corresponding to $v = 1.3$ is about 63° and that to $v = 1.6$ is about 36°. The frequency plot of the fractional integrator $\frac{1}{s^v}$ have the following characteristic:

- The magnitude plot is a straight line with a slope $-v \cdot 20dB$
- The phase plot is a horizontal line at $-v \cdot 90$

Additionally, Figure 2.2 shows that the phase around the desired crossover frequency $\omega_c^d = 1\ rad/s$ is flat. This fact guaranties that robustness will be achieved in case of static gain variations.

2.2.2 Time Domain Intepretation

To understand the impact of the robustness requirements given in terms of the equations (2.3) to (2.5) on the closed-loop performance, the time domain performance of the fractional integrator in case of static gain variations is explored. For this purpose, the closed-loop structure shown in Figure 2.3 is considered.

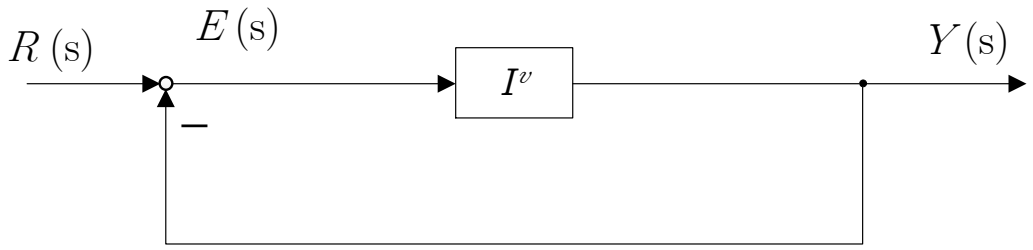

Figure 2.3: Feedback control structure with a fractional open-loop function I^v

The open-loop transfer function from the error signal E to the output Y is given by the fractional integrator $I^v(s)$ with $\frac{F}{s^v}$. Assume that the gain F of the fractional integrator is uncertain. The value of the fractional order v is fixed and known. The uncertain gain is written as $F = V \cdot F_0$ with F_0 consists the nominal static gain. Moreover, the range in which this parameter varies is given by the minimal and maximal value $F_{min} = V_{min} \cdot F_0$ and $F_{max} = V_{max} \cdot F_0$, respectively. The parameter V defines the variation to the nominal gain.

Now consider the nominal case, the related nominal closed-loop transfer function is given by

$$T_0(s) = \frac{\frac{F_0}{s^v}}{1 + \underbrace{\frac{F_0}{s^v}}_{L_0(s)}} = \frac{Y(s)}{R(s)}, \tag{2.13}$$

with $L_0(s)$ is the nominal open-loop transfer function. This fractional integrator $L_0(s)$ is replaced by its frequency band limited CRONE approximation (Melchior et al. 2002) given by $L_0^A(s)$ and discussed in Appendix B.3 through the equations (B.38) to (B.40). The approximated nominal closed-loop function is given by

$$T_0^A(s) = \frac{\prod_{i=1}^{N} \frac{1+\frac{s}{\omega_i'}}{1+\frac{s}{\omega_i}}}{1 + \underbrace{\prod_{i=1}^{N} \frac{1+\frac{s}{\omega_i'}}{1+\frac{s}{\omega_i}}}_{L_0^A(s)}}. \tag{2.14}$$

Now taking the parameter variation into consideration and replacing the function $L_0^A(s)$ with the parameter dependent open-loop function $L^A(s) = V \cdot L_0^A(s)$ leads to the parameter dependent closed-loop function

$$T^A(s) = \frac{V \cdot L_0^A(s)}{1 + V \cdot L_0^A(s)}. \tag{2.15}$$

This function can be used to describe the dynamic of the system as a function of the uncertain parameter V. To show the influence of this parameter, the step response of the transfer function (2.15) for $V_{min} = 0.5$, $V_0 = 1$ and $V_{max} = 1.5$ is considered. Thereby, the fractional order v is equal to 1.24 and the nominal gain $F_0 = 10$. Moreover, the magnitude and the phase of the function $L^A(s)$ are also considered.

The results are shown in Figure 2.4. The overshoot for all variations of the parameter V is about 10 %. It is almost constant. This is due to the constant phase of the fractional integrator in the whole frequency range. Nevertheless, the rise time of the step response is varying depending on the static gain $F = V \cdot F_0$. The Bode plot of the open-loop function $L^A(s)$ shows that the variation of this parameter causes a shifting of the magnitude. During this shifting, the phase margin does not change.

The example above shows that requiring the phase margin to be flat around the crossover frequency results in a constant overshoot of the related closed-loop function. Unfortunately, the overshoot depends on the achieved phase margin. In the scope of this work, the dependency between the desired phase margin and the maximal overshoot is explored. The goal is to find out what kind of relation exists and if it is possible to parametrize it.

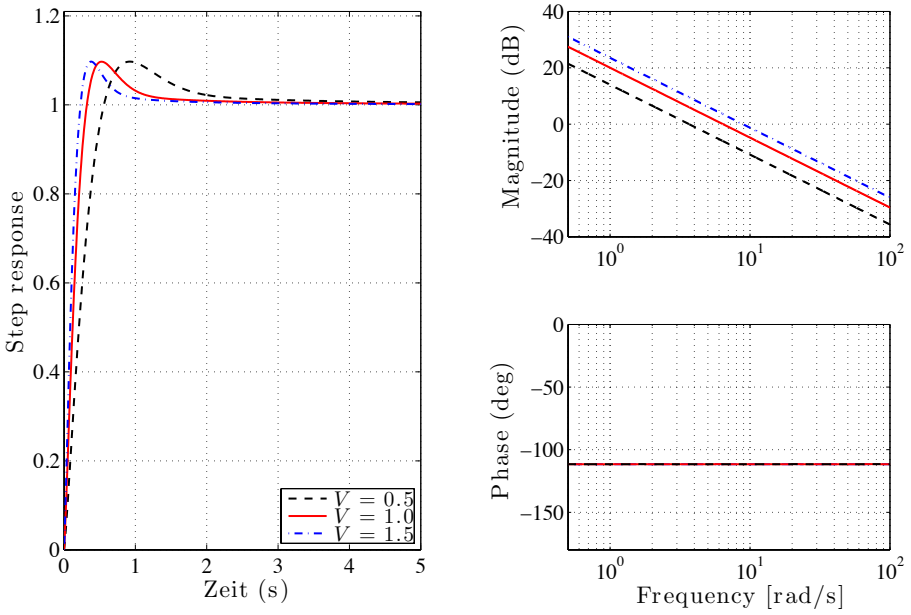

Figure 2.4: Step response (left) of the closed-loop function $T^A(s)$ and bode plot (right) of the open-loop function $L^A(s)$ for $V = 0.5$ (dashed black), $V = 1.0$ (red) and $V = 1.5$ (dashdot blue)

For this purpose, consider the structure in Figure 2.3. This time, the value of the static gain is fixed to K_0 and the fractional order v is varying. The resulting closed-loop function is defined as

$$T_0(s, v) = \frac{\frac{F_0}{s^v}}{1 + \frac{F_0}{s^v}} \qquad \text{with} \qquad 1 \leq v \leq 2, \tag{2.16}$$

which depends on the fractional order v. Using the CRONE approximation (Melchior et al. 2002), the step response for $T_0(s, v = 1)$, $T_0(s, v = 1.2)$, $T_0(s, v = 1.4)$ and $T_0(s, v = 1.8)$ is computed. The results are shown in Figure 2.5.

One can observe that the overshoot of the step response is proportional to the fractional order. Augmenting the order v increases the overshoot of the system. It is evident that a relationship between the overshoot of the closed-loop step response and the fractional order v or the phase margin φ_m exists. In order to get a better insight into the type of this relationship, the following procedure is considered. Let the phase margin φ_m vary from 1° to 90°. Then, compute the related fractional order v using the expression (2.12). Afterwards, substitute the obtained order in the closed-loop transfer function (2.16) and compute the overshoot e_{max} of the related step response.

The result is shown in Figure 2.6. One can remark that a zero overshoot requires a 90° phase margin. This correspond to closed-loop step response based on an open-loop

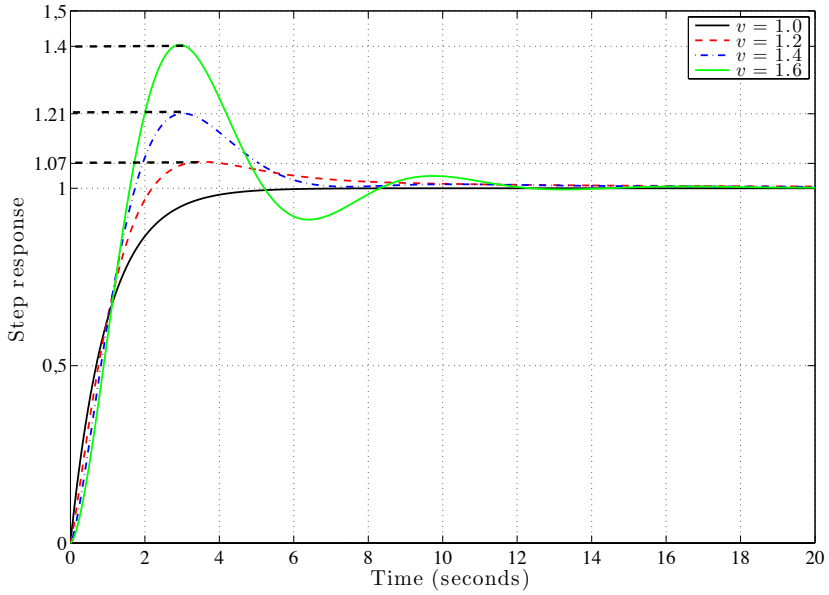

Figure 2.5: Step response of the closed-loop function $T_0(s, v)$ for the fractional orders $v = 1.0$ (black), $v = 1.2$ (red dashed), $v = 1.4$ (blue dashdot) and $v = 1.8$ (green)

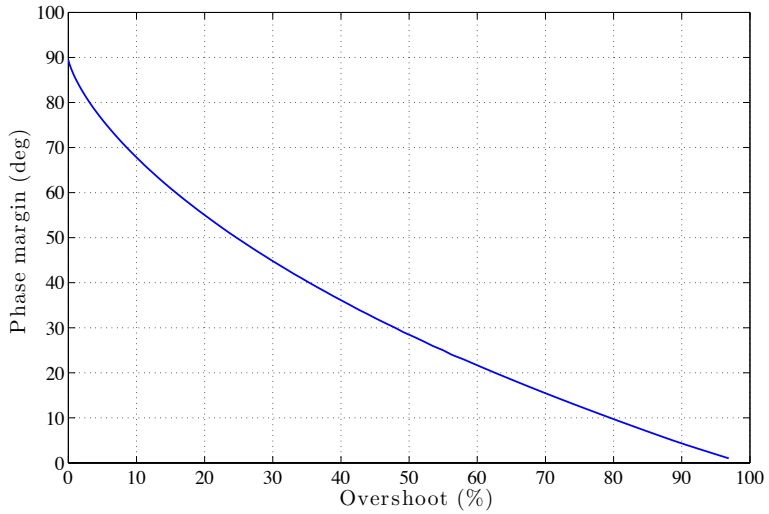

Figure 2.6: Dependency between the phase margin φ_m and the overshoot e_{max}

function in terms of an integrator. Reducing the phase margin of the system results in a higher overshoot. In this work, we are interested in the following relationship

$$\varphi_m(k) = f(e_{max}(k)) \text{ with } k = 1, \cdots, 90. \tag{2.17}$$

This function relates a given maximal overshoot $e_{max}(k)$ to a phase margin $\varphi_m(k)$ for the fractional open-loop function (2.10). For the integer order case, the dependency between the phase margin and the overshoot has been investigated in (Unbehauen 2008). Based on approximating the closed-loop transfer function through a second order system with conjugate complex pole pair, the following simple relationship $\varphi_m[^\circ] + e_{max}[\%] \approx 70$ can be considered, see (Unbehauen 2008, 8.3.22). As a counter part to the integer order case, we investigate here the fractional closed-loop function (2.16). Parametrizing the function (2.17) provides a method to compute the desired phase margin depending on a provided maximal overshoot of the uncertain closed-loop system. For this purpose, a polynomial representation is considered. Moreover, the order of this function is chosen to be four. For this reason, this dependency is approximated using the following polynomial form

$$f(e_{max}(k)) \approx \sum_{i=0}^{4} \mathbf{x}(i) e_{max}^i(k) \tag{2.18}$$

with $\mathbf{x}$ denotes the parameter vector. To compute the parameter vector $\mathbf{x}$, the following optimization problem

Problem 2.1 :

$$\underset{\mathbf{x}}{\text{minimize}} \quad \sum_{k=1}^{N} (\varphi_m(k) - \sum_{i=0}^{4} \mathbf{x}(i) e_{max}^i(k))^2 \tag{2.19}$$

is considered.

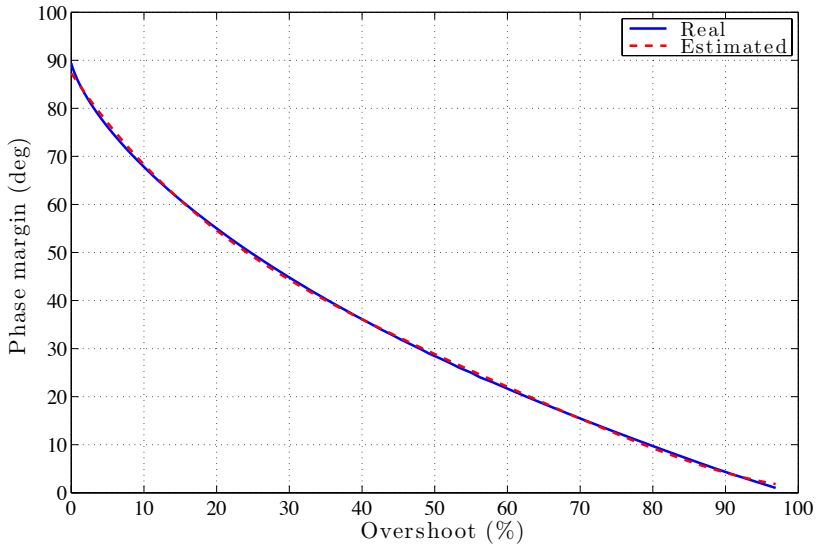

Figure 2.7: Validation of the polynomial approximation, real(blue) and approximated (dash red)

The formulation (2.19) defines a quadratic problem. It is solved using the pseudo-inverse formula, see (Boyd and Vandenberghe 2004, p. 649).

Figure 2.7 shows the results of the polynomial approximation. The optimal parameter vector $\mathbf{x}_{opt}$ provides through the polynomial form $\sum_{i=0}^{4} \mathbf{x}_{opt}(i) e_{max}^{i}(k)$ a good fit of the function (2.17). The obtained parameter $\mathbf{x}_{opt}$ can now be used to provide for each desired maximal overshoot the corresponding phase margin. Using Equation (2.12) and a given desired crossover frequency, the corresponding order of the fractional integrator can be computed.

Remark. *The procedure discussed above to provide the desired phase margin depending on a given overshoot is not general. It is related to the fact that the open-loop transfer function is a fractional integrator. It means that the achieved performance is depending whether or not a controller exists that provides the desired open-loop response given by the fractional integrator.*

2.3 Fractional H_∞ Controllers

In the previous section, the design requirements were discussed. Frequency as well as time domain interpretation of these requirements were presented. In this section, the formulation of the control problem is presented. It is based on the H_∞ control procedure. H_∞ control design is one of the well known modern control techniques. The availability of computer softwares (Sturm 1999) and (Balas et al. 2011) to solve the H_∞ problem and the constantly increasing computing power make H_∞ controllers very popular. The H_∞ continuous- and discrete-time control problems are discussed in detail in (Gahinet and Apkarian 1994). Solvability conditions for both cases are presented. Based on the bounded real lemma and the projection lemma, the H_∞ problem is transformed into a LMI. This optimization problem can then be solved efficiently using existing LMI solvers, see (Labit et al. 2002) or (Nemirovskii and Gahinet 1994). Moreover, an appraoch to reconstruct the controller is also given. A huge number af publications consider the application of H_∞ controllers. For example, the application of a robust controller for position and tracking control of a hydraulic system is presented in (Ahmed et al. 2012). Moreover, a comparison between the H_∞ controller, FXLMS algorithm, PPT_1 controller and a non-model based adaptive control is given. In (Werner et al. 2003), the design of a robust H_∞ controller for power system stabilizers is presented. The controller takes into account the plant parameter variations throught a special representation of the uncertainty.

In this section, the fractional robust control problem is formulated. For this purpose, the H_∞ norm is used to formulate the optimization problem in the frequency domain. Moreover, the loop-shaping technique is briefly discussed. Two fractional robust controllers are considered, namely $PI^\alpha D^\beta$ and $(PID)^n$. Afterwards, an approach is proposed to solve the resulting optimization problem in the controller parameters.

2.3.1 Frequency Domain Representation of the Control Problem

One method to represent the robust control problem is the application of the loop-shaping techniques. Details about this method are given in (Skogestad and Postlethwaite 2007). These techniques are also used in (Gahinet and Apkarian 2011) to shape the open-loop response. However, the authors did not consider the details of this method. In this brief section, the application of the loop-shaping method to formulate the robust control problem is discussed.

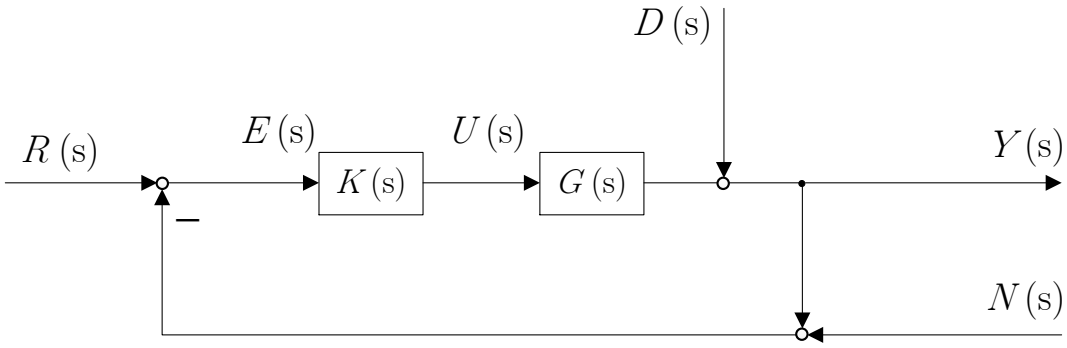

Figure 2.8: Feedback control structure with one degree-of-freedom controller (Skogestad and Postlethwaite 2007, p. 12)

Figure 2.8 shows a closed-loop configuration known as one degree-of-freedom control, see (Skogestad and Postlethwaite 2007). $G(s)$ is a LTI system and consists the plant to be controlled. $K(s)$ is a dynamic output feedback controller. The input signals to this structure are the reference R, the noise N and the disturbance D. The output is denoted by Y.

Now consider the control input defined as

$$U(s) = K(s)(R(s) - Y(s) - N(s)) \tag{2.20}$$

and the related control output as

$$Y(s) = G(s)U(s) + D(s) \ . \tag{2.21}$$

Substituting this in Equation (2.20) leads to the following

$$Y(s) \;=\; \underbrace{(1+G(s)K(s))^{-1}G(s)K(s)}_{T(s)}(R(s)-N(s)) + \underbrace{(1+G(s)K(s))^{-1}}_{S(s)}D(s)\;. \tag{2.22}$$

The transfer functions $S(s)$ and $T(s)$ are called the sensitivity and complementary sensitivity function. These satisfy the following equality

$$T(s)+S(s) \;=\; 1 \tag{2.23}$$

or

$$T(j\omega)+S(j\omega) \;=\; 1\;. \tag{2.24}$$

which defines a frequency constraint on S and T. Constraining the sensitivity function will influence the complementary sensitivity and vice versa. One way to overcome this difficulty is to divide the whole frequency range into low frequency and high frequency range. This can be achieved through the specification of the crossover frequency ω_c. The following approximations

$$|L(j\omega)| \gg 1 \Longrightarrow S(j\omega) = \frac{1}{1+L(j\omega)} \approx L^{-1}(j\omega) \quad \text{for} \quad \omega \ll \omega_c \tag{2.25}$$

and

$$|L(j\omega)| \ll 1 \Longrightarrow T(j\omega) = \frac{L(j\omega)}{1+L(j\omega)} \approx L(j\omega) \quad \text{for} \quad \omega \gg \omega_c \tag{2.26}$$

can be deduced, see (Skogestad and Postlethwaite 2007, p. 56).

Figure 2.9 shows the magnitude of a given open-loop function in terms of an integrator. The crossover frequency ω_c separates the low-frequency from the high frequency range. The main advantage of this separation is twofold. First, the sensitivity function S is in the low-frequency range approximately equal to the inverse of the open-loop function L^{-1}, see expression (2.25). It means that the frequency shaping of the inverse of sensitivity S^{-1} results in shaping the open-loop function in this range. This can be achieved using a transfer function $W_S(j\omega)$ as follows

$$|S(j\omega)| < \frac{1}{|W_S(j\omega)|} \Longrightarrow \frac{1}{|L(j\omega)|} < \frac{1}{|W_S(j\omega)|} \quad \forall\omega \tag{2.27}$$

or

$$|L(j\omega)| > |W_S(j\omega)| \quad \forall\omega. \tag{2.28}$$

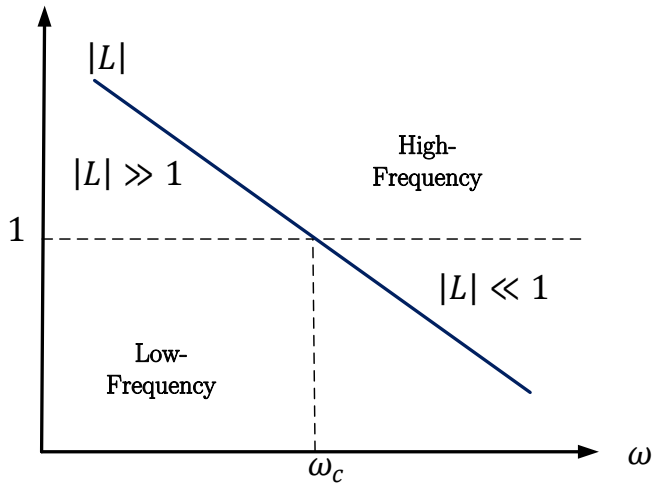

Figure 2.9: Low and high frequency range for the sensitivity magnitude $|L|$

The function W_S is called the sensitivity weighting function. Second, the complementary sensitivity T is in the high frequency range approximately equal to the open-loop function L, see expression (2.26). It means that the frequency shaping of the sensitivity T results in shaping the open-loop function in this range. The shaping of $T(j\omega)$ can be achieved using a transfer function $W_N(j\omega)$ as follows

$$|T(j\omega)| < \frac{1}{|W_N(j\omega)|} \Longrightarrow |L(j\omega)| < \frac{1}{|W_N(j\omega)|} \quad \forall\omega. \tag{2.29}$$

The function W_N is called the complementary sensitivity weighting function. $W_S(s)$ and $W_N(s)$ are knonw as weighting transfer function. They can be used to specify the frequency shape of the sensitivity and the complementary sensitivity functions in the related frequency range. The inequality (2.28) means that the transfer function W_S consists an lower bound on the open-loop function $L = GK$ in the low frequency range. Inequality (2.29), on the other hand, provides a upper bound on the function L given by the weighting transfer function W_N.

One possible procedure to shape the open-loop response $G(s)K(s)$ is to choose the transfer function $W_S(s) = L_d(s)$ and the transfer function $W_N(s) = L_d^{-1}(s)$. Now replacing this in inequality (2.28) leads to the following

$$|S(j\omega)| > |L_d(j\omega)| \qquad \forall\omega \tag{2.30}$$

which can be represented using the H_∞ norm as

$$\|S(j\omega)L_d(j\omega)\|_\infty \ < 1 \ . \tag{2.31}$$

The inequality (2.29) can be then formulated as

$$|T(j\omega)| < |L_d(j\omega)| \qquad \forall\omega \tag{2.32}$$

which can be formulated using the H_∞ norm as

$$\|T(j\omega)L_d^{-1}(j\omega)\|_\infty \ < 1 \ . \tag{2.33}$$

The key idea in the loop-shaping method of the open-loop function $L = GK$ is to combine the constraints given by the expressions (2.31) and (2.33). Inequality (2.31) is active at low frequencies. As it is shown here, it enforces the open-loop response $|L|$ to be above the desired response $|L_d|$. At high frequencies, the constraint (2.33) is active. It enforces the response $|L|$ to be below the response $|L_d|$. Combining both constraints results in shaping the response $|L|$ to have the desired frequency response L_d.

Remark. *Note here that shaping the response $|L|$ is based on the partition of the frequency range into low and high ranges. It does not provide an information about the shape of $|L|$ around the crossover frequency. The achieved performance can be checked by computing $L(s)$.*

After discussing the loop-shaping method, the structure used to realize this idea is given in Figure 2.10. Thereby, N_W and N consist the noise and the weighted noise signal. E and E_W are the error and weighted error signal. Now consider the transfer function from the reference R to the weighted error E_W given by

$$\frac{E_W(s)}{R(s)} = L_d(s)\frac{1}{1+G(s)K(s)} = S(s)L_d(s). \tag{2.34}$$

It represents the realization of the weighted sensitivity function. The counterpart of the weighted sensitivity is the weighted control sensitivity. It is given by the transfer function from the noise signal N to the output Y

$$\frac{Y(s)}{N(s)} = -L_d^{-1}(s)\frac{G(s)K(s)}{1+G(s)K(s)} = -L_d^{-1}(s)T(s). \tag{2.35}$$

The overall optimization problem can be formulated as follows

Problem 2.2 :

$$\underset{K}{\text{minimize}} \quad \gamma = \left\|T_{(R,N)\to(E_W,Y)}(K)\right\|_\infty . \tag{2.36}$$

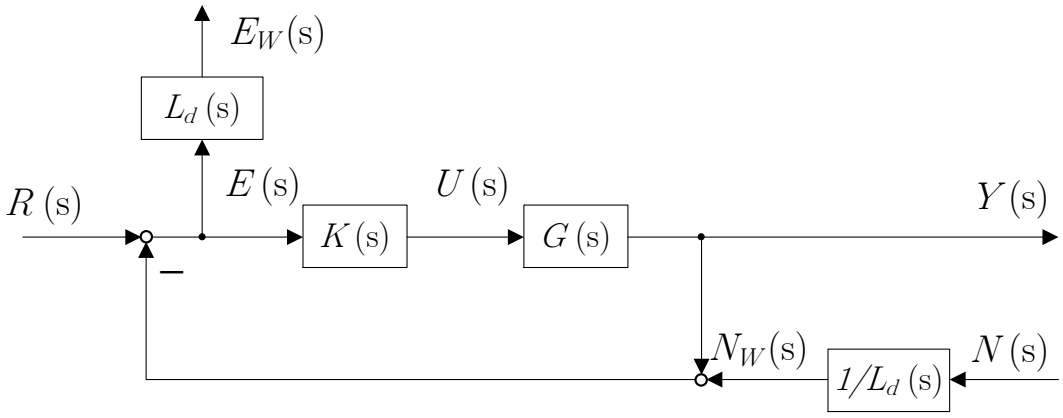

Figure 2.10: Control structure for the open-loop shaping technique

In case that the H_∞ norm γ is less than one, then the open-loop gain $|L| = |GK|$ approximately matches the desired response $|L_d|$, (Gahinet and Apkarian 2011). Problem (2.36) consists of minimizing the H_∞ norm γ in the controller K.

2.3.2 Optimization of $PI^\alpha D^\beta$ Controllers

In this section, the loop-shaping method discussed in the previous section is used to formulate the fractional control problem. For this purpose, consider the closed-loop structure shown in Figure 2.11. $G(s)$ is an uncertain transfer function with $G(s) = V \cdot G_0(s)$. Thereby, $G_0(s)$ is the nominal function and V is the uncertain parameter. The range in which the parameter varies is known and given by V_{min} and V_{max}.

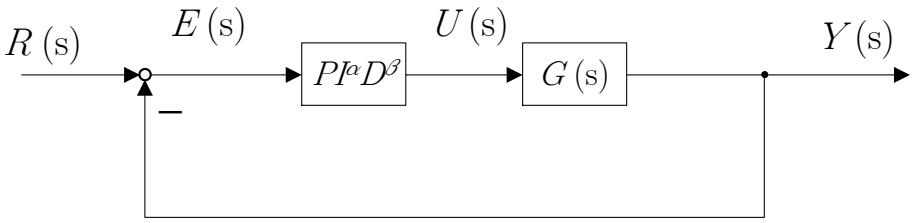

Figure 2.11: Feedback control structure with fractional $PI^\alpha D^\beta$ controller

The fractional order controller $PI^\alpha D^\beta$ is defined as follows

$$PI^\alpha D^\beta : K_P + K_I \frac{1}{s^\alpha} + K_D s^\beta \qquad \text{with } 1 < \alpha, \beta < 2. \tag{2.37}$$

This controller consists a generalization of the classical PID controller and was first introduced in (Podlubny 1999). The goal is to optimize the parameters K_P, K_I, K_D, α and β to achieve a desired phase margin ϕ_m^d, a desired crossover frequency ω_c^d and a flat phase around ω_c^d. Moreover, the discussion in Section 2.2.1 reveals that these requirements can be achieved by enforcing the nominal open-loop transfer function given by

$$L_0(s) = (K_P + K_I \frac{1}{s^\alpha} + K_D s^\beta) G_0(s) \tag{2.38}$$

to provide the response of a fractional order integrator given by

$$L_d(s) = \frac{K}{s^v} \qquad \text{with } K = (\omega_c^d)^v \text{ and } v = 2 - \frac{\pi}{2}\phi_m^d \,. \tag{2.39}$$

Due to the fractional order v, the desired response $L_d(s)$ is approximated using the CRONE approximation (Melchior et al. 2002) given by

$$L_d(s) \approx L_d^A(s) = \prod_{i=1}^{N} \frac{1 + \frac{s}{\omega_i'}}{1 + \frac{s}{\omega_i}}. \tag{2.40}$$

After defining the desired open-loop response L_d, the loop-shaping technique presented in Section 2.3.1 can be now applied. For this purpose, the structure shown in Figure 2.12 is used. Before proceeding with the formulation of the optimization problem, this structure is discussed. For this reason, consider the nominal open-loop transfer function $L_0(s, \alpha, \beta)$ defined as

$$L_0(s,\alpha,\beta) = G_0(s)\left(K_P + K_I \frac{1}{s^\alpha} + K_D s^\beta\right) = G_0(s)\begin{bmatrix} 1 & \frac{1}{s^\alpha} & s^\beta \end{bmatrix}\begin{bmatrix} K_P \\ K_I \\ K_D \end{bmatrix}. \tag{2.41}$$

Using the CRONE approximation (Melchior et al. 2002) for the orders α and β, the approximated nominal open-loop function is given by

$$L_0(s,\alpha,\beta) \approx L_0^A(s,\alpha,\beta) = \underbrace{G_0(s)\begin{bmatrix} 1 & F_I(s,\alpha) & F_D(s,\beta) \end{bmatrix}}_{G_p(s,\alpha,\beta)} \underbrace{\begin{bmatrix} K_P \\ K_I \\ K_D \end{bmatrix}}_{K_S}. \tag{2.42}$$

The transfer functions $F_I(s,\alpha)$ and $F_D(s,\beta)$ denote the CRONE approximation (Melchior et al. 2002) of the fractional integrator $\frac{1}{s^\alpha}$ and the fractional derivative s^β, respectively. Expression (2.42) is used to decompose the fractional $PI^\alpha D^\beta$ controller. Introducing the

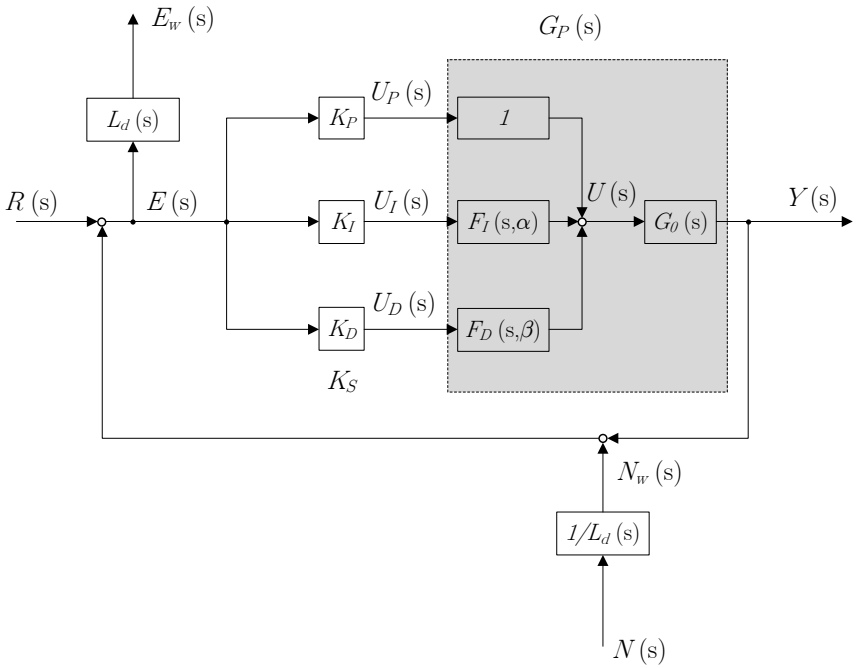

Figure 2.12: Open-loop shaping structure for the $PI^\alpha D^\beta$ controller

augmented plant $G_P(s, \alpha, \beta)$ separates the fractional orders α and β given by the functions $F_I(s, \alpha)$ and $F_D(s, \beta)$ from the parameters K_P, K_I and K_D. Moreover, the fractional orders α and β become now part of the augmented plant G_P. This manipulation allows another view of the control problem.

The system under consideration has now three inputs U_P, U_I and U_D and one output Y. In this case, the controller structure is given by the static controller K_S, see Figure 2.12. It has one input E and three outputs U_P, U_I and U_D. Now assume that the goal is to find a static output feedback controller K_S that enforces the open-loop function $\tilde{L}_0^A(s, \alpha_0, \beta_0)$ to achieve the desired open-loop response $L_d^A(s)$. Thereby α_0 and β_0 are fixed known values of the fractional orders α and β. For this purpose, let the transfer function from the inputs (R, N) to the output (E_W, Y) for $\alpha = \alpha_0$ and $\beta = \beta_0$ be denoted by $T_{(r,n_w)\to(y,e_w)}(K_S, \alpha_0, \beta_0)$. The entries K_S, α, β in $T_{(r,n_w)\to(y,e_w)}(\cdot, \cdot, \cdot)$ means that the optimization is in terms of the static controller K_S with known fractional order values α_0 and β_0. Using the method given in Section 2.3.1 given by the expression (2.36), the controller optimization problem can be formulated as follows

Problem 2.3 :

$$\underset{K_S \in \Omega}{\text{minimize}} \quad \underbrace{\left\| T_{(r,n)\to(y,e_w)}(K_S, \alpha_0, \beta_0) \right\|_\infty}_{\gamma(\alpha_0,\beta_0)}. \tag{2.43}$$

Problem 2.3 consists of minimizing the H_∞ norm $\gamma(\alpha_0, \beta_0)$ with respect to the parameters K_P, K_I and K_D given by the static controller K_S. The condition $K_S \in \Omega$ is a structural constraint on the controller. In our case, this constraint is represented by the set of all static feedback controllers. Without the restriction $K_S \in \Omega$, the full order H_∞ controller is covered. In this case, K_S is a dynamic dynamic with a specified state-space representation. Problem (2.43) falls into the scope of convex optimization and can be solved efficiently. To solve this problem, one can first define a state-space realization of the so-called generalized plant $\tilde{G}_P(s)$. It consists of the plant $G_P(s)$ and the weighting transfer functions $L_d^A(s)$ and $1/L_d^A(s)$ shown in Figure 2.12. This MIMO transfer function is defined as follows

$$\begin{bmatrix} E_W(s) \\ Y(s) \\ E(s) \end{bmatrix} = \underbrace{\begin{bmatrix} -L_d^A(s)G_P(s) & L_d(s) & -1 \\ G_P(s) & 0 & 0 \\ -G_P(s) & 1 & -\dfrac{1}{L_d^A(s)} \end{bmatrix}}_{\tilde{G}_P(s)} \begin{bmatrix} \mathbf{U}(s) \\ R(s) \\ N(s) \end{bmatrix}, \tag{2.44}$$

with the input vector $\mathbf{U}(s) = [U_P(s)\ U_I(s)\ U_D(s)]^T$. It describes the augmented control problem. The related time domain state-space representation of the above MIMO system (2.44) is given as follows

$$\tilde{G}_P : \begin{cases} \dot{\mathbf{x}}(t) &= \mathbf{A}\ \mathbf{x}(t) + \mathbf{B}_1\ \ \mathbf{w}(t) + \mathbf{B}_2\ \ \mathbf{u}(t) \\ \mathbf{z}(t) &= \mathbf{C}_1^T\, \mathbf{x}(t) + \mathbf{D}_{11}\, \mathbf{w}(t) + \mathbf{D}_{12}\, \mathbf{u}(t) \\ e(t) &= \mathbf{C}_2^T\, \mathbf{x}(t) + \mathbf{D}_{21}\, \mathbf{w}(t) + \mathbf{D}_{22}\, \mathbf{u}(t). \end{cases} \tag{2.45}$$

$\mathbf{x}(t)$ denotes the state vector of all variables describing the dynamic of the generalized plant. $\tilde{G}_P$ maps the control input $\mathbf{u}(t) = [u_1(t)\ u_2(t)\ u_3(t)]^T$ and the performance channel $\mathbf{w}(t) = [r(t)\ n_w(t)]^T$ to the output $\mathbf{z}(t) = [e_w(t)\ y(t)]^T$. The structure of the full-order controller is given by

$$K_S : \begin{cases} \dot{\mathbf{x}}_K(t) &= \mathbf{A}_K\ \mathbf{x}_K(t) + \mathbf{B}_K\ e(t) \\ \mathbf{u}(t) &= \mathbf{C}_K^T\ \mathbf{x}_K(t) + \mathbf{D}_K\ e(t)\ . \end{cases} \tag{2.46}$$

$\mathbf{x}_K(t)$ denotes the state vector of all variables required to describe the dynamic of the controller. With the help of the bounded real lemma and the projection lemma (Gahinet and Apkarian 1994), problem (2.43) can be transformed into a LMI form and then be

solved efficiently using LMI solvers. The obtained controller is full order: it means that the size of A_k is equal to the size of A. This controller is well known under H_∞-full order controller or only H_∞ controller.

As explained in (Apkarian and Noll 2006), requesting K_S to have a specified order and structure given by the set Ω changes the whole situation. Problem 2.43 can not be converted into the LMI form or any other convex form. To solve this problem, other methods are required. The authors in (Apkarian and Noll 2006) have proposed a new nonsmooth optimization technique to solve the H_∞ problem under structural constraints.

Example 2.1

To better understand the concept of structured H_∞ controller, a PID controller for a SISO-system is considered. The computation is based on the technique proposed in (Apkarian and Noll 2006), which is implemented in the Matlab function *Hinfstruct*. For this purpose, consider now the following uncertain plant

$$G(s) = V \cdot \underbrace{\frac{1}{s(Ts+1)}}_{G_0(s)} \text{ with } T = 0.4 \text{ and } V \in [0.5\ 1.5] \tag{2.47}$$

taken from (Luo and Chen 2009). Assume that a desired phase margin $\phi_m^d = 70°$ and a desired crossover frequency $\omega_c^d = 10\,rad/s$ are provided. Moreover, the controller should be robust in the presence of static gain variations. Due to the integral action of the plant, it is not necessary that the controller also has an integral action. For this reason, the optimization problem consists of finding a PD controller, which provides the desired requirements. Using now the expression (2.39), the desired open-loop response is

$$L_d(s) = \frac{16.68}{s^{1.22}} \approx \underbrace{\prod_{i=1}^{N} \frac{1+\frac{s}{\omega_i'}}{1+\frac{s}{\omega_i}}}_{L_d^A(s)} \tag{2.48}$$

with L_d^A is the fractional order approximation using the CRONE method (Melchior et al. 2002). The order N of this approximation is very high in case that the approximation should be valid in the whole frequency range. In case that this range is finite $[1e^{-4}\ 1e^{4}]\,rad/s$, the order N is set to be at least equal to the number of decades. To get an accurate approximation, the order N of the approximation (2.48) is set to 10. Moreover, we replace the ideal derivative D in the PD controller with the filtered derivative D_f to make it realizable. Now, the problem is to compute a robust PD_f controller for the uncertain plant given by (2.47), which provides the desired response (2.48).

After defining the control requirements, the optimization problem has to be set. It consists

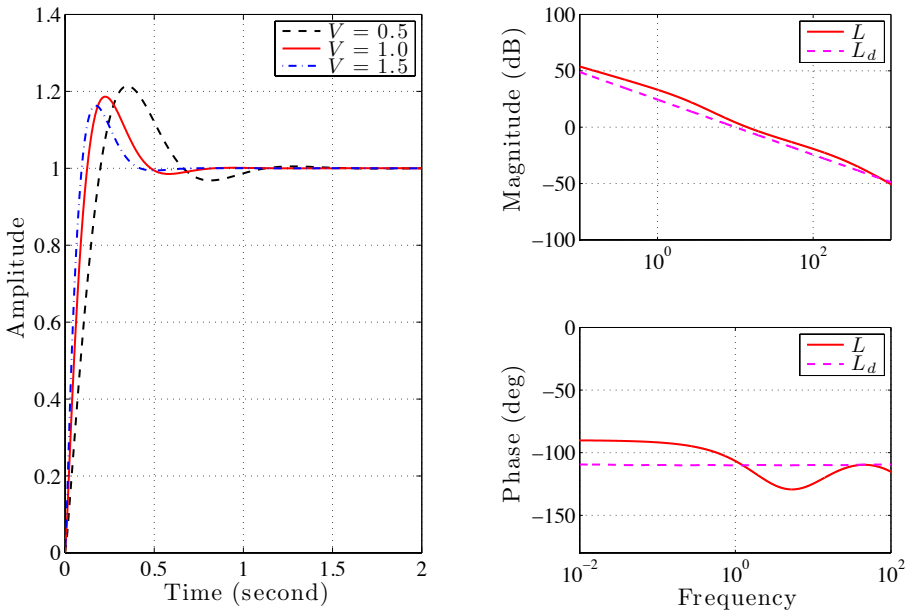

Figure 2.13: Step response with PD_f controller (left), desired and actual open-loop response (right)

of solving the problem (2.43) for $\alpha_0 = 0$ and $\beta_0 = 1$. Figure 2.13 shows the step response with the obtained PD_f controller

$$PD_f : K(s) = \underbrace{48.9}_{K_P} + \underbrace{4.43}_{K_D} \cdot \frac{s}{\underbrace{0.00376}_{\tau_f} s + 1} \tag{2.49}$$

for the nominal, minimal and maximal case. It is clear that the overshoot is not constant for all values of the uncertain parameter V. Moreover, the Bode diagram of the desired response L_d and the achieved open-loop response L with the obtained PD_f controller is also shown. The controller fails to provide a satisfactory solution. The requirements given through the desired open-loop response (2.48) are not achieved. The PD_f controller does not satisfy the specifications.

To improve the performance of the PD_f controller, we consider the optimization of the fractional orders α and β. They are used as an additional degree of freedom to further optimize the H_∞ norm in the optimization problem (2.43). Formally speaking, the following optimization problem

Problem 2.4 :

$$\underset{K_S,\alpha,\beta}{\text{minimize}} \quad \left\| T_{(r,n_w)\to(y,e_w)}(K_S,\alpha,\beta) \right\|_\infty \tag{2.50}$$

is considered. It consists of minimizing the H_∞ norm in terms of the static feedback

controller K_S and the fractional orders α and β. Problem 2.4 can not be solved directly using the technique in (Apkarian and Noll 2006). In this work, a method is presented to optimize the optimization parameters α, β and K_S. It is based the Steepest Descent method.

One method to solve the optimization problem (2.50) is to grid the α-β parameters space and then pick-up the optimal pair $(\alpha_{Opt}, \beta_{Opt})$ with the lowest H_∞ norm $\gamma(\alpha_{Opt}, \beta_{Opt})$. One can proceed as follows:

Algorithm 2.1 :

1. *Define a resolution step Δ_k in α and β.*
2. *Grid the parameter space α-β using $\alpha_k = 1 + k\Delta_k$ and $\beta_k = 1 + k\Delta_k$ with $k = 1, 2, \cdots N$ and $0 < \alpha_k, \beta_k < 2$.*
3. *Solve for each pair (α_k, β_k) the following optimization problem*

$$\gamma_k(\alpha_k, \beta_k) = \underset{K_{S_k} \in \Omega}{\text{minimize}} \quad \underbrace{\left\| T_{(r,n_w)\to(y,e_w)}(K_{S_k}, \alpha_k, \beta_k) \right\|_\infty}_{\gamma(\alpha_k, \beta_k)} \tag{2.51}$$

   and save $\gamma_k(\alpha_k, \beta_k)$.

4. *Pick-up the pair $(\alpha_{Opt}, \beta_{Opt})$ and the related controller K_{S_k} with the smallest H_∞ norm $\gamma_k(\alpha_{Opt}, \beta_{Opt})$.*

The main disadvantage of the gridding based method is the resolution step Δ_k. One has first to solve the optimization problem (2.43) for all pairs (α_k, β_k) and pick-up the optimal solution. For example, a step $\Delta_k = 0.01$ for α and β would leads to a parameter space 200×200. This requires 40000 function evaluations of the H_∞ norm. At each evaluation, the optimization problem has to be solved. A better way to find the optimal solution is to optimize over the fractional orders. For this reason, consider now the following problem

Problem 2.5 :

$$\underset{\alpha,\beta}{\text{minimize}} \quad \underbrace{\left(\underset{K_S}{\text{minimize}} \left\| T_{(r,n_w)\to(y,e_w)}(K_S, \alpha, \beta) \right\|_\infty \right)}_{\gamma(\alpha,\beta)}, \tag{2.52}$$

which consists of two optimization problems. The first problem is to optimize the controller K_S for a given pair α_k and β_k. The second problem consists in further minimizing the H_∞ norm by optimizing over the fractional orders. The approach can be summarized as follows

Algorithm 2.2 :

1. *Initialize* α_0 *and* β_0
2. *Compute the gradient* d_k *with respect to* α_k *and* β_k
3. *If* $\|d_k\|_2$ *smaller then a value t STOP, else update* α_k *and* β_k *and go back to step* (2).

To update α_k and β_k in step (3), we used the Steepest-Descent algorithm with fixed step. The computation of the gradient of the H_∞ norm in problem (2.52) with respect to α and β was performed numerically. The values of K_P, K_I and K_D are computed by substituting the obtained values of α_k and β_k in problem (2.51).

Remark. *The optimization problem (2.52) is nonconvex, due to the nonlinear dependence of the transfer function* $T_{(r,n_w)\to(y,e_w)}(\cdot)$ *on the parameters* α, β *and* K_S, *and nonsmooth due to the use of the* H_∞ *norm as a cost function. Basically, such problems are very hard to solve. The nonsmooth character of the problem is treated using nonsmooth techniques provided by (Apkarian and Noll 2006). In what concerns the nonconvexity, the authors proposed to restart the algorithms from different start values. Moreover, we provide using our Matlab Toolbox the possibility to further tune the controller parameters by hand after a minimization step, if necessary. In all the examples provided later, it was not necessary to tune the obtained controller further.*

2.3.3 Optimization of $(PID)^n$ Controllers

In the previous section, the optimization of robust $PI^\alpha D^\beta$ controllers was presented. To evaluate the performance of this controller, the design of another fractional controller is considered, namely $(PID)^n$. This kind of controller was presented in (Tenoutit et al. 2011) and (Tenoutit et al. 2011a). Contrary to these works, the fractional order n in this work is considered as a free parameter. It can be used to get an additional degree of freedom in optimizing the controller. The requirements to be fulfilled by the controller are the same used for the $PI^\alpha D^\beta$ controller.

Figure 2.14 shows the closed-loop structure with the fractional order controller $(PID)^n$. $G(s) = V \cdot G_0(s)$ consists the uncertain transfer function. The fractional controller under consideration is given by

$$(PID)^n : \frac{1}{s^n}\left(K_P + K_I\frac{1}{s} + K_D s\right) \approx \frac{1}{s^n}\left(K_P + K_I\frac{1}{s} + K_D \underbrace{\frac{s}{\tau s + 1}}_{D_f}\right). \tag{2.53}$$

The strict derivative s is replaced by a filtered derivative part D_f for practicability. As it is the case for the $PI^\alpha D^\beta$ controller, we are interested into designing a robust controller

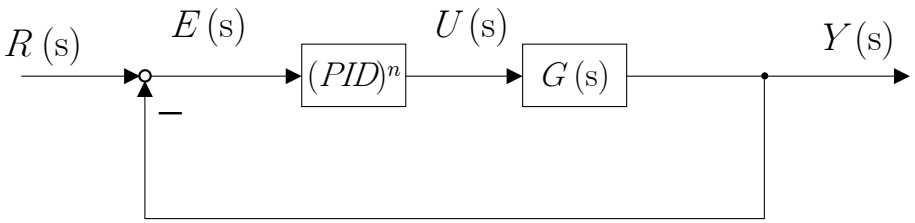

Figure 2.14: Feedback control structure with $(PID)^n$ controller

for the uncertain plant G. The goal is to optimize the parameters K_P, K_I, K_D and n to achieve a desired phase margin ϕ_m^d, a desired crossover frequency ω_c^d and a flat phase around ω_c^d. As it is the case for the $PI^\alpha D^\beta$ controller, these requirements are expressed in terms of a desired open-loop response $L_d(s)$ using the expression (2.39) or its approximation $L_d^A(s)$ using the expression (2.40).

A given controller $(PID_f)^n$ is said to be robust if the open-loop function

$$L_0(s) = G_0(s)\frac{1}{s^n}\underbrace{\left(K_P + K_I\frac{1}{s} + K_D\frac{s}{\tau s+1}\right)}_{K(s)} \tag{2.54}$$

provides approximately the response given by the fractional integrator (2.40).

Unlike in the $PI^\alpha D^\beta$ controller case, another structure to formulate the control problem is considered, namely Model Following Control (MFC), see (Gahinet and Apkarian 2011). It is based on defining a desired reference model instead of a desired open-loop response. For this reason, the structure shown in Figure 2.15 is considered.
The transfer function $T_d(s)$ provides the desired closed-loop response. It is computed as follows

$$T_d(s) = \frac{\frac{F}{s^v}}{1+\underbrace{\frac{F}{s^v}}_{L_d(s)}} = \frac{1}{\frac{1}{F}s^v+1} \approx \overbrace{\frac{1}{\frac{1}{F}\left(\prod_{i=1}^{N}\frac{1+\frac{s}{\omega_i'}}{1+\frac{s}{\omega_i}}\right)+1}}^{T_d^A(s)}. \tag{2.55}$$

$T_d^A(s)$ consists the approximated desired closed-loop response. The control problem can now be formulated. It consists of finding a controller that provides the response of the reference model given by the function (2.55). This can be achieved by minimizing the error E_W between the output Y and the output of the reference model Y_d using the H_∞

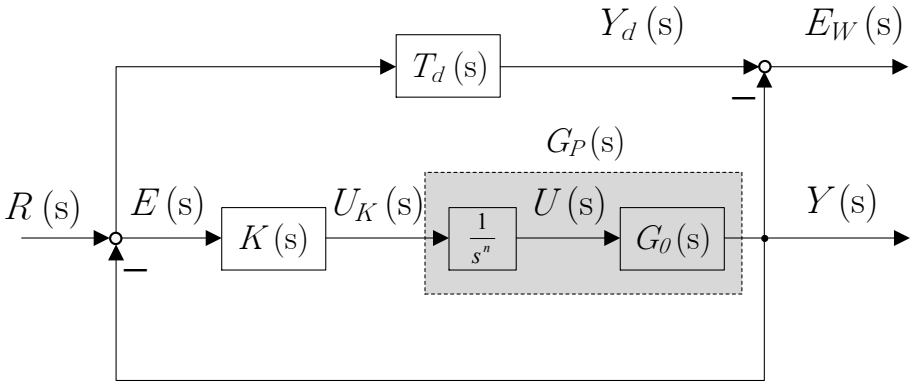

Figure 2.15: Model Following Control structure with $(PID)^n$ controller

norm.

Assume now that the fractional order n is set to a constant value n_k. Let the transfer $G_p(n_k, s)$ be defined as follows

$$G_P(s, n_k) = H(s, n_k)G_0(s) \tag{2.56}$$

with $H(s, n_k)$ is the CRONE approximation (Melchior et al. 2002) of the fractional integrator $\frac{1}{s^{n_k}}$. The function $G_P(s, n_k)$ is an augmented plant, which is used to make the understanding of the optimization problem easier. As it is the case in Section 2.3.2, the H_∞ norm is used to formulate the optimization problem. For this purpose, define the transfer function from the reference R to the error E_W as $T_{R\to E_W}(s, n_k)$. The optimization problem is given as follows

Problem 2.6 :

$$\gamma(n_k) = \underset{K}{\text{minimize}} \quad \|T_{R\to E_W}(K, n_k)\|_\infty . \tag{2.57}$$

$\gamma(n_k)$ is the H_∞ norm for a fixed order n_k. This problem consists of optimizing a PID_f controller for the augmented plant (2.56) with a fixed order n_k. Moreover, minimizing the H_∞ norm $\gamma(n_k)$ enforces the closed-loop system to achieve the desired response given by $T_d^A(s)$. Setting now the order n_k to zero applies the computation of a PID_f controller for the original plant $G_0(s)$. Solving this problem falls into the scope of application of the Matlab function *Hinfstruct* given by (Apkarian and Noll 2006).

The performance of the controller can be improved by letting the fractional order in Problem 2.6 be an additionally optimization parameter. Instead of solving Problem 2.6, the following problem

Problem 2.7 :

$$\underset{n}{\text{minimize}} \quad \underbrace{\left(\underset{K}{\text{minimize}} \left\|T_{R\to E_W}(K,n)\right\|_\infty\right)}_{\gamma(n)} \tag{2.58}$$

is considered. Problem 2.7 can be solved using the same approach presented in the previous section, which can be summarized as follows

Algorithm 2.3 :

1. *Initialize n by n_0*
2. *Compute the gradient d_k with respect to n*
3. *If $\|d_k\|_2$ smaller then a value t STOP, else update n_k and go back to step* (2).

To update the fractional order n_k in step (3), the steepest descent algorithm with fixed step is used. The computation of the gradient of the H_∞ norm $\gamma(n)$ into the optimization Problem 2.7 with respect to n was performed numerically. The values of K_P, K_I and K_D are computed by substituting the obtained values of n_k into the optimization problem (2.58).

Remark. *As in the $PI^\alpha D^\beta$ controller case, Problem 2.7 is nonconvex and nonsmooth. The solution obtained through the proposed optimization does not necessary consist the global minimum. For this purpose, the possibility to further tune the obtained controller if necessary is given through the FOPID-Toolbox. The examples shown in the next section did not require to use this feature. All the obtained controller did satisfy the robustness requirements imposed by the desired open-loop response.*

2.4 FOPID-Toolbox

In this section, the fractional order FOPID-Toolbox developed in this work is presented. It is dedicated to the design of fractional robust controllers in the form $PI^\alpha D^\beta$ or $(PID)^n$ for uncertain plants. It consists a Matlab Graphical User Interface (GUI) in which the user provides the plant to be controlled and the requirements to be achieved. The two approaches presented in Section 2.3.2 and 2.3.3 are implemented. The related optimization problems are internally set and solved iteratively. The achieved performance is plotted in term of the step response and the Bode diagram.

To be able to use the FOPID-Toolbox, the Matlab Robust Control Toolbox is required, see (Balas et al. 2011). Moreover, we are considering the controller optimization for the

following class of plant

$$G(s) = V \cdot \underbrace{\frac{1}{s^n + a_{n-1}s^{n-1} + \cdots + a_0}}_{G_0(s)} e^{-sT_d} \tag{2.59}$$

with $G_0(s)$ is a stable plant without any zeros. T_d is a given time-delay. Related to the optimization of fractional controllers using a Matlab-Toolbox, the CRONE Group has been active working on this area. The first contribution is given by the first generation of the CRONE-Toolbox, see (Oustaloup et al. 1993). In this case, the controller is given by the form $K(s) = C_0 s^n$ with n is a real number. It provides the constant phase $n\frac{\pi}{2}$ around its crossover frequency. The second generation is based on defining the desired open-loop response as a fractional integrator $L_d(s) = \left(\frac{\omega_C^d}{s}\right)^n$. The controller is obtained as $K(s) = \frac{L_d(s)}{G_0(s)}$, see (Oustaloup et al. 1993). The use of complex orders is provided in (Lanusse et al. 1993). In all these contributions, the optimization of fractional controller was achieved indirectly. It means that the author does not optimize the controller but rather provides the design of a fractional optimal open-loop response. The controller is obtained by approximating the function $K(s) = \frac{L_d(s)}{G_0(s)}$ using optimization techniques as simplex.

The main advantage of using our FOPID-Toolbox is that we are considering the optimization of the given fractional controller structures

$$(PID)^n : \frac{1}{s^n}\left(K_P + K_I\frac{1}{s} + K_D s\right) \quad \text{and} \quad PI^\alpha D^\beta : K_P + K_I\frac{1}{s^\alpha} + K_D s^\beta \tag{2.60}$$

in the parameters K_P, K_I, K_D, α, β and n, directly. The optimization problems (2.58) and (2.52) are set in terms of these parameters. This is not the case for the CRONE-Toolbox. Moreover, the optimal values of these parameter have to provide a fractional controller that achieves the required performance given by a desired open-loop response $L_d(s) = \left(\frac{\omega_c^d}{s}\right)^v$, see Equation (2.10) . The comparison of the obtained performance to that of a classical PID controller is easily obtained through setting n to zero or α and β to one.

2.4.1 Application to a Second Order System

To show the simplicity of using the proposed approach to optimize fractional controllers, the obtained PD^β controller in our work (Svaricek and Lachhab 2016) for the second order plant presented in (Luo and Chen 2009) is given in this section. The plant is defined as follows

$$G(s) = V \cdot \underbrace{\frac{1}{s(Ts+1)}}_{G_0(s)} \qquad \text{with } V \in [0.8\ 1.2]. \tag{2.61}$$

Thereby, $G_0(s)$ is the nominal plant. The time constant T is equal to $0.4s$. The requirements to be achieved by the controller are given by the desired phase margin $\phi_m^d = 70°$, the desired crossover frequency $\omega_c^d = 10\,rad/s$ robustness in case of varations of the gain V. These specifications are transformed using the expression (2.39) into the following desired open-loop function

$$L_d(s) = \frac{16.68}{s^{1.22}}. \tag{2.62}$$

The goal is to find a PD^β controller

$$PD^\beta(s) = K_P + K_D s^\beta. \tag{2.63}$$

The parameters K_P, K_D and β are optimized such that the frequency response of the nominal open-loop function $L_0(s) = G_0(s)PD^\beta(s)$ approximately fits the desired response given by the function 2.62. Using the CRONE method, $L_d(s)$ is approximated with the order $N = 8$ in the frequency range given by $\omega_{min} = 0.05\,rad/s$ and $\omega_{max} = 4000\,rad/s$. For the fractional part s^β in 2.63, a CRONE approximation with $N = 4$, $\omega_{max} = 0.1\,rad/s$ and $\omega_{max} = 1000\,rad/s$ is used.

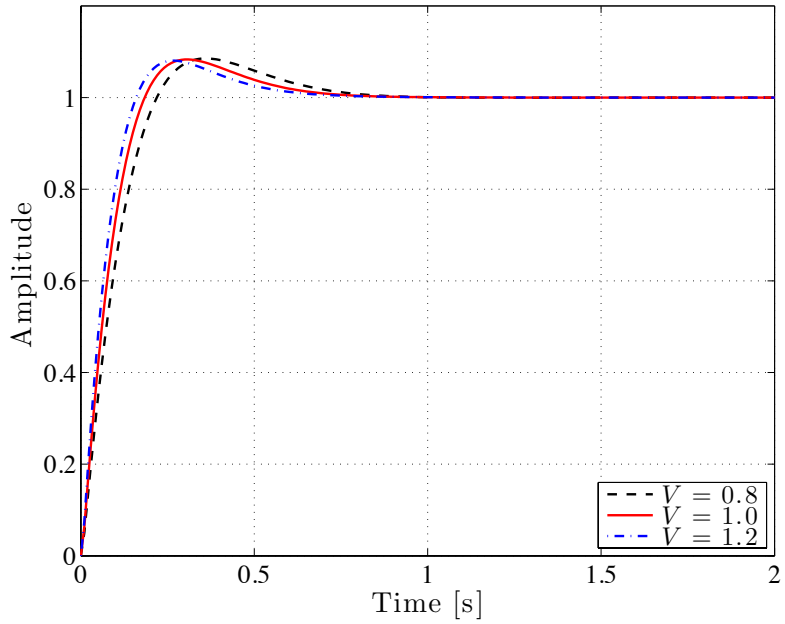

Figure 2.16: Step response with PD^β controller, for $V = 0.8$ (dashed black), $V = 1.0$ (red) and $V = 1.2$ (dashdot blue)

The optimized controller parameters are $K_P = 21.69$, $K_D = 4.8$ and $\beta = 0.9535$. The

CRONE approximation leads to the following controller

$$K(s) = \frac{0,0000751s^5 + 4,914s^4 + 602,4s^3 + ...}{0,00000346s^5 + 0,005s^4 + 1,807s^3 + ...} \cdot \frac{... + 9092s^2 + 34750s + 24840}{... + 142,8s^2 + 1313s + 1118}. \tag{2.64}$$

The controller order is five as the CRONE approximation was extended with a low-pass transfer function with the time constant $T_1 = 0.0025s$. The step response with the controller (2.64) is shown in Figure 2.16. The overshoot for all variations of the static gain $V = 0.8$, $V = 1.0$ and $V = 1.2$ is approximately constant.

Consider now the frequency response of nominal open-loop function $L_0(s) = G_0(s)K(s)$ and the desired response $L_d(s)$ shown in Figure 2.17. The phase around the crossover frequency $10\,rad/s$ is approximately constant. This leads to a constant overshoot in case of variations of the static gain.

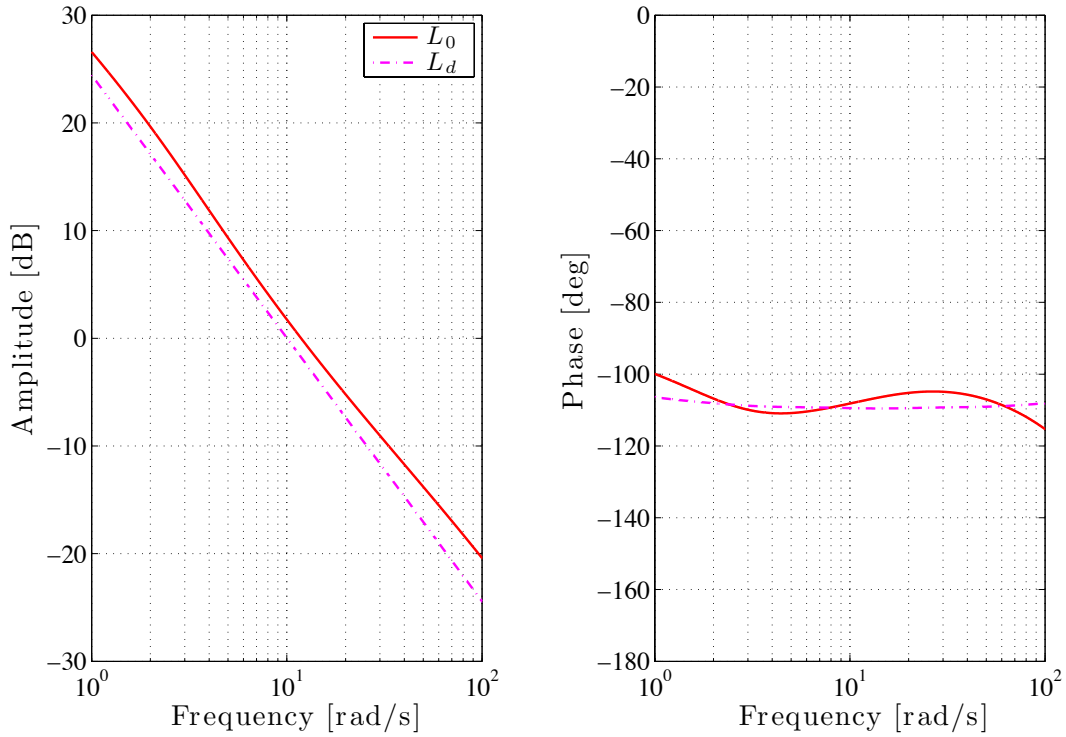

Figure 2.17: Comparison desired L_d and nominal L_0 open-loop response

In case that the parameter variation is $\pm 20\%$, the required performance is achieved. Even if this variation is $\pm 50\%$, the robustness is also guaranteed, see Figure 2.18. In (Li et al. 2010) and (Luo and Chen 2009), the results of a PD^β controller with $K_P =$

13.68, $K_D = 5.1$ and $\beta = 0.844$ are given. The presented figures show approximately an overshoot around 7%. Due to the low K_P value, the step response shows a larger rise time and settling time.

The fractional order $\beta = 0.9535$ of the obtained PD^β controller using our approach is not significantly far away from one, such that the proposed method for $\beta = 1$ provides a classical PD controller with $K_P = 17.94$ and $K_D = 3.84$, which provides the required performance.

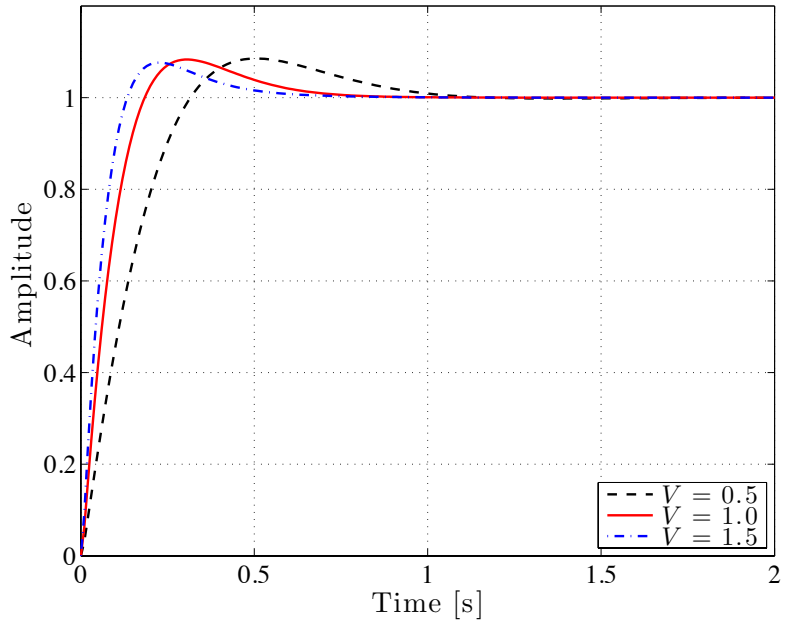

Figure 2.18: Step response with PD^β controller, for $V = 0.5$ (dashed black), $V = 1.0$ (red) and $V = 1.5$ (dashdot blue)

2.4.2 Application to a Time Delay System

The second example is a second order time delay system. In (Tenoutit et al. 2011a) and (Tenoutit et al. 2012), the authors consider the design of a robust controller for such systems to ensure a constant overshoot in case of static gain variations. The performance of this controller will be here compared to the results in (Svaricek and Lachhab 2016), obtained using our proposed method.

The plant under consideration is given by

$$G(s) = V \cdot \frac{1}{(1+s)(1+2s)} e^{-sT_d} \qquad \text{with } V \in [0.5\ 2]. \tag{2.65}$$

The delay constant T_d is equal to $0.2s$. The desired crossover frequency is $\omega_c^d = 0.5\, rad/s$.

The robustness in case of variations of the parameter V in the range [0.5 2] is taken into consideration through the flat phase around ω_c^d. In what the desired phase margin concerns, the authors in (Tenoutit et al. 2011a) provide a desired maximal overshoot, instead. It is 24% for all variations of the gain V. Using the dependency explored in Section 2.2.2, the overshoot is transformed using

$$\phi_m^d = \sum_{i=0}^{4} \mathbf{x}(i)\mathbf{\Psi}_v^i(k) \approx 51^\circ \tag{2.66}$$

into a desired phase margin. Now, the desired open-loop response can be computed using ϕ_m^d and the expression (2.39). The obtained desired response is given by

$$L_d(s) = \frac{0.37}{s^{1.43}}. \tag{2.67}$$

Due to the fact that the controller can not compensate the delay part of the plant (2.65), a new desired open-loop response is replaced by

$$\tilde{L}_d(s) = \frac{0.37}{s^{1.43}} e^{-sT_d} \tag{2.68}$$

The fractional order 1.43 is approximated using the CRONE method with $N = 7$, $\omega_{min} = 0.001\,rad/s$ and $\omega_{max} = 7.5\,rad/s$. These parameters have to be chosen such that the desired open-loop response is approximately constant around the crossover frequency, despite the time delay given by T_d. Through appropriate values for N, ω_{min} and ω_{max}, the falling phase of the time delay part can be compensated by the CRONE approximation resulting increase of the phase.

The optimized controller parameters are $K_P = 1.13$, $K_I = 0.64$, $K_D = 0.41$ and $n = 0.469$. Based on the CRONE approximation, the controller transfer function is

$$\begin{aligned} K(s) &= \frac{0,4195s^5 + 1,703s^4 + 2,226s^3 + ...}{0,00754s^5 + 1,511s^4 + 0,5995s^3 + ...} \\ &\quad \frac{... + 1,004s^2 + 0,08s + 0,0005}{... + 0,01626s^2 + 0,00003s} \end{aligned} \tag{2.69}$$

Therby, the parameters $N = 3$, $\omega_{min} = 0,001$ rad/s and $\omega_{max} = 2.5$ are used. The controller order is five, which is due to filtering the ideal derivative by a first-order system with the time constant $T_1 = 0.05s$.

The open-loop response with the optimized controller (2.69) and the desired response (2.68) are shown in Figure 2.19. The phase of the function $L_0(s)$ drop off earlier than the phase of $\tilde{L}_d(s)$. This is due to the fact that the controller order is kept small by chosen the CRONE approximation order $N = 3$. Nevertheless, the controller achieves the

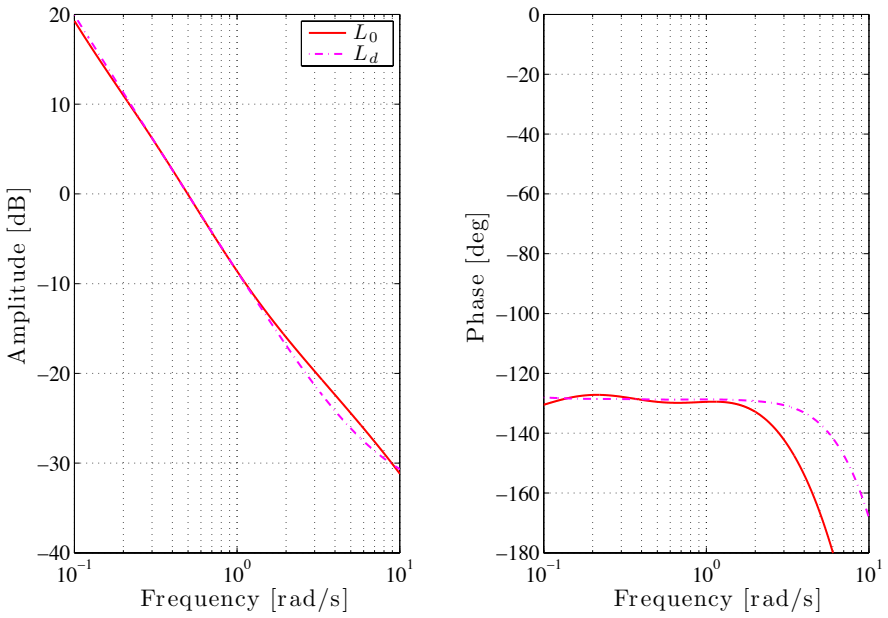

Figure 2.19: Comparison desired L_d and nominal L_0 open-loop response

desired performance. The step response for $V = 0.5$, $V = 1.0$ and $V = 2.0$ shows that the overshoot is almost constant, $23,2\%$, $23,7\%$ and $23,7\%$ respectively.

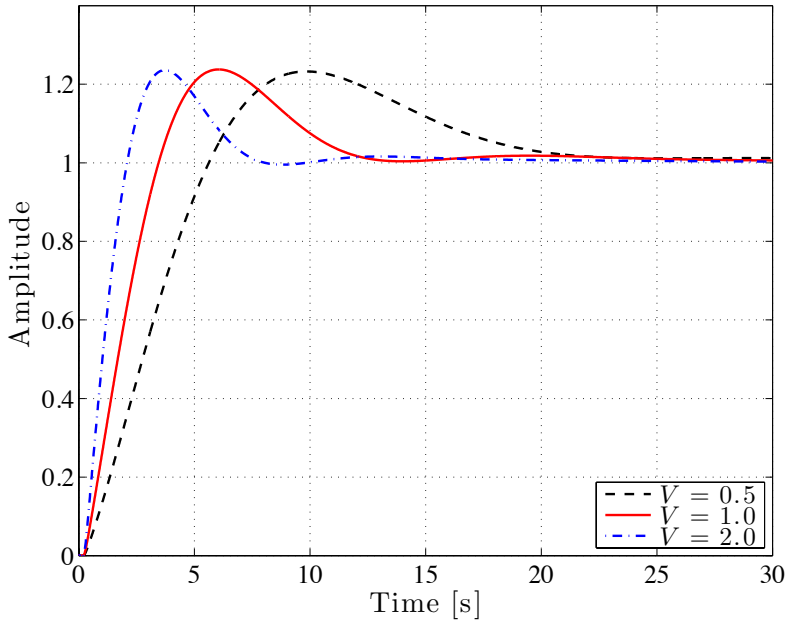

Figure 2.20: Step response with $(PID)^n$ controller, for $V = 0.5$ (dashed black), $V = 1.0$ (red) and $V = 2.0$ (dashdot blue)

This achieved performance is compared with the results obtained in (Tenoutit et al. 2011a) and (Tenoutit et al. 2012). The summary is given by Table 3.1. It shows imprecively that the controller (2.69) achieves a much better robust performance in comparison to the controller in (Tenoutit et al. 2011a).

Table 2.1: Overshoots in (%)

V	0.5	1	2
(Svaricek and Lachhab 2016) $(PID_f)^n$	23.2%	23.7%	23.7%
(Tenoutit et al. 2011a) $(PID_f)^n$	22%	24%	29.5%

As in (Tenoutit et al. 2011a), the proposed method also does not provide any classical PID controller which achieves the required robust performance. The obtained performance with the parameters $K_P = 0,3849$, $K_I = 1,452$, $K_D = 0,0425$ and $T_1 = 0,05$ s is $5,3\%$ for $V = 0,5$, $23,9\%$ for $V = 1,0$ and $49,3\%$ for $V = 2.0$

2.5 Summary

In this chapter, the design and optimization of robust fractional order controllers have been discussed. In Section 2.1, an introduction to this chapter is given. Then in Section 2.2, the robustness requirements have been introduced. Moreover, frequency as well as time domain interpretations have been discussed. Afterwards, the dependency between the phase margin and the overshoot based on a fractional open-loop integrator have been explored. A method was proposed to parametrize this dependency. The formulation of the control problem is presented in Section 2.3. Thereby, two approaches has been used, namely shaping the open-loop response for the $PI^{\alpha}D^{\beta}$ controller and shaping the closed-loop response for the $(PID)^n$ controller. The optimization of the controller parameters have been achieved using the Steepest Descent method. In Section 2.4, a Matlab Tooolbox (FOPID) has been presented dedicated to the computation of the controller parameters. Examples shows that the obtained controllers using the proposed method achieve the required performance. Moreover, a comparison to the fractional controllers presented in the literature reveals that our FOPID-$(PID)^n$ outperforms the other controllers.

3 Optimization of PID Controllers using MPC and LMI

3.1 State of the Art (MPC)

In Section 2, the design and optimization of the fractional controllers $(PID)^n$ and $PI^{\alpha}D^{\beta}$ were considered. It is an extension of the recently developed method (Apkarian and Noll 2006) applied now on fractional controllers. Regarding PID controllers, there exists a huge number of tuning and optimization methods dedicated to this class of controllers. Depending on the plant to be controlled and the control requirements, the control designer has to choose the appropriate method. For example, in case that the order of the plant is low and the requirements, which has to be satisfied are not too severe, heuristic methods can be used. A very prominent method is the Ziegler and Nichols tuning method. For the case that the requirements are severe and have to be strictly satisfied, modern optimization algorithms are necessary, see (Khatibi et al. 2008). In (Sadeghpour et al. 2012), a Toolbox to design robust PID controllers is presented. It is based on the frequency response of the system and convex optimization. The design and optimization of H_2 and H_∞ fixed-structure controllers (gains, $PIDs$, etc) is presented in (Apkarian et al. 2014). The authors present a method based on nonsmooth optimization techniques to optimize the parameters of these controllers. Simulation results are presented.

The goal of this chapter is to explore the possibility of using modern methods to optimize PID controllers. To achieve this task, a novel method dedicated to the optimization of the parameters of this controllers is presented. Moreover, robustness requirements in case of static gain variations are considered. This is taken into consideration through the specification of a desired closed-loop response. Specifically, discrete-time closed-loop response is considered. The control problem formulation is performed using model predictive control. The resulting optimization problem is nonconvex and can not be directly solved. A new method is proposed to transform this problem into a set of linear matrix inequalities. The resulting LMI optimization problem can be solved efficiently using any LMI solver. The controller parameters are decision variables of the optimization problem which are now optimized instead of tuned, in contrast to classical PID tuning methods.

The main idea of this work is to optimize the parameters of classical PID controllers based on MPC appraoch and LMIs. The goal is not to provide a new solution to robust MPC problems. There exists a high number of papers and contributions which propose several appraoches to solve robust MPC using LMIs. A very prominent paper in this field is (Kothare et al. 1996), which describes in details the application of LMIs to solve MPC problems. An approach is proposed to compute a MPC controller for uncertain and time varying plant taking into consideration input and output constrains. The method is based on defining the controller to be a state-feedback, which is computed at each sampling time by solving a LMI problem. In our work, we are not solving LMIs online and the controller which is considered is a PID controller.

Additionally to the work (Kothare et al. 1996), it exists many other contribution re-

lated to robust MPC controller. In (Fukushima and Bitmead 2003), the authors present an approach for uncertain systems with bounded disturbance to optimize MPC control. Simulation results are presented with a sampling time $0.1\,sec.$ In (Mayne et al. 2006), a method is presented based on luenberger state estimator and a feedback control for syytems with uncertain states and bounded disturbance. A simple double integrator is used as an example to show the performance of the method.

MPC is a powerful method to design controllers imposed to input, output or states contraints. MPC is well known to be a receding horizing method. The optimization problem is solved at each iteration, but only the first input value is applied to the plant. The optimization problem consists of minimizing a constrained objective. Two types of constraints are considered, namely soft and hard contraints. Hard constraints have to be strictly satisfied. Soft constraints on the other hand are only partially satisfied. The first type of constraints is characterized with equalities, inequalities or both. The second type is incorporated into the cost function using a penality term.

In case of hard constraints, there exists no closed form for the solution. Iterative and modern algorithms are required to solve the constrained optimization problem. For instance, Sequential Quadratic Programmig (SQP) or Interior Point Methods (IPM) can be used. The main obstacle in applying iterative algorithms is the computational power required to solve the problem in real-time. Due to this difficulty, MPC has been reserved to plants with slow dynamic. Many authors are working on methods to make MPC more practical.

In (Shah and Engell 2011), the authors propose a method to tune the parameters of the MPC optimization problem. Examples are shown with a sampling time of $T_s = 1s$ and $T_s = 10s$. In (Kapernick and Graichen 2014), a model predictive control software named GRAMPC is presented. It deals with the computation of the optimal control strategy for nonlinear systems. To test the practical aspect of this software, simulation results based on dSPACE are provided. In (Witt et al. 2007), the concept of approximate predictive control is used to design a controller for a 3-DOF Helicopter. The control method is based on a linearization of a neural network model of the plant and a generalized predictive control. In (Wang and Boyd 2010), an approach is proposed to render the MPC with hard constraints practical to plants with fast dynamic. The idea is based on exploring the structure of the optimization problem. Specifically, an efficient computation of the Newton step is proposed. The authors propose another simplification, which consists in fixing the maximal number of the Newton iterations.

As it is mentioned in (Bemporad 2006), the contributions related to solving the MPC problem can be classified into two methods, namely online (real-time) and offline approaches. For the first case, the idea is to solve the MPC related Quadratic Problem (QP) in real-time. This is possible for slow process with a sufficient high computing power. Moreover, by exploiting the problem structure an approximated solution can be computed for fast processes. In case that a real-time implementation of MPC is not pos-

sible, the second approach can be considered. It is based on solving the optimization problem offline. Afterwards, the optimal solution is implemented with the help of look-up tables. A prominent work in this field is the multi-parametric method proposed by (Bemporad et al. 2002). It consists of solving the MPC problem by considering the states vector as an additional parameter. The real-time implementation consists of evaluating a look-up table.

The proposed approach in this section belongs to the second class. We are solving the MPC problem using a LMI formulation of the optimization problem, offline. Moreover, the real-time implementation is not based on look-up tables. Instead, we are approximating the optimal solution using a fixed-structure controller, namely PID. Our motivation is to generate the obtained optimal input trajectory using this controller. In real-time, the PID controller provides this trajectory. In this section, the developed method is presented in details.

3.2 Control Performance Specifications

In this section, the performance specifications considered in this chapter are presented. These are given in terms of a desired closed-loop step response. Before going further with these specifications, we want to recall the definition of the plant considered. In order to remind the reader, this class of plants is defined as follows

$$G(s) = V \cdot G_0(s) \tag{3.1}$$

with $G_0(s)$ denotes the nominal plant and V represents the uncertain parameter in $[V_{min}\ V_{max}]$, respectively. Unlike in Section 2, we are here interested in designing a robust controller K that provides a step response without any overshoot. This has to be guaranteed for all variations of the static gain V. Moreover, the resulting crossover frequency of the nominal open-loop transfer function $L_0(s) = K(s)G_0(s)$ is approximately equal to a desired frequency ω_c^d. Formally, these requirements are as follows

$$|\underbrace{K(j\omega_c^d)G_0(j\omega_c^d)}_{L_0(j\omega_c^d)}|_{dB} = 0 \quad \text{and} \quad \max_{t=0\to\infty} y(t,V) = 1, \tag{3.2}$$

with $L_0(j\omega)$ is the nominal open-loop transfer function and $y(t,V)$ consists the unit step response of the closed-loop transfer function, which depends on the parameter V.

In Section 2.2, integrators fail to represent the robustness specifications (2.3) to (2.5). This was due to the specification of a desired phase margin φ_m^d different from 90°. In this section, the application of integrators to formulate the robustness specifications given by the equations (3.2) seems to be convenient. To make this clear, consider the closed-loop system shown in Figure 3.1. Thereby, R, E and Y denote the reference, error and output signals, respectively. Assume that the nominal open-loop transfer function

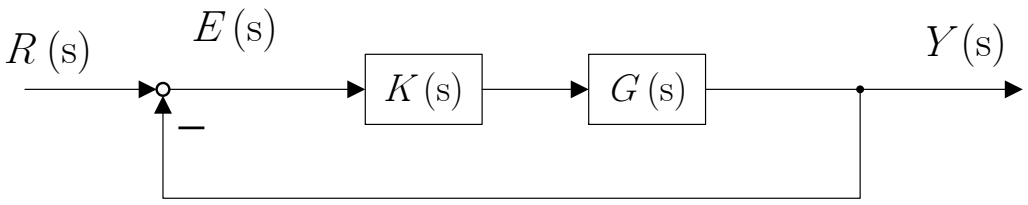

Figure 3.1: Feedback control structure

$L_0(s) = G_0(s)K(s)$ provides the desired response given by the integrator

$$L_d(s) = \frac{\omega_c^d}{s}. \tag{3.3}$$

Thereby, ω_c^d denotes the desired crossover frequency. The nominal closed-loop transfer function is given by

$$T_0(s) = \frac{L_0(s)}{1 + L_0(s)}. \tag{3.4}$$

Assume that the nominal response $L_0(s)$ is equal to the desired response $L_d(s)$. In this case, the following

$$T_0(s) = \frac{\omega_c^d}{s + \omega_c^d} = \frac{1}{\underbrace{\frac{1}{\omega_c^d}}_{\tau}\, s + 1} \tag{3.5}$$

is obtained. The transfer function $T_0(s)$ has a low-pass character with the time constant $\frac{1}{\omega_c^d}$. In this case, the overshoot of the step response is zero. It is clear that requiring the open-loop function $L_0(s)$ to have the response provided by the integrator (3.3) results in a closed-loop system with the time constant τ and no overshoot. As it is mentioned here, the goal is not only to provide this performance for the nominal case, but for all values of the uncertain parameter V.

Now consider the parameter dependent open-loop function $L(s) = V \cdot K(s)G_0(s)$, which describes the open-loop response in case of parameter variations. Due to the assumption above, we can write this function as follows

$$L(s) = V \cdot L_d(s) = V \cdot \frac{\omega_c^d}{s}. \tag{3.6}$$

In this case, the open-loop function turns out to be a parameter dependent integrator. Computing the closed-loop transfer function using the function (3.6) leads to

$$T(s) = \frac{V \cdot L_d(s)}{1 + V \cdot L_d(s)} = \frac{V \cdot \omega_d}{s + V \cdot \omega_d} = \frac{1}{\frac{1}{V \cdot \omega_c^d} s + 1}. \tag{3.7}$$

The transfer function (3.7) describes the response of the closed-loop systen in case of parameter variations. $T(s)$ defines a parameter dependent low-pass function. Its time constant is given by $\tau_V = \frac{1}{V \cdot \omega_c}$. In case that the parameter V varies, this results in a different bandwidth $V \cdot \omega_c^d$ of the closed-loop function. The overshoot in case of a step response will be zero for all variations. The required robustness is achieved thanks to the use of the integrator (3.3).

Example 3.1

For a better understanding of the integral formulation of the robustness requirements, consider the following open-loop function

$$L(s) = V \cdot \frac{K}{s} \qquad \text{with} \quad K = 1\,rad/s \quad \text{and} \quad V \in [0.5\ 1.5]\,. \tag{3.8}$$

The closed-loop function depending on the uncertain parameter V is given by

$$T(s) = \frac{1}{\frac{1}{V \cdot K} \cdot s + 1}. \tag{3.9}$$

The minimal and maximal time constant of this function are given by $\tau_{min} = \frac{1}{V_{max} \cdot K}$ and $\tau_{max} = \frac{1}{V_{min} \cdot K}$. Moreover, the response of the closed-loop system is bounded in the frequency domain by

$$T_{min} = T(s, V = V_{min}) = \frac{1}{\tau_{max} \cdot s + 1} \tag{3.10}$$

and

$$T_{max} = T(s, V = V_{max}) = \frac{1}{\tau_{min} \cdot s + 1}. \tag{3.11}$$

The parameter variations cause a shift of the bandwidth ω_b of the low-pass transfer function, see Figure 3.2. In the nominal case, it is $\omega_b = \omega_0$. In the uncertain case, it is $\omega_b = V \cdot \omega_0$. Figure 3.2 shows also the step response for V_{min}, V_0 and V_{max}. In all cases the overshoot is zero. The required robustness is achieved.

Remark. *The definition of the desired closed-loop response is given in the continuous time domain. The proposed approach in this chapter is a discrete-time method. For this reason, a discretization of the transfer function (3.5) is performed.*

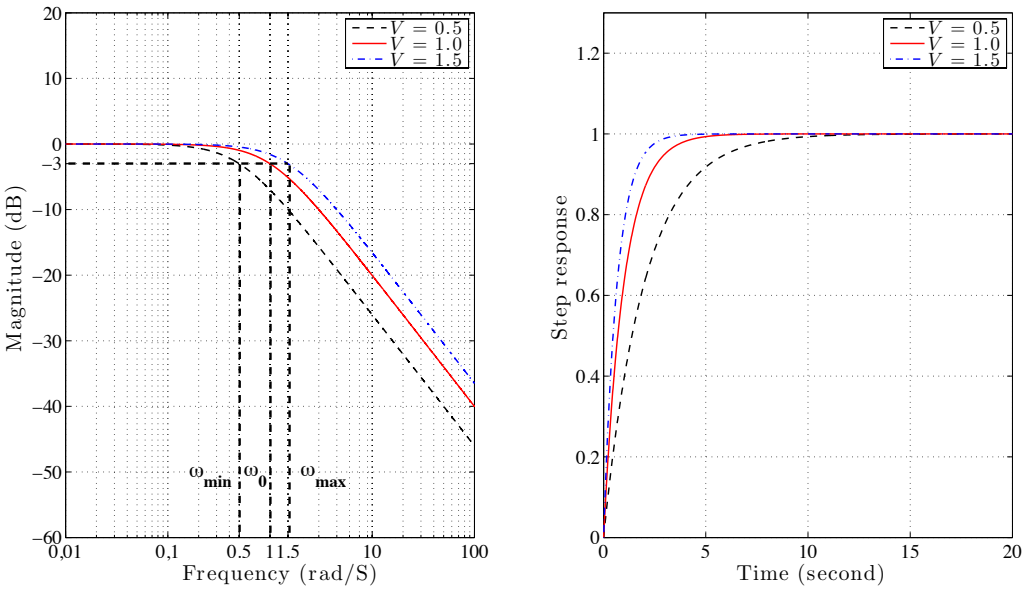

Figure 3.2: Step response (right) and Bode magnitude (left) of the transfer function $T(s)$ for $V = 0.5$, $V = 1.0$ and $V = 1.5$

3.3 Formulation of the Control Problem

In this section, the formulation of the PID control problem is presented. It is based on Model Predictive Control (MPC). First, the open-loop MPC optimization problem is briefly discussed. Moreover, a LMI solution to this problem is presented. Then, the closed-loop MPC problem is discussed. The related optimization problem is given in terms of the control input and controller parameters. The resulting optimization consists in solving a set of Bilinear Matrix Inequalities (BMI). This type of inequalities is well known to be nonconvex and is hard to solve. This fact is discussed at the end of this section.

3.3.1 Formulation of the Open-Loop LMI Problem

MPC is a receding horizing strategy. The related constrained optimization problem is solved at each sampling time. The first entry of the control sequence is applied to the system and the procedure is repeated again. This work considers the controller design for SISO as well as for MIMO systems subject to input and output constraints. The controller structure is predefined, namely PID_f. As the MPC approach is a discrete-time method, the discrete-time version of the PID_f controller is considered here.
For a better understanding of the MPC main idea, a discrete-time model of the electronic throttle given by the expression (B.48) in Appendix B is used. Thereby, the nominal case

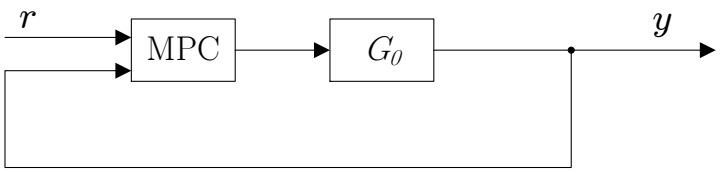

Figure 3.3: MPC closed-loop structure

with $V = 1$ is considered. It is given by the following state-space realization

$$G_0 : \left\{ \begin{array}{rcl} \mathbf{x}(k+1) & = & \mathbf{A}\,\mathbf{x}(k) + \mathbf{b}\,u(k) \\ y(k) & = & \mathbf{c}_0^T\,\mathbf{x}(k) \end{array} \right. \tag{3.12}$$

with $\mathbf{A} \in \mathbb{R}^{2\times 2}$, $\mathbf{b} \in \mathbb{R}^{2\times 1}$ and $\mathbf{c}_0^T \in \mathbb{R}^{1\times 2}$. The signals $\mathbf{x}(k)$, $u(k)$ and $y(k)$ are the discrete-time state vector, input and output, respectively. The general uncertain case is covered by introducing the uncertain output vector $\mathbf{c}^T = V \cdot \mathbf{c}_0^T$.

The related MPC closed-loop structure is presented in Figure 3.3. The signal $r(k)$ constitutes the reference or desired trajectory. Based on the current and the predicted values of the output $y(k)$, the MPC strategy computes the optimal input sequence with respect to a predefined cost function $J(u)$. Figure 3.4 shows a time sequence of the output signal

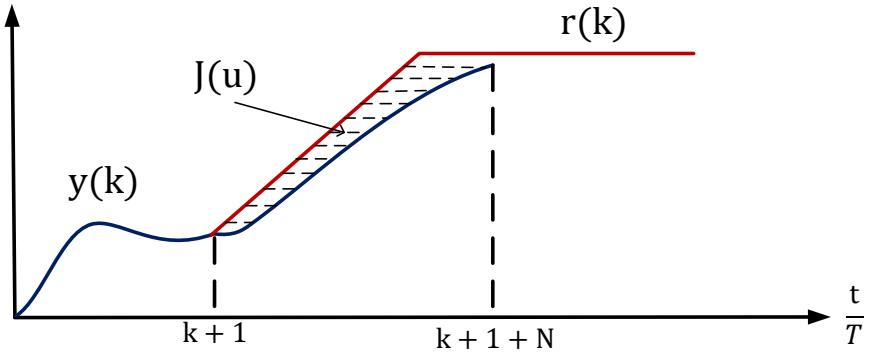

Figure 3.4: Prediction horizing N of MPC with reference trajectory $r(k)$ and output $y(k)$

$y(k)$ and its reference $r(k)$. At the sampling time $k+1$, the optimal control sequence with

respect to $J(u)$ is computed. For example, the mean square root of the error $r(k) - y(k)$ can be used to define the cost function. It is worth to mention here that the computation of the optimal input is based on predicted values beginning from the sampling time $k+1$ until $k+1+N$. The number N of discrete-time steps into the future is known as the prediction horizon. Mostly, the value of N is chosen between 5 and 15 depending on the order of the plant and the dynamic of the reference signal $r(k)$.

To formulate the MPC optimization problem, a forward model of the state-space system (3.12) is needed. Following the formulation (Maciejowski 2002, p. 54-55), the plant output at successive time instant k is given by

$$\begin{aligned}
y(k) &= \mathbf{c}_0^T\ \mathbf{x}(k) \\
y(k+1) &= \mathbf{c}_0^T\mathbf{A}\ \mathbf{x}(k) + \mathbf{c}_0^T\mathbf{b}\ u(k) \\
y(k+2) &= \mathbf{c}_0^T\mathbf{A}^2\ \mathbf{x}(k) + \mathbf{c}_0^T\mathbf{A}\mathbf{b}\ u(k) + \mathbf{c}_0^T\mathbf{b}\ u(k+1) \\
&\vdots \qquad\qquad \vdots \\
y(k-1+N) &= \mathbf{c}_0^T\mathbf{A}^{N-1}\ \mathbf{x}(k) + \mathbf{c}_0^T\mathbf{A}^{N-2}\mathbf{b}\ u(k) + \quad \cdots \\
&\qquad\qquad \cdots + \quad \mathbf{c}_0^T\mathbf{b}\ u(k+N-2). \qquad (3.13)
\end{aligned}$$

Now introducing the input and output vectors

$$\mathbf{u} = \begin{bmatrix} u(k) \\ u(k+1) \\ \vdots \\ u(k-1+N) \end{bmatrix} \quad \text{and} \quad \mathbf{y} = \begin{bmatrix} y(k) \\ y(k+1) \\ \vdots \\ y(k-1+N) \end{bmatrix},$$

the expression (3.13) can be written as

$$\mathbf{y} = \mathbf{\Omega}\ \mathbf{u} + \mathbf{\Psi}, \qquad (3.14)$$

with

$$\mathbf{\Omega} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ \mathbf{c}_0^T\mathbf{b} & 0 & 0 & \cdots & 0 \\ \mathbf{c}_0^T\mathbf{A}\mathbf{b} & \mathbf{c}_0^T\mathbf{b} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{c}_0^T\mathbf{A}^{N-2}\mathbf{b} & \mathbf{c}_0^T\mathbf{A}^{N-1}\mathbf{b} & \cdots & \mathbf{c}_0^T\mathbf{b} & 0 \end{bmatrix} \quad \text{and} \quad \mathbf{\Psi} = \begin{bmatrix} \mathbf{c}_0^T \\ \mathbf{c}_0^T\mathbf{A} \\ \mathbf{c}_0^T\mathbf{A}^2 \\ \vdots \\ \mathbf{c}_0^T\mathbf{A}^{N-1} \end{bmatrix} \mathbf{x}(k). \qquad (3.15)$$

The goal is to solve the following optimization problem

$$\underset{\mathbf{u}}{\text{minimize}} \quad \|\mathbf{y}_d - \mathbf{y}\|_2^2 \qquad (3.16)$$

with $\mathbf{y}_d$ denotes the desired response in vector notation, which is equal to the reference $\mathbf{y}_d = \mathbf{r}$.

Problem (3.16) consists of minimizing the square of the 2-norm of the error vector $\mathbf{e} = \mathbf{y}_d - \mathbf{y}$. The optimization parameter is the input vector $\mathbf{u}$. This optimization represents an unconstrained least square problem. The solution to this problem has a closed form. To solve this problem consider the equivalent formulation

$$\underset{\mathbf{u}}{\text{minimize}} \quad (\mathbf{y}_d - \mathbf{y})^T(\mathbf{y}_d - \mathbf{y}). \tag{3.17}$$

After substituting Equation (3.14) in the minimization problem (3.17), the following

$$\underset{\mathbf{u}}{\text{minimize}} \quad f(\mathbf{u}) = \mathbf{y}_d^T\mathbf{y}_d - 2\mathbf{y}_d^T\mathbf{\Omega u} - 2\mathbf{y}_d^T\mathbf{\Psi} + (\mathbf{\Omega u} + \mathbf{\Psi})^T(\mathbf{\Omega u} + \mathbf{\Psi}) \tag{3.18}$$

is obtained. The optimal vector $\mathbf{u}_{opt}$ can be computed by setting the derivative of the function $f(\mathbf{u})$ with respect to $\mathbf{u}$ to zero. The optimal solution is as follows

$$\mathbf{u}_{opt} = (\mathbf{\Omega}^T\mathbf{\Omega})^{-1}\mathbf{\Omega}^T(\mathbf{y}_d - \mathbf{\Psi}). \tag{3.19}$$

The optimal value is given by the first entry $\mathbf{u}_{opt}(k)$ of the optimal vector $\mathbf{u}_{opt}$.

One of the main advantages of the MPC approach is the capability of constraints handling. It means that one can impose constraints on the input, output or states. For this purpose, the general constrained MPC problem

$$\underset{\mathbf{u}\in\tilde{\mathbf{U}},\mathbf{y}\in\tilde{\mathbf{Y}}}{\text{minimize}} \quad \|\mathbf{y}_d - \mathbf{y}\|_2^2 \tag{3.20}$$

is considered. The sets $\tilde{\mathbf{U}}$ and $\tilde{\mathbf{Y}}$ describe constraints on the input and output vectors, which are given as following

$$\tilde{\mathbf{U}} := \{\mathbf{u} \in \mathbb{R}^N, \mathbf{E}_u\mathbf{u} \leq \mathbf{f}_u\} \tag{3.21}$$

and

$$\tilde{\mathbf{Y}} := \{\mathbf{y} \in \mathbb{R}^N, \mathbf{E}_y\mathbf{y} \leq \mathbf{f}_y\}. \tag{3.22}$$

Here $\mathbf{E}_u$ and $\mathbf{E}_y$ are predefined matrices describing the structure of the constraints. The vectors $\mathbf{f}_u$ and $\mathbf{f}_y$ represent the related upper and lower bounds.

Before presenting the main result, problem (3.20) is replaced with the following equivalent problem

Problem 3.1 :

$$\begin{aligned} &\underset{\mathbf{u}\in\tilde{\mathbf{U}},\mathbf{y}\in\tilde{\mathbf{Y}}}{\text{minimize}} && \gamma \\ &\text{subject to} && \|\mathbf{y}_d - \mathbf{y}\|_2^2 < \gamma \end{aligned} \tag{3.23}$$

with the slack variable γ. To solve the above problem, we apply the Schur complement given by the expressions (B.7) and (B.8) in Appendix B. For this purpose, the constraint in Problem 3.1 is rewritten as follows

$$-(\mathbf{y}_d - \mathbf{y})^T (-\mathbf{I}_N) (\mathbf{y}_d - \mathbf{y}) + (-\gamma) < 0. \tag{3.24}$$

Applying the Schur complement on (3.24) leads to the following LMI

$$\begin{bmatrix} -\gamma & (\mathbf{y}_d - \mathbf{y})^T \\ (\mathbf{y}_d - \mathbf{y}) & -\mathbf{I}_N \end{bmatrix} < 0 \tag{3.25}$$

or

$$\begin{bmatrix} -\gamma & (\mathbf{y}_d - \mathbf{\Omega u})^T \\ (\mathbf{y}_d - \mathbf{\Omega u}) & -\mathbf{I}_N \end{bmatrix} < 0 \tag{3.26}$$

by substituting $\mathbf{\Omega u}$ for $\mathbf{y}$, since we assume that $\mathbf{x}(0) = 0$. $\mathbf{I}_N$ represents the identity matrix of dimension N.

Putting now all constraints together, the MPC optimization problem can be summarized as follows

Problem 3.2 :

$$\begin{aligned} &\underset{\mathbf{u}}{\text{minimize}} && \gamma \\ &\text{subject to} && \begin{bmatrix} -\gamma & (\mathbf{y}_d - \mathbf{\Omega u})^T \\ (\mathbf{y}_d - \mathbf{\Omega u}) & -\mathbf{I}_N \end{bmatrix} < 0, \\ & && \mathbf{E}_u\mathbf{u} \leq \mathbf{f}_u, \;\; \mathbf{E}_y(\mathbf{\Omega u}) \leq \mathbf{f}_y. \end{aligned} \tag{3.27}$$

The LMI optimization problem (3.27) provides the optimal input vector $\mathbf{u}_{opt}$ at time k. Only the first component of this vector is applied to the system. In the next step, the whole procedure is repeated again.

3.3.2 Optimal Input Trajectory for the Electronic Throttle

To solve the LMI problem (3.27), the modeling and optimization Toolbox YALMIP (Lofberg 2004) in combination with the SeDuMi solver (Labit et al. 2002) are used. At this point, it is worth to mention that we are solving an optimization problem numerically. This means that the optimization problem should be well conditioned. The optimal

solution denoted by $\mathbf{u}_{opt}$ should be numerically robust and not exceed an upper bound. To achieve this goal, the following LMI

$$\begin{bmatrix} -\sigma & (\mathbf{u})^T \\ (\mathbf{u}) & -\mathbf{I}_N \end{bmatrix} < 0 \tag{3.28}$$

is added to the set of LMIs (3.27). Then, the objective $\gamma + \sigma$ is minimized instead of minimizing γ. The LMI (3.28) can be deduced from the following condition

$$\|\mathbf{u}\|_2^2 = \mathbf{u}^T\mathbf{u} < \sigma, \tag{3.29}$$

by applying the Schur complement.

For a better understanding of the LMI-Problem 3.2, the computation of an optimal input trajectory for the electronic throttle based on the nominal discrete-time model (3.12) is considered. The problem under consideration consists of computing the optimal input, which brings the output of the electronic throttle from 0° to 10° under input constraints. Thereby, the maximal value of the input u is given by U_{max}. Three cases are considered, namely $U_{max} = 1$, $U_{max} = 2$ and $U_{max} = 4$. For this purpose, the following problem definition

$$\mathbf{y}_d = \begin{bmatrix} 0 \\ 10 \\ 10 \\ 10 \\ \vdots \\ 10 \end{bmatrix}, \ \mathbf{f}_u = U_{max} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \text{ and } \mathbf{E}_u = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ -1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & -1 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & -1 \end{bmatrix}$$

is introduced. The vector $\mathbf{y}_d \in \mathbb{R}^{201\times 1}$ denotes the desired response given in terms of a step. The vector $\mathbf{f}_u \in \mathbb{R}^{402\times 1}$ and the matrix $\mathbf{E}_u \in \mathbb{R}^{402\times 201}$ describes the input constraint. The matrix $\mathbf{\Omega} \in \mathbb{R}^{201\times 201}$ is given by the expression (3.15).

After defining the problem matrices $\mathbf{y}_d$, $\mathbf{f}_u$, $\mathbf{E}_u$ and $\mathbf{\Omega}$, the optimization is set. Now using the LMI solver (Labit et al. 2002), the optimization problem (3.27) is solved. Figure 3.5 shows time plots of the obtained optimal input $u_{opt}(k)$ and the related optimal output $y_{opt}(k)$ for all three cases.

The constraint imposed by U_{max} on the input trajectory is satisfied. The optimal input sequence $u_{opt}(k)$ is below U_{max}. Moreover, applying the optimal input trajectory results in the output trajectory shown on the left hand side of Figure 3.5. The rise time of the optimal trajectory depends on the maximal input value U_{max}. Note at this point that the optimal input sequence $u_{opt}(k)$ is applied as a feedforward signal. This corresponds to an open-loop structure. The closed-loop problem is considered in the next section.

In some cases, it is desired to constrain, additionally to the input or output signal, the

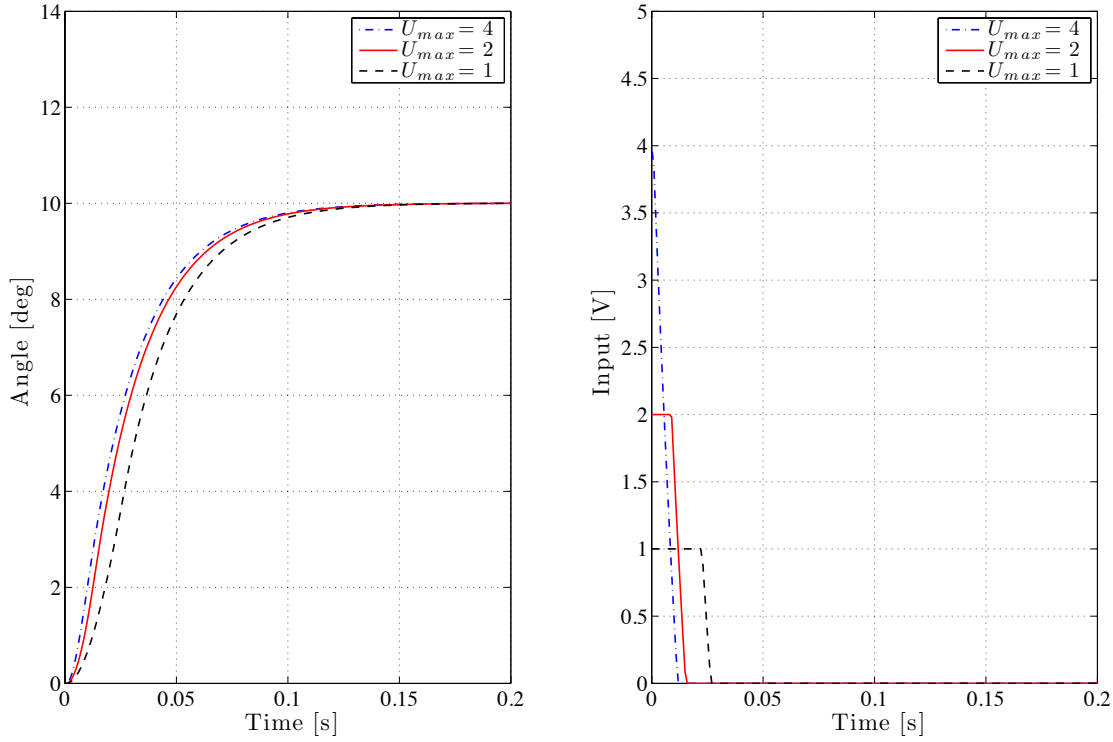

Figure 3.5: Optimal output trajectory (left) and optimal input trajectory (right) for $U_{max} = 2$ (dashed black), $U_{max} = 4$ (red) and $U_{max} = 8$ (dashdot blue)

related first or second order derivative or both. This can be achieved by considering the definition of the discrete-time derivative. This is given by $y(k) - y(k-1)$ for the first order derivative $\delta y(k)$. For the second order derivative $\delta^2 y$, its discrete-time implementation is given by $y(k) - 2y(k-1) + y(k-2)$. The sampling time is omitted for simplicity. Now assume that the discrete-time difference $\delta y(k)$ is constrained by a given lower and upper bound, Δy_{min} and Δy_{max}, respectively. This can be written as follows

$$\Delta y_{min} \leq (y(k+1) - y(k)) \leq \Delta y_{max} \tag{3.30}$$

or

$$\begin{aligned} (y(k+1) - y(k)) &\leq \Delta y_{max} \\ (y(k) - y(k+1)) &\leq -\Delta y_{min}. \end{aligned} \tag{3.31}$$

Now using the following matrices

$$\mathbf{E}_y = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 1 & -1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & -1 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & -1 \end{bmatrix} \quad \text{and} \quad \mathbf{f}_y = \begin{bmatrix} \Delta y_{max} \\ -\Delta y_{min} \\ \Delta y_{max} \\ -\Delta y_{min} \\ \vdots \\ -\Delta y_{min} \end{bmatrix},$$

the inequalities (3.31) can be transformed into the form (3.22) with $\mathbf{f}_y \in \mathbb{R}^{600\times 1}$ and $\mathbf{E}_y \in \mathbb{R}^{600\times 301}$. After building the constraint matrices, the optimization problem (3.27) is solved. Three cases are considered, namely $\Delta y_{max} = 0.10$, $\Delta y_{max} = 0.12$ and $\Delta y_{max} = 0.14$.

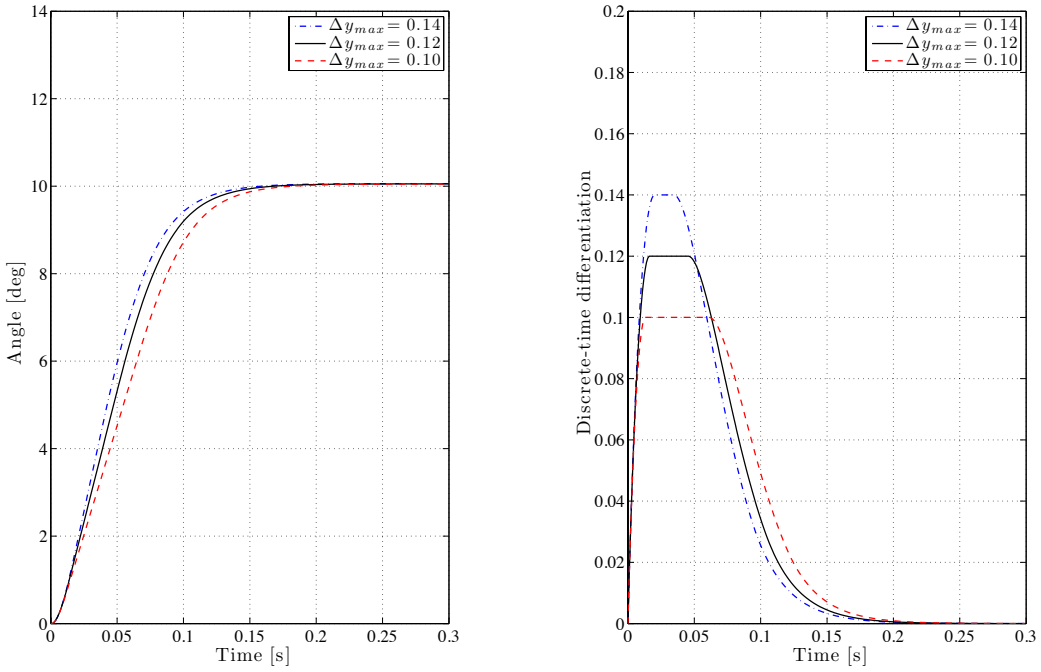

Figure 3.6: Optimal output trajectory (left) and optimal input trajectory (right) for $\Delta y_{max} = 0.10$, $\Delta y_{max} = 0.12$ and $\Delta y_{max} = 0.14$

The results are given by Figure 3.6. It shows that the constraint on the first order difference $\delta y(k)$ is achieved. The optimal output trajectory is also presented. The rise time depends on the maximal value Δy_{max}. Whether constraining the input or the output, the set of LMIs (3.27) is very convenient to be used. In all cases, the optimal input subject to the imposed constraints is achieved. Moreover, the lists of constraints that can be used or can be encountered using the formulation proposed here is very general.

Additionally to constraining the velocity, one can constrain the acceleration, jerk or other linear constraints.

3.3.3 Formulation of the Closed-Loop BMI Problem

The optimization problem (3.27) represents the open-loop MPC problem. It means that the optimal input trajectory $u_{opt}(k)$ is directly applied to the plant. Its dynamic possibilities are much more reduced than in the case, where the parameters of a controller can be also adjusted. This latter case is the object of this section by considering the closed-loop MPC problem. In this case, the optimal input $u_{opt}(k)$ is generated through a LTI dynamic discrete-time controller K. The optimization problem (3.27) is extended to include the controller parameters. In the case of PID_f controllers, these are given by K_P, K_I and K_D.

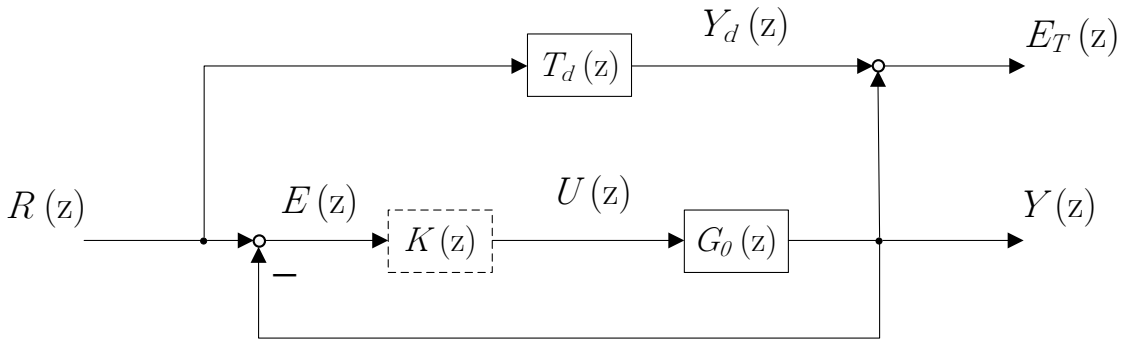

Figure 3.7: Discrete-time model following control structure with reference model $T_d(z)$, controller $K(z)$ and nominal plant $G_0(z)$

Figure 3.7 shows the considered closed-loop structure used to optimize the controller. Thereby, R, E, U, Y, Y_d and E_T are the reference, output error, control input, plant output , desired output and the error between the desired output and the plant output, respectively.

$T_d(z)$ is the discrete-time reference model, which generates the desired response. Based on the discussion in Section 3.2, this model is defined using a given sampling time T_s as follows

$$T_d(z) = \frac{1 - e^{-T_s\omega_c^d}}{z - e^{-T_s\omega_c^d}} \tag{3.32}$$

through the discretization of the continuous-time transfer function given by the expression (3.5). For this task, Zero Order Hold (ZOH) method is used, see (Unbehauen 2007, p.

121). Now consider a given state-space realization of the system $T_d(z)$ as follows

$$T_d : \begin{cases} x_d(k+1) &= a_d\, x_d(k) + b_d\, r(k) \\ y_d(k) &= c_d\, x_d(k). \end{cases} \tag{3.33}$$

The output of this reference model at successive time instant k is given by

$$\begin{aligned} y_d(k) &= c_d\ x_d(k) \\ y_d(k+1) &= c_d a_d\ x_d(k) + c_d b_d\ r(k) \\ y_d(k+2) &= c_d a_d^2\ x_d(k) + c_d a_d b_d\ r(k) + c_d b_d\ r(k+1) \\ &\vdots \\ y_d(k-1+N) &= c_d a_d^{N-1}\ x(k) + c_d a_d^{N-2} b_d\ r(k) + \quad \cdots \\ &\qquad \cdots + \quad c_d b_d\ r(k+N-2). \end{aligned} \tag{3.34}$$

The vector representation of the desired response is given by

$$\underbrace{\begin{bmatrix} y_d(k) \\ y_d(k+1) \\ \vdots \\ y_d(k-1+N) \end{bmatrix}}_{\mathbf{y}_d} = \underbrace{\begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ c_d b_d & 0 & 0 & \cdots & 0 \\ c_d a_d b_d & c_d b_d & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 \\ c_d a_d^{N-2} b_d & c_d a_d^{N-1} b_d & \cdots & c_d b_d & 0 \end{bmatrix}}_{\mathbf{\Omega}_d} \underbrace{\begin{bmatrix} r(k) \\ r(k+1) \\ \vdots \\ r(k-1+N) \end{bmatrix}}_{\mathbf{r}}$$

for a zero initial condition $x_d(0) = 0$. $\mathbf{\Omega}_d$ and $\mathbf{r}$ denote the transition matrix and the reference in vector notation. Substituting $\mathbf{\Omega}_d\mathbf{r}$ for the desired response $\mathbf{y}_d$, the previous optimization problem (3.27) can be replaced by the following problem

Problem 3.3 :

$$\begin{aligned} &\underset{\mathbf{u}}{\text{minimize}} \quad \gamma \\ &\text{subject to} \quad \begin{bmatrix} -\gamma & (\mathbf{\Omega}_d\mathbf{r} - \mathbf{\Omega}\mathbf{u})^T \\ (\mathbf{\Omega}_d\mathbf{r} - \mathbf{\Omega}\mathbf{u}) & -\mathbf{I}_N \end{bmatrix} < 0, \\ &\qquad\qquad \mathbf{E}_u\mathbf{u} \le \mathbf{f}_u, \ \ \mathbf{E}_y(\mathbf{\Omega}\mathbf{u}) \le \mathbf{f}_y. \end{aligned} \tag{3.35}$$

Now the main idea of this section is presented. Instead of seeking the optimal input $u_{opt}(k)$ that provides the desired response $\mathbf{y}_d$, the goal is to find a controller $K(z)$ which

generates $u_{opt}(k)$. Specifically, a discrete-time PID controller given by

$$K(z) = K_P + K_I \underbrace{\frac{1}{1-z^{-1}}}_{F_I(z)} + K_D \underbrace{1-z^{-1}}_{F_D(z)} \tag{3.36}$$

is considered for this task. The transfer functions $F_I(z)$ and $F_D(z)$ are the discrete-time integrator and ideal differentiation, see (Unbehauen 2007, p. 127). For simplicity, the sampling time is omitted. Considering Figure 3.7, the output of the controller is as follows

$$U(z) = \underbrace{K_P E(z)}_{U_P(z)} + \underbrace{K_I \left(\frac{1}{1-z^{-1}} \right) E(z)}_{U_I(z)} + \underbrace{K_D \left(1-z^{-1}\right) E(z)}_{U_D(z)} \tag{3.37}$$

where U_P, U_I and U_D denote the output P, I and D-component of the PID controller. Using the sampling step k, we get the following update-formula

$$u(k+1) = K_P e(k+1) + K_I u_I(k+1) + K_D(e(k+1) - e(k)) \tag{3.38}$$

with $u_I(k+1) = (e(k+1) + u_I(k))$. The expression above is linear in the error $e(k)$ and can be formulated in vector notation. This can be achieved as follows

$$\mathbf{u} = K_P\, \mathbf{e} + K_I \mathbf{\Omega}_I\, \mathbf{e} + K_D \mathbf{\Omega}_D\, \mathbf{e} \tag{3.39}$$

with the error vector

$$\mathbf{e} = \begin{bmatrix} e(k) \\ e(k+1) \\ \vdots \\ e(k-1+N) \end{bmatrix}.$$

The matrices $\mathbf{\Omega}_D$ and $\mathbf{\Omega}_I$ consist the derivative and integral operators in matrix form. These are defined as follows

$$\underbrace{\begin{bmatrix} e(k) \\ e(k+1) - e(k) \\ e(k+2) - e(k-1) \\ \vdots \\ e(k-1+N) - e(k-2+N) \end{bmatrix}}_{\mathbf{e}_D} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ -1 & 1 & 0 & \cdots & 0 \\ 0 & -1 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 \\ 0 & 0 & \cdots & -1 & 1 \end{bmatrix}}_{\mathbf{\Omega}_D} \underbrace{\begin{bmatrix} e(k) \\ e(k+1) \\ \vdots \\ e(k-1+N) \end{bmatrix}}_{\mathbf{e}} \tag{3.40}$$

and

$$\underbrace{\begin{bmatrix} e(k) \\ e(k)+e(k+1) \\ \vdots \\ \sum_{k=0}^{N-1} e(k) \end{bmatrix}}_{\mathbf{e}_I} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ 1 & 1 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 \\ 1 & 1 & \cdots & 1 & 1 \end{bmatrix}}_{\mathbf{\Omega}_I} \underbrace{\begin{bmatrix} e(k) \\ e(k+1) \\ \vdots \\ e(k-1+N) \end{bmatrix}}_{\mathbf{e}} \tag{3.41}$$

with $\mathbf{e}_D$ and $\mathbf{e}_I$ consist the discrete-time derivative and integral of the error vector $\mathbf{e}$ where we assume that $e(k-1)=0$. Now replacing $\mathbf{e}$ by $(\mathbf{r}-\mathbf{y})$ on the right hand side of the expression (3.39) results in the following equality

$$\mathbf{u} = K_P(\mathbf{r}-\mathbf{y}) + K_I\mathbf{\Omega}_I(\mathbf{r}-\mathbf{y}) + K_D\mathbf{\Omega}_D(\mathbf{r}-\mathbf{y}) \tag{3.42}$$

or substituting $\mathbf{\Omega u}$ for $\mathbf{y}$

$$\mathbf{u} = K_P(\mathbf{r}-\mathbf{\Omega u}) + K_I\mathbf{\Omega}_I(\mathbf{r}-\mathbf{\Omega u}) + K_D\mathbf{\Omega}_D(\mathbf{r}-\mathbf{\Omega u}) \tag{3.43}$$

for the control input vector $\mathbf{u}$.

To this point, the formulation of the control problem related to the optimization of PID controllers is presented. It consists of solving the LMI problem (3.27) subject to the constraint given by Equation (3.43). It means that the input vector $\mathbf{u}$ is constrained to be the output of the discrete-time PID controller. It is nonlinear since the parameters to be optimized, K_P, K_I and K_D are multiplied by the input vector $\mathbf{u}$, which is also an optimization parameter. Due to this fact, this problem is very hard to solve. In the next section, a method is proposed to overcome this difficulty.

3.4 Optimization of PID Controllers

In this section, the optimization of discrete-time PID controllers is presented. A method is given to approximately formulate the nonlinear linear constraint (3.43) in terms of linear matrix inequalities. The resulting optimizing problem is expressed as a set of LMIs in the controller parameters K_P, K_I and K_D. The obtained controller can be then implemented to generate the optimal input sequence $u_{opt}(k)$. A simulation example is used to show the effectiveness of this approach. Moreover, the performance of the optimized controller is compared to the performance of a robust PID controller with Nonsmooth method (Apkarian and Noll 2006) implemented in the Matlab function *Hinfstruct*. The pros and cons of our method are discussed at the end of this section.

Before presenting the main result of this section, the overall optimization problem is recalled. It consists of the following set of LMIs

Problem 3.4 :

$$\begin{aligned} \underset{\mathbf{u}}{\text{minimize}} \quad & \gamma \\ \text{subject to} \quad & \begin{bmatrix} -\gamma & (\mathbf{y}_d - \boldsymbol{\Omega}\mathbf{u})^T \\ (\mathbf{y}_d - \boldsymbol{\Omega}\mathbf{u}) & -\mathbf{I}_N \end{bmatrix} < 0, \\ & \mathbf{E}_u\mathbf{u} \leq \mathbf{f}_u, \;\; \mathbf{E}_y(\boldsymbol{\Omega}\mathbf{u}) \leq \mathbf{f}_y. \end{aligned} \tag{3.44}$$

together with the equality constraint

$$\mathbf{u} = K_P\left(\mathbf{r} - \boldsymbol{\Omega}\mathbf{u}\right) + K_I\boldsymbol{\Omega}_I\left(\mathbf{r} - \boldsymbol{\Omega}\mathbf{u}\right) + K_D\boldsymbol{\Omega}_D\left(\mathbf{r} - \boldsymbol{\Omega}\mathbf{u}\right). \tag{3.45}$$

This equality is a hard nonlinear constraint. It requires that the plant input $\mathbf{u}$, left hand side of the equality (3.45), is equal to the controller output defined by the right hand side of this equality. The idea of this work is to replace the equality (3.45) with an inequality constraint in the form

$$\|\mathbf{u} - K_P(\mathbf{r} - \boldsymbol{\Omega}\mathbf{u}) + K_I\boldsymbol{\Omega}_I(\mathbf{r} - \boldsymbol{\Omega}\mathbf{u}) + K_D\boldsymbol{\Omega}_D(\mathbf{r} - \boldsymbol{\Omega}\mathbf{u})\|_2^2 < \sigma. \tag{3.46}$$

This is a soft version of the equality constraint. Minimizing σ will result in a controller that approximately generates the input signal instead of exactly. Nevertheless, inequality (3.46) is still not an LMI. The goal is to get rid of the multiplication between the parameters K_P, K_I and K_D and the input vector $\mathbf{u}$. To obtain the required LMI form, assume that the optimization problem (3.44) is feasible. Moreover, assume that the value of γ is nearly zero or at least very small. In this case, the optimal input vector $\mathbf{u}_{opt}$ satisfies $\boldsymbol{\Omega}\mathbf{u}_{opt} = \mathbf{y} \approx \mathbf{y}_d$. It means that the plant output $\mathbf{y}$ approximately fits the desired response $\mathbf{y}_d$. With the help of this assumption, inequality (3.46) can be formulated as follows

$$\|\mathbf{u} - K_P(\mathbf{r} - \mathbf{y}_d) + K_I\boldsymbol{\Omega}_I(\mathbf{r} - \mathbf{y}_d) + K_D\boldsymbol{\Omega}_D(\mathbf{r} - \mathbf{y}_d)\|_2^2 < \sigma. \tag{3.47}$$

The term $\boldsymbol{\Omega}\mathbf{u}$ is now replaced with the desired response $\mathbf{y}_d$, which is a fixed known vector. This gets rid of the multiplication between the input vector $\mathbf{u}$ and the controller parameters in Equation (3.46). It can be used to replace the nonlinear constraint (3.46). To this point, the main approach is now presented. For this reason, consider inequality (3.47). This can be formulated as

$$(\mathbf{u} - \tilde{\mathbf{u}})^T(\mathbf{u} - \tilde{\mathbf{u}}) < \sigma \tag{3.48}$$

with

$$\tilde{\mathbf{u}} = K_P(\mathbf{r} - \mathbf{y}_d) + K_I\boldsymbol{\Omega}_I(\mathbf{r} - \mathbf{y}_d) + K_D\boldsymbol{\Omega}_D(\mathbf{r} - \mathbf{y}_d). \tag{3.49}$$

Applying the Schur complement on (3.48) leads to

$$\begin{bmatrix} -\sigma & (\mathbf{u}-\tilde{\mathbf{u}})^T \\ (\mathbf{u}-\tilde{\mathbf{u}}) & -I_N \end{bmatrix} < 0, \tag{3.50}$$

which is a LMI in the input vector $\mathbf{u}$ and in the controller parameters K_P, K_I and K_D. Putting all together, the overall optimization problem is defined by the following formulation

Problem 3.5 :

$$\begin{aligned} \underset{\mathbf{u},\, K_P, K_I, K_D}{\text{minimize}} \quad & \gamma + \sigma \\ \text{subject to} \quad & \begin{bmatrix} -\gamma & (\mathbf{y}_d - \mathbf{\Omega u})^T \\ (\mathbf{y}_d - \mathbf{\Omega u}) & -\mathbf{I}_N \end{bmatrix} < 0, \\ & \begin{bmatrix} -\sigma & (\mathbf{u}-\tilde{\mathbf{u}})^T \\ (\mathbf{u}-\tilde{\mathbf{u}}) & -\mathbf{I}_N \end{bmatrix} < 0, \\ & \mathbf{E}_u\mathbf{u} \leq \mathbf{f}_u, \;\; \mathbf{E}_y(\mathbf{\Omega u}) \leq \mathbf{f}_y. \end{aligned} \tag{3.51}$$

It represents an approach to approximate MPC for LTI systems and at the same time a method to optimize the parameters of PID controllers satisfying input and output constraints.

Remark. *The existence of a PID controller using the proposed method is dependent on the feasibility of Problem 3.5. As the resulting optimal input is generated through the discrete-time PID controller, classical linear stability theorems can be used to check the stability, see for example (Unbehauen 2008)[p. 140], of the resulting closed-loop system. In the scope of this work, SISO- as well MIMO-LTI stable systems have been considered.*

To show the effectiveness of the LMI formulation (3.51), we consider the optimization of a robust controller for the electronic throttle based on the discrete-time model (3.12). The parameter V ranges in the interval [0.5 1.5].

The dynamic of the electronic throttle contains an integrator. For this reason, the following PD_f controller given by

$$K(z) = K_P + K_D F_d(z) \tag{3.52}$$

is considered. $F_d(z)$ consists the discrete-time filtered derivative. The goal it to optimize the parameters K_P and K_D such that the robustness requirements defined in Section 3.2 are satisfied. The goal is to bring the output of the throttle from 0° up to 10° without

any overshoot for all variations of the parameter V. These are given in terms of the desired bandwidth $\omega_c^d = 40\ rad/s$. The desired output response $\mathbf{y}_d$ is given through the step response of the reference discrete-time function $T_d(z)$ discussed in Section 3.2 based on ω_c^d. Additionally to the robust performance given by the desired trajectory $\mathbf{y}_d$, the plant input is constrained by a maximal value $U_{max} = 5$. The constraint for the input in problem (3.51) is $\mathbf{E}_u\mathbf{u} \leq \mathbf{f}_u(U_{max} = 5)$. The output constraint given by $\mathbf{E}_y(\mathbf{\Omega u}) \leq \mathbf{f}_y$ is not used due to the fact that the plant output $\mathbf{y}$ has to track the desired trajectory $\mathbf{y}_d$.

Now the optimization problem can be formulated. For this purpose, substitute the desired response $\mathbf{y}_d$ in Problem 3.5. Moreover, due to the structure of the PD_f controller the control input (3.49) is given by

$$\tilde{\mathbf{u}} = K_P(\mathbf{r} - \mathbf{y}_d) + K_D\mathbf{\Omega}_D(\mathbf{r} - \mathbf{y}_d)_f. \tag{3.53}$$

Thereby, $\mathbf{r}$ consists of a step command from 0° to 10° in vector notation. The operator $(\mathbf{r} - \mathbf{y}_d)_f$ denotes the filtered error in vector notation. After defining the optimization problem, the SeDuMi solver in combination with the modeling language YALMIP is used to compute the optimal controller parameters K_P and K_D.

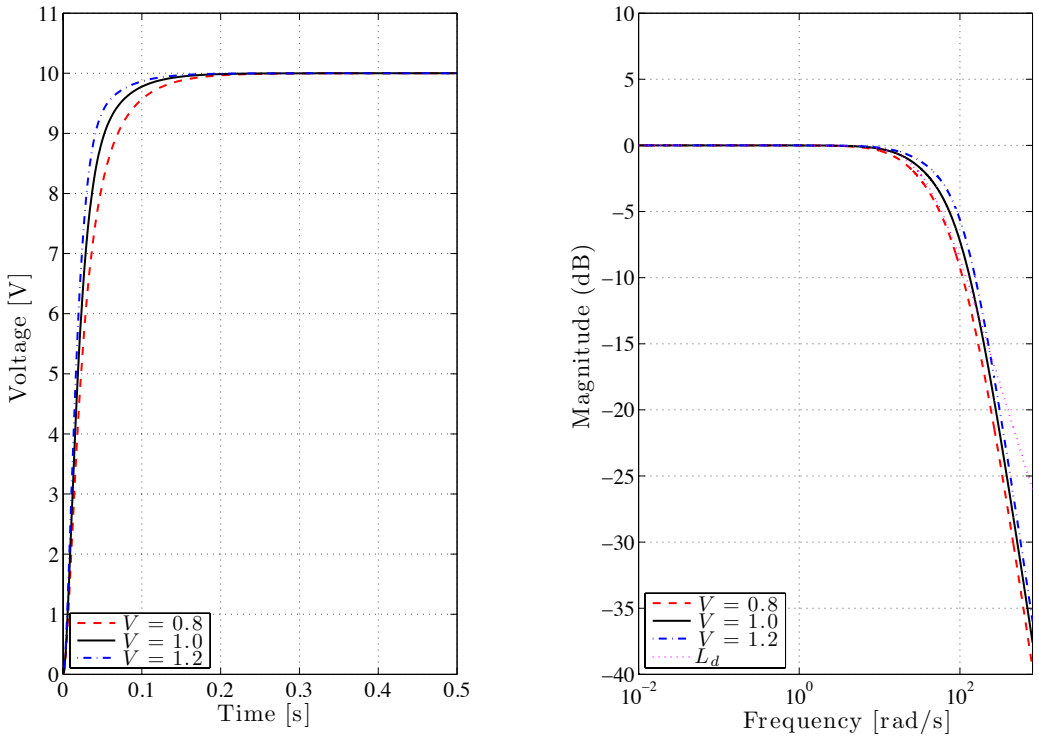

Figure 3.8: Closed-loop step response (left) and bode magnitude of the desired (yellow dashed) and achieved closed-loop response for $V = 0.8$ (dashed red), $V = 1.0$ (black) and $V = 1.2$ (dashdot blue)

Figure 3.8 shows the step and the frequency response of the closed-loop system with the obtained discrete-time controller

$$K(z) = \frac{0.5127z - 0.4955}{z - 0.8187} \tag{3.54}$$

with the sampling time 1 ms. For all values of the uncertain parameter V, the step response does not show any overshoot. This is due the fact that a parameter variation causes a shift of the closed-loop bandwidth. On the other hand, the amplitude at low frequencies is for all values of V one. The robustness requirements are in this case achieved. The controller PD_f provides the required performance in the presence of parameter variations.

Additionaly to the robust performance, the controller output should fullfil the constraint given by the maximal input value $U_{max} = 5$. Figure 3.9 shows the MPC open-loop optimized input trajectory and the input signal generated by the PD controller. It is clear that both trajectory satisfy the input constraint (smaller than 5). Moreover, the generated inputs for $V = 0.8$, $V = 1.0$ and $V = 1.2$ are also within the specified range.

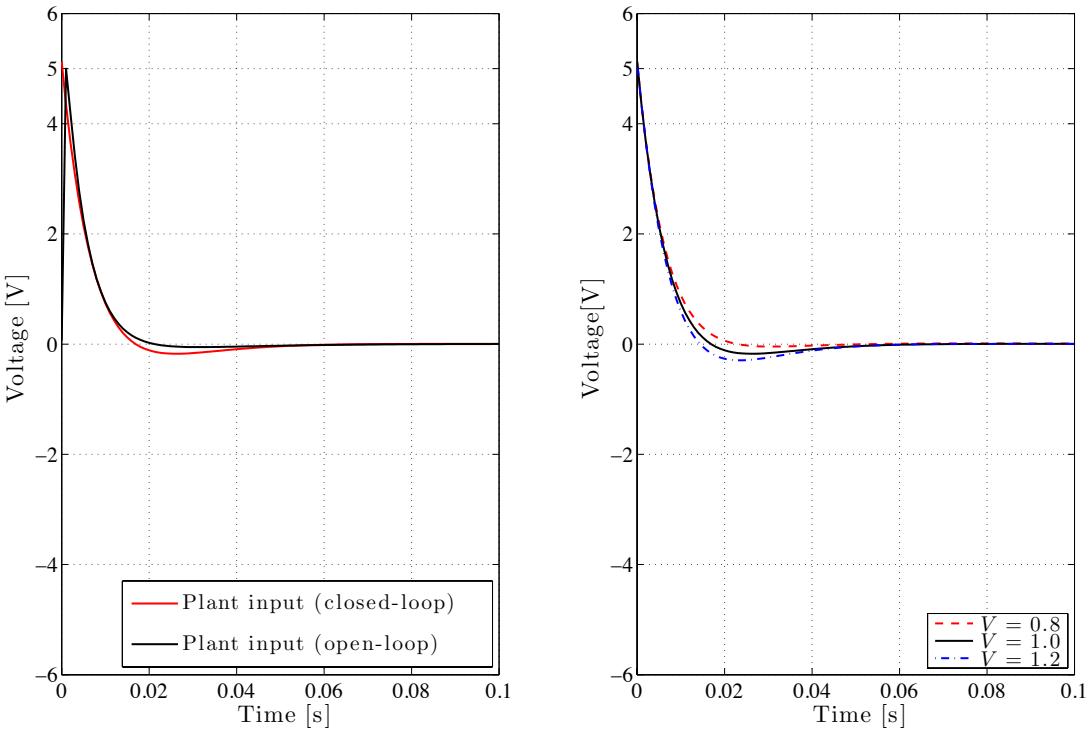

Figure 3.9: Controller output and optimized MPC open-loop plant input (left), Controller output for $V = 0.8$ (dashed red), $V = 1.0$ (black) and $V = 1.2$ (dashdot blue)

The above example shows that the PD controller which is optimized using our method

fullfil the specified robust performance in case of static gain variation. Moreover, the input constraint imposed on the controller output is also satisfied. To further investigate the application of this controller, a comparison to existing method to compute robust PD controllers is given. A well known appraoch that can be used is the H_∞ based PD controller, see (Gahinet and Apkarian 2011). One has to mention here that this method does not consider any type of hard constraints on the plant input.

For the same plant model and using the same robust requirement given in terms of the desired bandwidth $\omega_c^d = 40\ rad/s$, an H_∞-PD controller is computed using the Matlab function *Hinfstruct*.

The step response with the obtained controller

$$K(z) = \frac{0.638z - 0.617}{z - 0.8187} \tag{3.55}$$

are shown in Figure 3.10. At first sight, one can say that the required robust performance is approximately achieved, except a small deterioration in the step response for $V = 1.2$. Considering the bode magnitude response for the nominal case, the achieved nominal bandwidth is $\omega_c^0 = 66\,rad/s$ which is higher than the achieved bandwidth with PD controller using our method $50\,rad/s$.

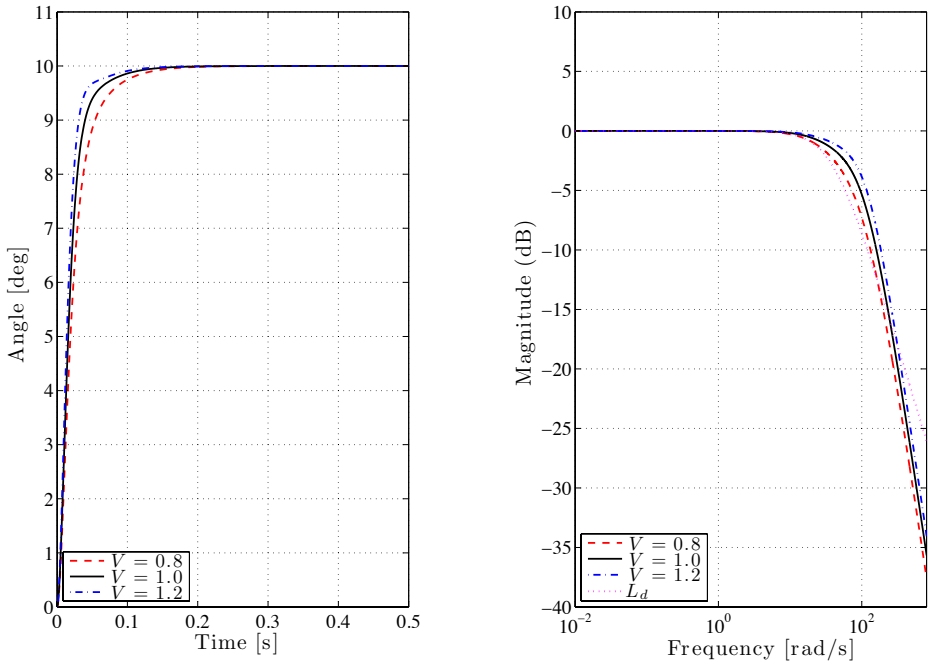

Figure 3.10: Closed-loop step response with H_∞ controller (left) and bode magnitude of the desired (yellow dashed) and achieved closed-loop response for $V = 0.5$ (dashed red), $V = 1.0$ (black) and $V = 1.5$ (dashdot blue)

Additionally to the required robust performance, the H_∞ controller has to generate an input signal within the maximal value given by $U_{max} = 5$. This is shown in Figure 3.11. The controller output is out of this range. This is due to the fact that the method in (Apkarian and Noll 2006) does not take any type of hard constraints.

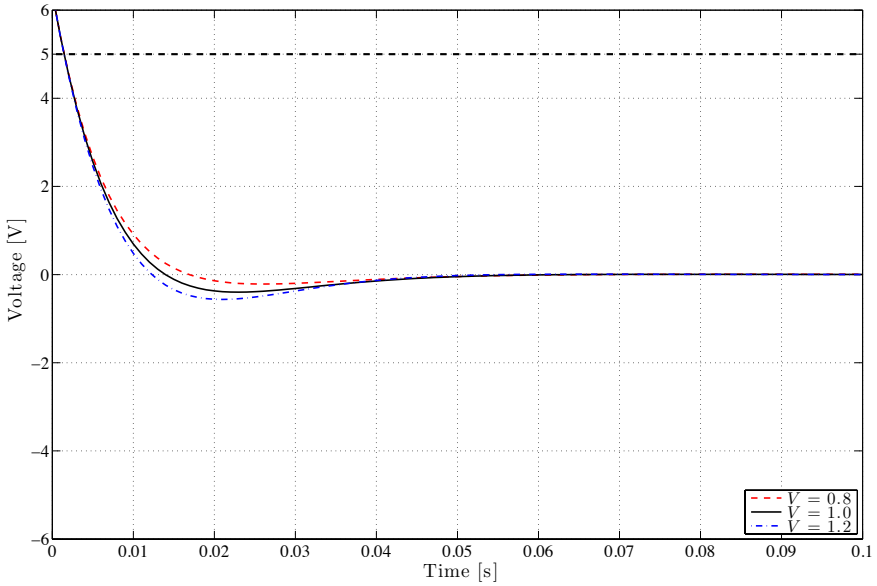

Figure 3.11: Plant input for $V = 0.8$ (dashed red), $V = 1.0$ (black) and $V = 1.2$ (dashdot blue)

Table 3.1: Overshoots in (Apkarian and Noll 2006)

Criteria	Complexity	Constraint (Input/Output)	Controller	Optimization
Method (3.51)	Mittel	Possible	PID	Offline
(Apkarian and Noll 2006)	High	No	PID	Offline
(Kothare et al. 1996)	High	Possible	State feedback (adaptive)	Online

By complexity, we mean the effort required to implement the related method. The method in (Apkarian and Noll 2006) is based on Nonsmooth optimization techniques, which requires the computation of the H_∞ subgradient, see (Clarke 1990). The method in (Kothare et al. 1996) is based on the LMI formulation of the related MPC problem. Adaptive state feedback controller is used to approximate the MPC optimal input trajectory. The controller computation is performed online by solving an LMI problem at each sampling time. It constrains the controller application to plants with slow sampling time. Contrary to

(Kothare et al. 1996), the optimization problem (3.5) is solved in the controller parameters offline. The optimal input trajectory is approximated using a PID controller. Robustness is taking into consideration through the specification of a desired output trajectory.

Our Method and the approach in (Apkarian and Noll 2006) consider only linear system without time delay. This can be overcome by approximating the delay part using a linear transfer function. The appraoch in (Kothare et al. 1996) deals directly with time delay.

Our appraoch is not part of the Matlab Toolbox (Robust Control). It is based on the free LMI solver (Labit et al. 2002) and (Lofberg 2004), which makes it free accessible.

3.5 Summary

In this chapter, a novel method was presented to optimize the parameters of PID controllers, which provides an alternative method to be used instead of the Matlab function *Hinfstruct* for the optimization of fractional controllers in the previous section. Moreover, it consists a modern appraoch to optimize the parameters of PID controllers. Our proposed technique in this section is based on MPC approach to formulate the control problem. First of all, Section 3.2 deals with the considered performance specifications. The feedforward as well as the feedback or closed-loop MPC problem has been discussed in Section 3.3. Moreover, a method is presented to transform the resulting control problem into a LMI. This was successfully applied to compute the optimal trajectory for a model of the electronic throttle. The design and optimization of a robust PID controller is also treated. The resulting optimization problem turns out to be a BMI. It can be not solved using existing LMI solvers. In Section 3.4, a method has been presented to transform this problem into a set of LMIs. To test the effectiveness of the whole approach, a robust PD_f is design for the electronic throttle. Moreover, simulation results have shown that the controller provides the desired performance for all variations of the static gain of a given plant.

4 Artificial Neural Network Controllers

4.1 State of the Art (ANN)

The permanent development of nonlinear black-box tools and methodologies is aimed to identify high nonlinear systems. Among modern nonlinear black-box model structures (see e.g. (Sjöberg et al. 1995) and references therein) ANNs have been extensively suggested and used. Especially, this is due to their inherent approximation capabilities. A three layered feedforward neural network containing a sufficient number of neurons with Sigmoidal activation functions can (at least theoretically) approximate, to any level of accuracy, a given continuous nonlinear function (Sontag 1997). Neural networks constitute, beside a tool for modeling purpose, an alternative method to design controllers for linear and nonlinear systems. Therefore, as a further technique to be implemented later in a real system neural ANNS are explored. Contrary to the method proposed in Section 3, we are considering the optimization of general linear state-space controllers instead of PID controllers.

Feedforward neural networks are static network structures. They are used in nonlinear system identification, see (Nørgård et al. 1996) and (Nørgård 2000). The way these structures approximate the dynamic of the system to be identified is carried out by providing a sequence of past inputs and past outputs to the neural network. This type of networks is more suitable to approximate and identify static functions.

A different way to represent dynamic systems is to incorporate feedback connections within the hidden layers. The resulting neural network is known as Recurrent Neural Network (RNN). These structures represent a broader class of nonlinear dynamic systems. They can be used for grey-box as well as for black-box identification. The first one is achieved by building a RNN with the same structure as the plant to be identified. Furthermore, RNNs can be used to identify open-loop as well as closed-loop systems. A RNN structure has been successfully used to identify a nonlinear system operating in closed-loop, see (Lachhab et al. 2008). The idea is based on representing the closed-loop system as a network consisting of two RNNs. One RNN represents the implemented controller. Its weights are fixed to the controller parameters. Another recurrent network represents the plant to be identified. The weights of this network are adapted during the training or the optimization stage.

In (Widrow and Bilello 1993), the identification of a nonlinear system is shown. Moreover, the authors used a neural network to approximate the inverse of the nonlinear model. The validation of this method is given only by means of simulations. In (Mohagheghi et al. 2005), a dynamic neural network structure is used for nonlinear identification in power system. The training of this network shows that the system dynamic is well identified thanks to the use of a nonlinear state-space representation. In (Arsie et al. 2008), the application of RNNs to the Air Fuel Ratio (AFR) estimation is proposed. Based on measured signals, a RNN is trained to approximate the AFR. Validation of the RNN model

using measured data is given. Moreover, the authors identify an inverse model of the plant and apply the related controller to the real plant. In (Hunt and Sbarbaro 1991), Internal Model Control (IMC) principle is applied to a nonlinear system. First, a nonlinear model of the plant is obtained using a neural network. Based on this network and an inverse model identification, IMC is applied to the nonlinear plant. Application of neural network compensation in combination with sliding mode control and feedback linearization is presented in (Xu et al. 1991). In (Huang et al. 1999), a neural network is used to estimate the load torque of the induction motor and to identify the related model. In the field of Linear Parameter Varying (LPV) system identification, linear recurrent neural networks are used in (Abbas et al. 2010) for identification purpose in open- as well as in closed-loop. Other applications of neural networks can be found in (Isermann and Müller 2001) and (Dias and Mota 2001).

Actually, the use of neural networks can be classify as follows

- **System Identification**. Neural networks are used to identify nonlinear dynamics. The resulting structure constitutes the counterpart to the well known linear identification methods. These are summarized here (Nørgård 2000, p. 38 and p.122):
  - FIR/NNFIR: Neural Network Finite Impulse Response. Only past inputs are used for identification.
  - ARX/NNARX: Neural Network Auto Regressive External. Past inputs and observed outputs are used for identification.
  - OE/NNOE: Neural Network Output Error. Identification is based on past inputs and output predictions.
  - ARMAX/NNARMAX: Neural Network Auto Regressive Moving Average External. Identification is based on past inputs, output predictions and residuals.
  - SSIF/NNSSIF: Neural Network State-Space Innovation Form. Identification is based on past inputs, states estimation and residuals.
- **Control Design**. Neural networks are used for control purposes. This approach can be classified as follows:
  - Direct design: This approach means that the controller itself is a neural network. In this case, a model is not necessary required to design the controller. The neural network is trained online to achieve a specified performance. Such a design includes, among others, direct inverse control, internal model control, feedback linearization, feedforward strategy and optimal control.
  - Indirect design: This approach means that neural network strutctures are trained to approximate the dynamic of a given nonlinear plant. Once a neural network is obtained, the controller can be optimized based on this model. A well-known method is given by model predictive control.

4.2 Neural Network Topology

In this section, ANN are briefly introduced. Two structures are considered, namely feedforward and feedback. Moreover, a numerical example given by a nonlinear static function is considered to demonstrate the approximation capability of neural networks. Afterword, the main contribution in the field of neural networks, which is based on the work (Lachhab et al. 2008) is discussed. The main difference to the work (Lachhab et al. 2008) is that the plant is already identified, but the controller is not known. Instead of using RNN for identification purposes, it is used to optimize the controller parameters in closed-loop. The resulting RNN controller is linear and time invariant.

4.2.1 Feedforward and Recurrent Neural Networks

In the last years, ANNs have become a wide attention in the control community. Thanks to its learning capability, ANNs can be used to solve linear and nonlinear identification problems as well as control problems. The efforts by many researchers to understand the humain brain (nervous system) and its elementary units gave birth to ANNs. The humain brain consists of a huge number of neurons which interact with each other, see (Sarangapani 2006). Neurons constitute the basic unit of the nervous system.

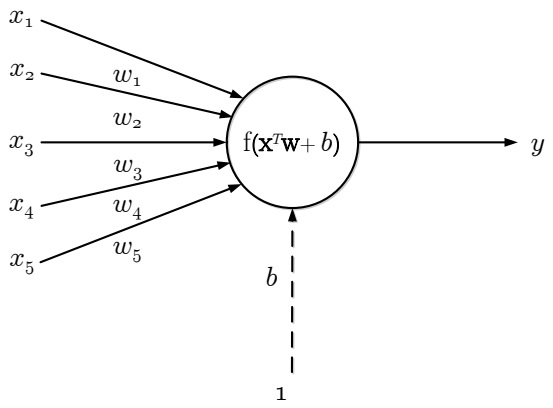

Figure 4.1: Artificial neuron (Nørgård 2000, p. 6)

The structure of a simplified neuron is shown in Figure 4.1. The inputs to the neuron are x_1 to x_5. The coefficients w_1 to w_5 are weights of the related inputs. Additionally, the neuron has a constant input, which is set to one. This provides a bias or an offset b. The function $f(\cdot)$ is called the network activation function. It maps the inputs of the neuron

to the output y, which is given by

$$y = f\left(\sum_{i=1}^{5} w_i x_i + b\right) \tag{4.1}$$

or in vector form

$$y = f(\mathbf{x}^T \mathbf{w} + b) \tag{4.2}$$

with

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_5 \end{bmatrix} \quad \text{and} \quad \mathbf{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_5 \end{bmatrix}. \tag{4.3}$$

The scalar activation function $f(\cdot)$ plays an important role in the field of neural networks. It exists a high number of differently defined activation functions. Generally, these functions are classified into linear and nonlinear activation functions. A detailed description of the commonly used functions is given in (Sarangapani 2006).

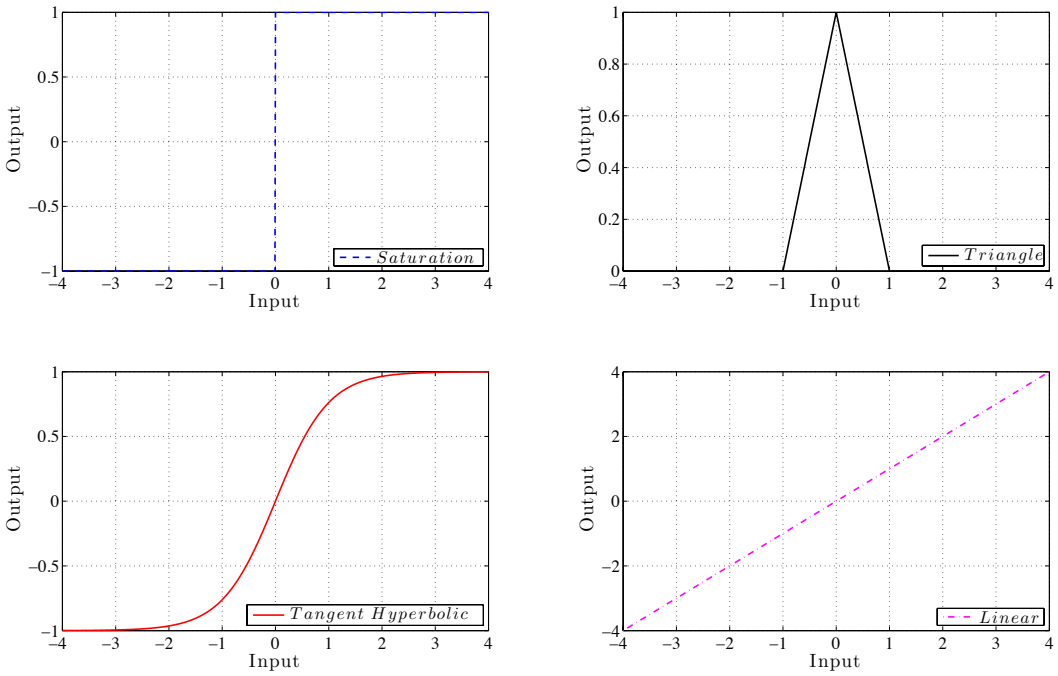

Figure 4.2: Linear and nonlinear activation functions

Figure 4.2 shows four well known functions. Depending on the input-output behavior to

be approximated, the type of the neuron activation function is defined. In case that the dynamic to be approximate is nonlinear, the *Hyberbolic Tangent* function is frequently used. In (Sontag 1997), the identification based on this nonlinear function is discussed. The author proves that a neural network with a sufficiently high number of *Hyberbolic Tangent* neurons can approximate any arbitrary given nonlinear function, to any desired accuracy.

Figure 4.3 shows a neuron, which represents the *Hyberbolic Tangent* function with a bias b. The output of this neuron is given by

$$y = \tanh\left(\mathbf{x}^T\mathbf{w} + b\right) = \frac{e^{\left(\mathbf{x}^T\mathbf{w}+b\right)} - e^{\left(-\mathbf{x}^T\mathbf{w}-b\right)}}{e^{(\mathbf{x}^T\mathbf{w}+b)} + e^{(-\mathbf{x}^T\mathbf{w}-b)}}. \tag{4.4}$$

This activation function maps values x between $-\infty$ and $+\infty$ to the bounded output y between -1 and $+1$. A neuron constitutes the basic unit of neural networks. Combining several neurons to a global network leads to the so-called layers. A neural network with more than one layer is called Multilayer Perceptron network (MLP). It represents the standard and general neural network structure.

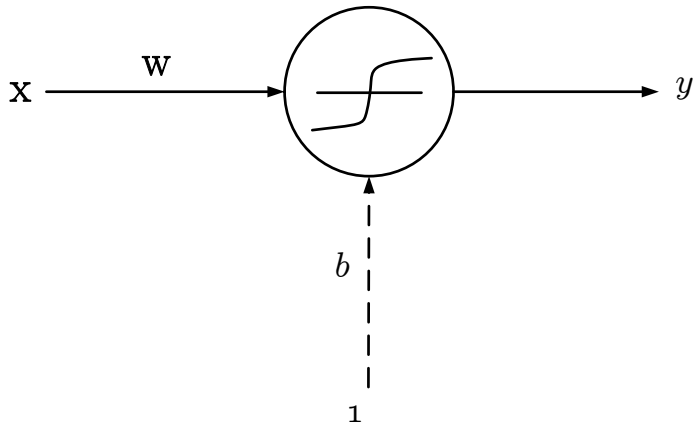

Figure 4.3: Artificial neuron with nonlinear activation function

Figure 4.4 shows a MLP network with r inputs and q outputs. $f_1(\cdot)$ to $f_n(\cdot)$ and $h_1(\cdot)$ to $h_m(\cdot)$ are the activation functions. The MLP network consists of an input layer with inputs φ_1 to φ_r, a hidden layer and an output layer that generates the outputs z_1 to z_q. Mostly, the activation functions of the input layer are nonlinear. On the other hand, the output layer is represented with linear activation functions.

The outputs of the MLP in Figure 4.4 are computed as follows (Nørgård 2000, p. 8)

$$z_i = h_i\left(\sum_{j=1}^{n} v_{ji} f_j\left(\sum_{l=1}^{m} \varphi_l w_{lj} + b_{Il}\right) + b_{Hi}\right). \tag{4.5}$$

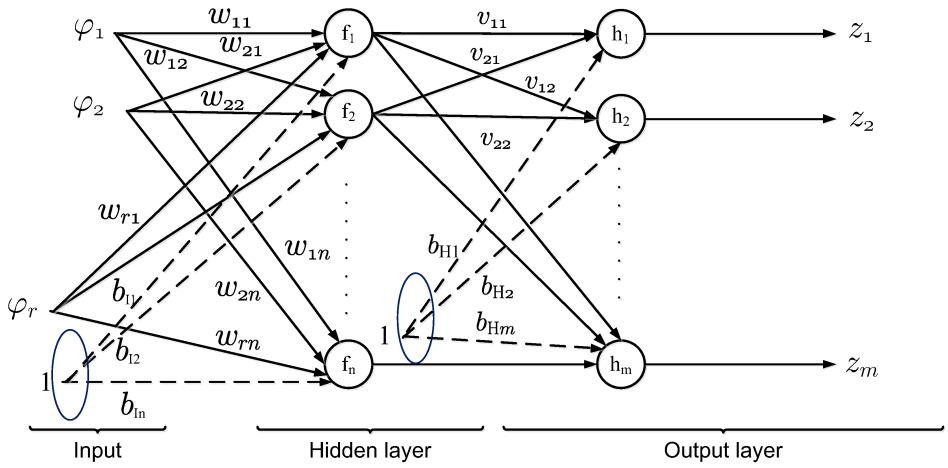

Figure 4.4: Multilayer Perceptron Neural Network (two layers: hidden and output), (Nørgård 2000, p. 8)

It is common that the activation functions of one layer are equal. It means that $f_1(\cdot) = f_2(\cdot) = \cdots = f_n(\cdot) = f(\cdot)$ and $h_1(\cdot) = h_2(\cdot) = \cdots = h_m(\cdot) = h(\cdot)$. Now define the vector field activation functions $\mathbf{f}(\cdot)$ and $\mathbf{h}(\cdot)$ as follows

$$\mathbf{f}(\cdot) = \begin{bmatrix} f(\cdot) \\ f(\cdot) \\ \vdots \\ f(\cdot) \end{bmatrix} \quad \text{and} \quad \mathbf{h}(\cdot) = \begin{bmatrix} h(\cdot) \\ h(\cdot) \\ \vdots \\ h(\cdot) \end{bmatrix}. \tag{4.6}$$

The input and output weights can be regrouped as follows

$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1r} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ w_{n1} & w_{r2} & \cdots & w_{nr} \end{bmatrix} \quad \text{and} \quad \mathbf{V} = \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1n} \\ v_{21} & v_{22} & \cdots & v_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ v_{m1} & v_{r2} & \cdots & v_{mn} \end{bmatrix}. \tag{4.7}$$

Now based on the input vector $\varphi = \begin{bmatrix} \varphi_1 & \varphi_2 & \cdots & \varphi_r \end{bmatrix}^T$, the corresponding output vector $\mathbf{z} = \begin{bmatrix} z_1 & z_2 & \cdots & z_m \end{bmatrix}^T$ can be written in the following compact form

$$\mathbf{z} = \mathbf{h}(\mathbf{V} \cdot \mathbf{f}(\mathbf{W} \cdot \varphi + \mathbf{b}_I) + \mathbf{b}_H). \tag{4.8}$$

Thereby, $\mathbf{b}_I$ and $\mathbf{b}_H$ consist the input and output bias vectors, respectively. These are defined as follows

$$\mathbf{b}_I = \begin{bmatrix} b_{I1} \\ b_{I2} \\ \vdots \\ b_{In} \end{bmatrix} \quad \text{and} \quad \mathbf{b}_H = \begin{bmatrix} b_{H1} \\ b_{H2} \\ \vdots \\ b_{Hm} \end{bmatrix}. \tag{4.9}$$

The expression (4.8) is very useful to represent state-space models as will be shown later. The main characteristic of the MLP network shown in Figure 4.4 is that it has a feedforward structure. It means that the neural network does not contain any delay or feedback component. It is a static network. One of the main application fields of feedforward neural networks is the identification of nonlinear systems. Linear and nonlinear system identification based on this type of networks are provided by the Matlab Toolbox *NNSYSID*, which is developed by (Nørgård et al. 1996). After providing the measured input and output signals, the user of the *NNSYSID* Toolbox can choose the number of layers, the number of inputs and outputs and the type of activation functions. Internally, an optimization problem is set and solved iteratively. The goal is to fit the measured output to the output of the built feedforward neural network.

The main idea is summarized here. Consider for this purpose a given SISO-nonlinear discrete-time system as follows

$$y_k = g(y_{k-1}, y_{k-2}, \cdots, y_{k-n}, u_{k-1}, u_{k-2}, \cdots, u_{k-m}) \tag{4.10}$$

with u_k and y_k denote the input and output at sampling time k with the related orders n and m, respectively. The function $g(\cdot)$ is nonlinear and maps past inputs and outputs onto the actual output y_k. The objective is to approximate the nonlinear function $g(\cdot)$ using the structure (4.4). For this purpose, set now the input and output vectors in this structure as

$$\varphi = \begin{bmatrix} y_{k-1} \\ \vdots \\ y_{k-n} \\ u_{k-1} \\ \vdots \\ u_{k-m} \end{bmatrix} \qquad \text{and} \qquad z_{1k} = h(\mathbf{v}^T \cdot \mathbf{f}(\mathbf{W} \cdot \varphi + \mathbf{b}_I) + b_{H1}) \tag{4.11}$$

with $\mathbf{v} = [v_{11}, v_{21}, \cdots, v_{m1}]^T$. Thereby, z_{1k} consists the output of the MLP network at sampling time k. It represents also the predicted output $\hat{y}_k$ based on the past input and output values, y_{k-1}, y_{k-2}, $\cdots$, y_{k-n} and u_{k-1}, u_{k-2}, $\cdots$, u_{k-m}, respectively. In this case, φ is called the regressor vector. The problem under consideration consists of approximating the output of the real system y_k defined by the unknown function $g(\cdot)$

using the past data. For this purpose, a cost function has to be defined. In this context, the mean square error defined as follows

$$J(e) = \frac{1}{N}\sum_{k=0}^{N} e(k)^2 \qquad \text{with} \qquad e(k) = y(k) - \hat{y}(k) \tag{4.12}$$

is mostly used. N denotes the length of the error signal. The goal is to minimize the cost function $J(e)$ in the neural network parameters $\mathbf{v}$, $\mathbf{W}$, $\mathbf{b}_I$ and b_{H1}. The optimization problem can be formulated as follows

$$\underset{\mathbf{v},\mathbf{W},\mathbf{b}_I,b_{H1}}{\text{minimize}} \quad \frac{1}{N}\sum_{k=0}^{N} e(k)^2. \tag{4.13}$$

Now substituting for the predicted output $\hat{y}_k$ the network output given by the expression (4.11) results in

Problem 4.1 :

$$\underset{\mathbf{v},\mathbf{W},\mathbf{b}_I,b_{H1}}{\text{minimize}} \quad \frac{1}{N}\sum_{k=0}^{N}(y(k) - h(\mathbf{v}^T \cdot \mathbf{f}(\mathbf{W}\cdot\varphi(k) + \mathbf{b}_I) + b_{H1}))^2. \tag{4.14}$$

This problem is quadratic in the error $e(k)$, but it is nonlinear in the network parameters. This is due to the nonlinear activation functions $\mathbf{f}(\cdot)$ and $h(\cdot)$ and the multiplication between the network weights. Due to this fact, the minimizing of this nonlinear problem requires special algorithms. The minimizing of this problem as well as a discussion of the different existing algorithms that can be used is given in Appendix D.

Example 4.1

For a better understanding of MLP feedforward neural networks, the approximation of a given static nonlinear function is presented. For this reason, consider the following given function

$$y(k) = \underbrace{\cos(\pi x(k))\sin(\pi x(k))\exp(-|x(k)|)}_{g(x(k))}. \tag{4.15}$$

The static function $g(x(k))$ maps the sequence $x(k) \in [-1\ 1]$ into the output $y(k)$. Suppose that the nonlinear function $g(\cdot)$ is not known and has to be approximated. Based on the input and output vectors x and y, the goal is to approximate this function using the feedforward structure shown in Figure 4.4. The MLP network has one input, which is $\varphi_1(k) = x(k)$, and one output which is the approximated output $\hat{y}(k) = z_1(k)$. The output function $h(\cdot)$ is chosen to be linear. On the other hand, the hidden layer function

$f(\cdot)$ is chosen to be *Hyperbolic Tangent*.

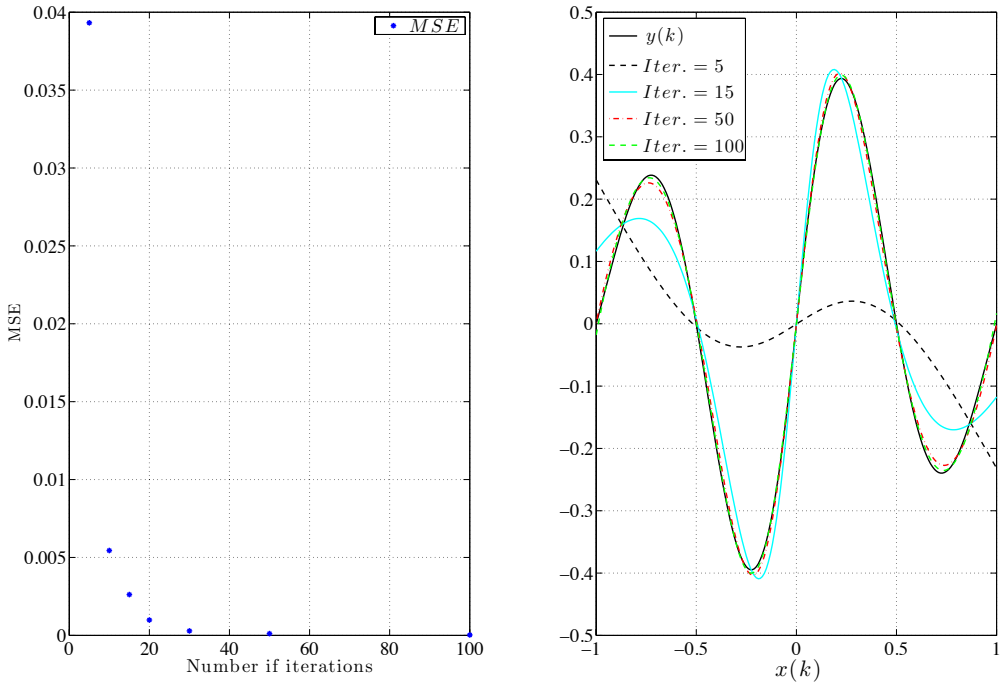

Figure 4.5: Training of the MLP, Mean Square Error (MSE) versus number of iteration (left) and output of the MLP (right)

In what concerns the bias, we set b_{I1} to zero. The output bias b_{h1} is a scalar network weight. Thereby, the hidden and output layer weights are given by

$$\mathbf{w} \in \mathbb{R}^{5\times 1},\ \mathbf{v}^{1\times 5} \in \mathbb{R}. \tag{4.16}$$

The related output of the MLP is described by the following function

$$\hat{y}(k) = \mathbf{v} \cdot \tanh(\mathbf{w} \cdot x(k)) + b_{h1}. \tag{4.17}$$

The goal is to optimize the MLP-parameter given by the vectors $\mathbf{v}$ and $\mathbf{w}$, and the bias b_{h1}. For this purpose, the whole problem is formulated using the Matlab Neural Network Toolbox. Thereby, the cost function is the mean square error given by the expression (4.13). The algorithm used to train the neural network is Levenberg-Marquadt, see Appendix D.

The results with the following obtained network weights

$$\mathbf{w} = \begin{bmatrix} -5.20 \\ -3.35 \\ -3.35 \\ -1.71 \\ 1.71 \end{bmatrix} \qquad \mathbf{v}^T = \begin{bmatrix} -17.89 \\ 17.64 \\ 18.63 \\ -14.96 \\ 8.24 \end{bmatrix} \qquad b_{h1} = -5.82 \tag{4.18}$$

are shown in Figure 4.5. The convergence of the algorithm shows that after only 20 iterations, the output of the MLP approximately fits the real output.

The static problem above is nonlinear, which makes the optimization hard. However, thanks to the algorithm Levenberg-Marquadt a good convergence was achieved. The obtained set of parameters provides a good fit to the function output.

4.2.2 Recurrent Neural Networks

The example considered above shows that feedforward networks are capable to approximate static nonlinear functions. They are suitable in case that the system under consideration does not display any dynamics. However, in the field of control and modeling the systems under consideration are mostly dynamic. For this reason, the MLP network given by the structure shown in Figure 4.4 has to be modified. Another class of MLP networks is needed. Such a class has to contain tapped delay lines. These class are well known as recurrent or dynamic network. Figure 4.6 shows such a dynamic neural net-

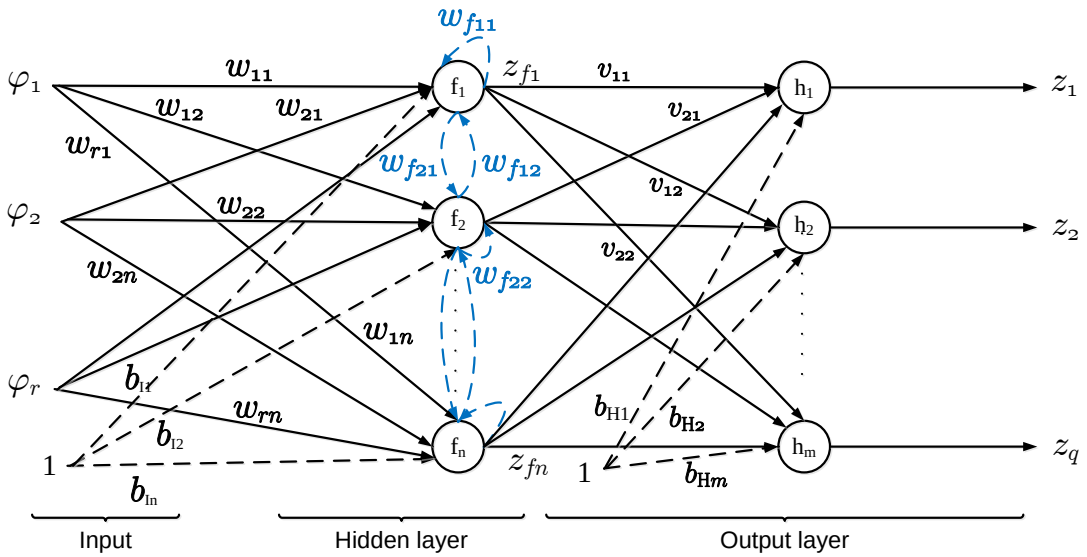

Figure 4.6: Multilayer recurrent neural network (Nørgård 2000, p. 11)

works. The hidden layer contains a delay component. In this case, the output of the

activation functions $f_1(\cdot)$ to $f_n(\cdot)$ depends on the values at time k and on the previous values at time $k-1$. Such neural networks are called recurrent networks. They can be seen as discrete-time nonlinear systems. Using the structure shown in Figure 4.6 with the appropriate choice of the related weights, nonlinear dynamic systems can now be represented as recurrent neural networks. This structure consists a bridge between neural networks and dynamic nonlinear systems.

The output of the hidden layer z_{fi} can be computed as follows

$$z_{fj}(k+1) = f_i\left(\sum_{l=1}^{r} \varphi_l(k) w_{li} + \sum_{j=1}^{n} z_{fi}(k) w_{fij} + b_{Ii}\right). \tag{4.19}$$

Now using the weights definition (4.7), the bias definition (4.9), the input and output vectors φ and $\mathbf{z}$ and the following feedback matrix

$$\mathbf{W}_f = \begin{bmatrix} w_{f11} & w_{f12} & \cdots & w_{f1n} \\ w_{f21} & w_{f22} & \cdots & w_{f2n} \\ \vdots & \vdots & \vdots & \vdots \\ w_{fr1} & w_{fr2} & \cdots & w_{frn} \end{bmatrix}, \tag{4.20}$$

the output vector of the hidden layer can be written as

$$\underbrace{\mathbf{z}_f(k+1)}_{\mathbf{x}(k+1)} = \mathbf{f}(\underbrace{\mathbf{W}\varphi(k)}_{\mathbf{Bu}(k)} + \underbrace{\mathbf{W}_f \mathbf{z}_f(k)}_{\mathbf{Ax}(k)} + \mathbf{b}_I). \tag{4.21}$$

Thereby, the vector $\mathbf{z}_f$ is defined as follows

$$\mathbf{z}_f = \begin{bmatrix} z_{f1} \\ z_{f2} \\ \vdots \\ z_{fn} \end{bmatrix}. \tag{4.22}$$

Based on this vector, the output layer provides the following network output given by

$$\underbrace{\mathbf{z}(k)}_{\mathbf{y}(k)} = \mathbf{h}(\underbrace{\mathbf{V} \cdot \mathbf{z}_f(k)}_{\mathbf{Cx}(k)} + \mathbf{b}_h). \tag{4.23}$$

The representation of the recurrent network outputs (4.21) and (4.23) relates neural networks to nonlinear dynamic systems. Moreover, assume that the activation functions $\mathbf{f}(\cdot)$ and $\mathbf{g}(\cdot)$ are linear and that the bias vectors $\mathbf{b}_h$ and $\mathbf{b}_I$ are set to zero. Then, the recurrent network given by the structure shown in Figure 4.6 and the equations (4.21) and (4.23) defines a nonlinear MIMO discrete-time model. Moreover, this system has q outputs, r inputs and n states.

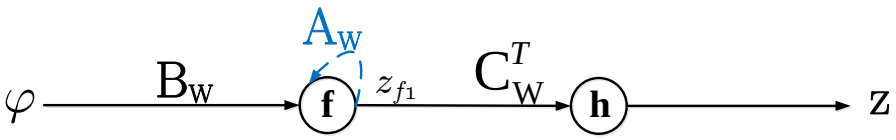

Figure 4.7: State-space RNN

Remark. *To make the discussion later easier, the recurrent network given by Figure 4.6 will be replaced by the simplified representation shown in Figure 4.7. The subscript W in the matrices $\mathbf{A}_W$, $\mathbf{B}_W$ and $\mathbf{C}_W$ means that these matrices are network matrices. It aims to distinguish between general discrete-time state-space models and network models.*

4.2.3 Closed-Loop Recurrent Neural Networks

In the previous sections, feedforward as well as feedback or recurrent neural networks have been presented and discussed. Due to their inherent dynamic, recurrent networks present an interesting network structure. In this case, dynamic linear as well as nonlinear discrete-time systems can be represented as a neural network. This fact opens new possibilities to apply neural networks in the field of identification and control. The successful application of such a dynamic network is presented in (Lachhab et al. 2008), which was used to identify a nonlinear unstable system operating in closed-loop. Moreover, the authors present additionally a novel class of recurrent networks. This is characterized by the fact that it contains two recurrent networks instead of one. The idea is based on representing the plant to be identified as well as the implemented controller, each by a recurrent network. Based on the idea in (Lachhab et al. 2008), the main approach of this work is

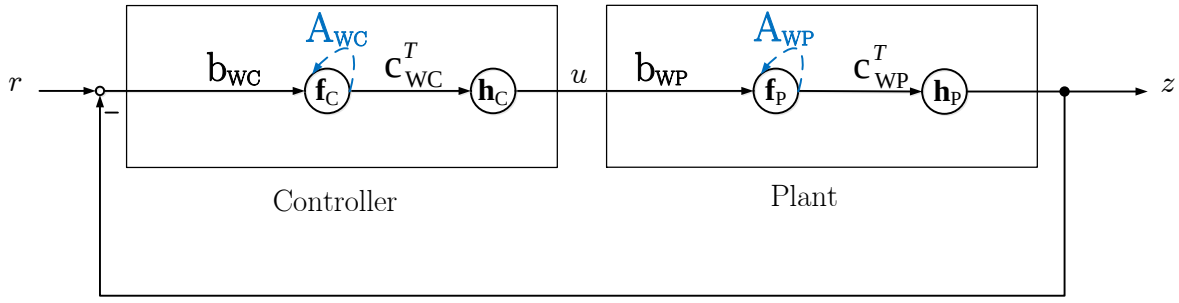

Figure 4.8: Closed-loop recurrent neural network (nonlinear structure)

now presented. For this purpose, the SISO case is considered. Thereby, the output z of the closed-loop structure shown in Figure 4.8 is given by

$$z(k) = h_P(\mathbf{c}_{WP}^T \cdot \mathbf{z}_{fP}(k)), \tag{4.24}$$

with h_P, $\mathbf{c}_{WP}^T$ and $\mathbf{z}_{fP}$ denote the output activation function, the output weight and states of the recurrent neural network, respectively. The update of the state vector $\mathbf{z}_{fP}$ is given by

$$\mathbf{z}_{fP}(k+1) = \mathbf{f}_P(\mathbf{A}_{WP} \cdot \mathbf{z}_{fP}(k) + \mathbf{b}_{WP} \cdot u(k)), \tag{4.25}$$

with $\mathbf{f}_P$, $\mathbf{A}_{WP}$ and $\mathbf{b}_{WP}$ denotes the hidden activation function, the state and input weights of the plant network, respectively. Equation (4.24) and (4.25) describe the dynamic of a given nonlinear system with input u, states $\mathbf{z}_{fP}$, output z and the system matrices $\mathbf{A}_{WP}$, $\mathbf{b}_{WP}$ and $\mathbf{c}_{WP}$. Moreover, assume that a controller is also provided that generates the input signal $u(k)$ given by

$$u(k) = h_C(\mathbf{c}_{WC}^T \cdot \mathbf{z}_{fC}(k)), \tag{4.26}$$

with the controller state vector $\mathbf{z}_{fC}$

$$\mathbf{z}_{fC}(k+1) = \mathbf{f}_C(\mathbf{A}_{WC} \cdot \mathbf{z}_{fC}(k) + \mathbf{b}_{WC} \cdot (r(k) - z(k))). \tag{4.27}$$

Thereby, the controller matrices are denoted by $\mathbf{A}_{WC}$, $\mathbf{b}_{WC}$ and $\mathbf{c}_{WC}^T$. The input and output activation functions are $\mathbf{f}_C$ and h_C, respectively. The internal controller state vector is denoted by $\mathbf{z}_{fC}$. Thereby, r denotes the reference.

To this point, the closed-loop recurrent network shown in Figure 4.8 can be seen as a MLP network with four layers. The first layer is given by the function $\mathbf{f}_C(\cdot)$. The input to this layer is the signal $r(k) - z(k)$. The second layer is given by the function $h_C(\cdot)$, which generates the input signal $u(k)$. The function $\mathbf{f}_P(\cdot)$ constitutes the third layer, which updates the states of the plant. The network output is given by the last layer, namely $h_P(\cdot)$.

The computation of the network output z is given through the back-substitution in the equations (4.24), (4.25), (4.26) and (4.27). Assume first that the reference signal r is provided. In this case, the output of the network is fully characterized through the signal r and the controller weigths $\mathbf{A}_{WC}$, $\mathbf{b}_{WC}$ and $\mathbf{c}_{WC}$, which will be denoted for simplicity by the matrix

$$\mathbf{W}_C = \begin{bmatrix} \mathbf{A}_{WC} & \mathbf{b}_{WC} \\ \mathbf{c}_{WC}^T & \mathbf{0} \end{bmatrix} \tag{4.28}$$

and the plant weigths $\mathbf{A}_{WP}$, $\mathbf{b}_{WP}$ and $\mathbf{c}_{WP}$ given by

$$\mathbf{W}_P = \begin{bmatrix} \mathbf{A}_{WP} & \mathbf{b}_{WP} \\ \mathbf{c}_{WP}^T & \mathbf{0} \end{bmatrix}. \tag{4.29}$$

The representations of the controller and plant matrices (4.28) and (4.29) are introduced to make the understanding of our approach easier. Now assume that a desired output

signal y_d is provided. Moreover, the goal is to fit the network output z to y_d. To achieve this goal, the following mean square representation

Problem 4.2 :

$$\underset{\mathbf{W}_C,\mathbf{W}_P}{\text{minimize}} \quad \underbrace{\frac{1}{N}\sum_{k=0}^{N}(y_d(k)-z(k,\mathbf{W}_C,\mathbf{W}_P))^2}_{J(\mathbf{W}_C,\mathbf{W}_P)=e^2(k)} \tag{4.30}$$

is used. The cost function $J(\mathbf{W}_C, \mathbf{W}_P)$ is formulated in terms of the controller matrices given by $\mathbf{W}_C$ and the plant matrices given by $\mathbf{W}_P$. Minimizing this cost function can be classified into two approaches, which are summarized below.

Plant Identification

In this case, the controller is known. A model of the plant using the reference and the measured output signal has to be identified. The plant can be linear as well as nonlinear and has to be identified. The update of the plant weights with respect to the cost function (4.30) can be performed as follows

$$\begin{bmatrix}\mathbf{W}_C^0\\ \mathbf{W}_P^{k+1}\end{bmatrix} = \begin{bmatrix}\mathbf{W}_C^0\\ \mathbf{W}_P^{k}\end{bmatrix} - \left(\begin{bmatrix}\mathbf{0}_{m\times n}\\ g(\mathbf{W}_P^k)\end{bmatrix}\right). \tag{4.31}$$

This means that the value of the controller matrix $\mathbf{W}_C$ is not updated. It is set to the known controller matrix given by $\mathbf{W}_C^0$. To update the current plant matrix $\mathbf{W}_P^k$, first or second order informations based on the first or second derivative of the cost function in the matrix $\mathbf{W}_P^k$ given by the function $g(\mathbf{W}_P^k)$ are used.

This idea was successfully applied to identify an unstable nonlinear model in closed-loop with a known controller, (Lachhab et al. 2008).

Controller Training

In this case, the plant is known. A controller has to be optimized to provide a specified performance. The controller can be linear as well as nonlinear. The optimization of the controller weights with respect to the cost function (4.30) can be performed as follows

$$\begin{bmatrix}\mathbf{W}_C^{k+1}\\ \mathbf{W}_P^0\end{bmatrix} = \begin{bmatrix}\mathbf{W}_C^{k}\\ \mathbf{W}_P^0\end{bmatrix} - \left(\begin{bmatrix}g(\mathbf{W}_C^k)\\ \mathbf{0}_{m\times n}\end{bmatrix}\right). \tag{4.32}$$

This means that the value of the plant matrix $\mathbf{W}_P$ is not updated. It is set to the known plant matrix given by $\mathbf{W}_C^0$. To update the current controller matrix $\mathbf{W}_P^k$, first or second order informations based on the first or second derivative of the cost function in the matrix $\mathbf{W}_C^k$ given by the function $g(\mathbf{W}_C^k)$ are used.

This idea is the core contribution of this work. It was successfully used to design a robust controller for a MIMO system. Moreover, the controller was implemented on the real plant.

4.3 LTI-Neural Network Controllers

In this section, the robust controller problem is presented. The related closed-loop recurrent neural network is discussed. For this purpose, the structure shown in Figure 4.9 is considered. This structure constitutes a special case of the closed-loop network shown in Figure 4.8. All activation functions are set linear. It means that

$$\mathbf{f}_C(\mathbf{x}) = \mathbf{f}_P(\mathbf{x}) = \mathbf{x} \quad \text{and} \quad h_C(x) = h_P(x) = x \tag{4.33}$$

with x and $\mathbf{x}$ denotes the scalar and vector input, respectively. The linear recurrent neural

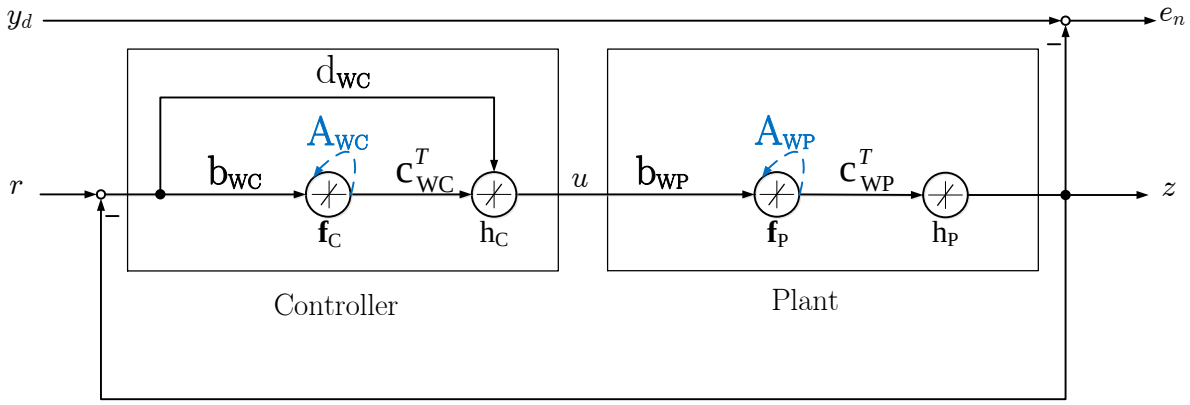

Figure 4.9: Closed-loop recurrent neural network (linear structure)

network describing the plant is given by

$$\begin{aligned} \mathbf{z}_{fP}(k+1) &= \mathbf{A}_{WP} \cdot \mathbf{z}_{fP}(k) + \mathbf{b}_{WP} \cdot u(k) \\ z(k) &= \mathbf{c}_{WP}^T \cdot \mathbf{z}_{fP}(k). \end{aligned} \tag{4.34}$$

The network matrices $\mathbf{A}_{WP}$, $\mathbf{b}_{WP}$ and $\mathbf{c}_{WP}^T$ consist the plant state-space matrices, which are fixed and will not be updated during the network training stage. In Section 3.4, the design of a robust PD_f controller for the electronic throttle was considered. In this section, the design of a LTI neural network controller for the same plant is considered. For this reason, these matrices $\mathbf{A}_{WP}$, $\mathbf{b}_{WP}$ and $\mathbf{c}_{WP}^T$ are set to the system matrices $\mathbf{A}$, $\mathbf{b}$ and $\mathbf{c}_0$ given by the expression (3.12) in Section (3.3.1). In this section, we consider the design of a robust state-space controller defined as follows

$$\begin{aligned} \mathbf{z}_{fC}(k+1) &= \mathbf{A}_{WC}\, \mathbf{z}_{fC}(k) + \mathbf{b}_{WC}\, (r(k) - y(k)) \\ u(k) &= \mathbf{c}_{WC}^T\, \mathbf{z}_{fC}(k) + d_{WC}\, (r(k) - y(k)). \end{aligned} \tag{4.35}$$

One of the main benefit of using the closed-loop recurrent structure is that the order of the controller can be defined independently of the plant order. This provides the user the freedom to define the controller order. In the case of the electronic throttle, a second

order controller is chosen. This results in the controller matrices $\mathbf{A}_K \in \mathbb{R}^{2\times 2}$, $\mathbf{b}_K \in \mathbb{R}^{2\times 1}$, $\mathbf{c}_K^T \in \mathbb{R}^{1\times 2}$ and $d_K \in \mathbb{R}$.

After defining the controller structure (4.35), the desired closed-loop performance with this controller is defined. This is given in terms of a desired crossover frequency ω_c^d, which is equal to 10 rad/s. Moreover, the controller has to provide a step response without any overshoot for all variations of the uncertain parameter V of the electronic throttle. It was shown in Section 3.2 that this performance is equivalent to specifying a desired closed-loop response $T_d(z)$ defined by the expression (3.32) in Section (3.3.3). Thereby, the open-loop function is given in terms of an integrator with a gain equal to ω_c^d. Moreover, the neural network plant output z is required to fit the step response y_d of the transfer function $T_d(z)$.

Set now the matrices $\mathbf{W}_{WC}$ and $\mathbf{W}_{WP}$ in the expressions and as follows

$$\mathbf{W}_C = \begin{bmatrix} \mathbf{A}_{WC} & \mathbf{b}_{WC} \\ \mathbf{c}_{WC}^T & d_{WC} \end{bmatrix} \tag{4.36}$$

and the plant weigths $\mathbf{A}_{WP}$, $\mathbf{b}_{WP}$ and $\mathbf{c}_{WP}$ given by

$$\mathbf{W}_P = \begin{bmatrix} \mathbf{A}_{WP} & \mathbf{b}_{WP} \\ \mathbf{c}_{WP}^T & \mathbf{0} \end{bmatrix}, \tag{4.37}$$

the overall optimization problem can be set. The control minimization problem can be defined as follows

Problem 4.3 :

$$\underset{\mathbf{W}_C}{\text{minimize}} \quad \underbrace{\frac{1}{N}\sum_{k=0}^{N}(y_d(k) - \mathbf{z}(k, \mathbf{W}_C, \mathbf{W}_P^0))^2}_{J(\mathbf{W}_C, \mathbf{W}_P^0) = \frac{1}{N}\sum_{k=0}^{N} e^2(k)} . \tag{4.38}$$

The formulation above defines an optimization problem in the matrix $\mathbf{W}_C$. Thereby, the matrix $\mathbf{W}_P^0$ is fixed to the plant state-space matrices given by $\mathbf{W}_P$. This optimization problem is quadratic in the error $e(k)$, but nonconvex in the controller matrices $\mathbf{W}_C$. The suboptimal controller matrix $\mathbf{W}_C^{opt}$ can be computed using the update rule (4.32). Based on the gradient and Hessian of the cost function $J(\mathbf{W}_C, \mathbf{W}_P^0)$, the update function $g(W_C^k)$ can be computed. Two approaches can be used to achieve this gaol. The first method is to use the Steepest-Descent search, which is based on the gradient of the cost function. The second method is to use the Newton method based on second order information. Another approach, which is more convenient to minimize the nonconvex problem (4.38) can be used: it is given by the Levenberg-Marquadt method, which is a combination of the Newton and the Steepest-Descent methods. Details about this approach is given in the Appendix D.

Remark. *The main benefit of using the structure shown in Figure 4.9 is that the resulting controller is linear and time-invariant. This means that the stability of the closed-loop system can be easily checked using known methods for LTI systems.*

To train the controller, the structure given by Figure 4.9 is built and trained using the Matlab Neural Network Toolbox. The algorithm used to train the controller is the Levenberg-Marquadt algorithms.

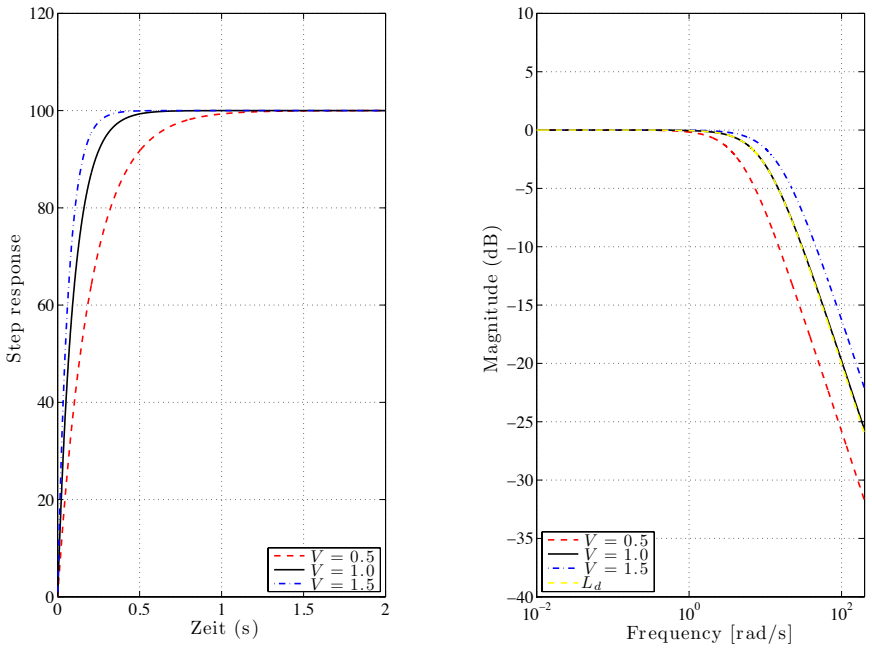

Figure 4.10: Closed-loop step response (left) and Bode magnitude of the desired (yellow dashed) and achieved closed-loop response for $V = 0.5$ (dashed red), $V = 1.0$ (black) and $V = 1.5$ (dashdot blue)

Figure 4.10 shows the obtained time and frequency response with the obtained controller. The closed-loop step response does not shows any overshoot for all values of the uncertain parameter V. This is confirmed through the closed-loop frequency response. Variations in the parameter V causes a shifting of the closed-loop bandwidth. Moreover, at low frequencies the amplitude of the Bode magnitude is for all values of V one. The robustness requirements are in this case achieved. The neural network controller provides the required performance in the presence of parameter variations.

Now comparing the achieved performance with the NN controller and the PD_f controller in Section 3.4, one can observe that there is approximately no difference. For the simplified model of the electronic throttle (B.48), both controllers satisfy the robustness requirements, except that the RNN provides a more accurate crossover frequency 9.95 rad/s

Table 4.1: Comparison of the achieved performance (LMI/RNN controllers)

Controller	LMIs	RNN	Desired Performance
Overshoots in (%)	0%	0%	0%
Crossover frequency	9.6%	9.95 rad/s	10 rad/s

in comparison to the desired crossover frequency 10 rad/s. Nevertheless, the NN based approach in this section has a main advantage over the LMI based method which is the structure of the controller. It means that the NN controller is not restricted to be PID as it is the case for the LMI based controller. RNN can build any state-space LTI controller to be used. However, for a given system one can first optimize a PID controller using the method in Section 3. In case that related LMI problem (3.51) in Section (3.4) is not feasible, one can compute a RNN controller using the technique in this section.

4.4 Summary

In this chapter, a method to optimize robust controllers using recurrent neural networks was presented. First of all, Section 4.2, feedforward and recurrent neural network are revised. Moreover, the training of a feedforward NN to fit the output of a nonlinear function has been discussed. In what concerns RNNs, a novel structure has been pretend specific to be applied to closed-loop system. Thereby, it consists of two RNNs, which are used to separately represent the plant and the controller. Given a model of the plant, a RNN controller can be trained to achieve a specified performance. In Section 4.3, this performance is specified. It is given in term of a desired step response. This performance is guaranteed in case of variations of the plant static gain. To show the effectiveness of the proposed approach, a robust LTI controller was computed for the electronic throttle. Simulation results have shown that the required performance is achieved.

5 Parallel Hybrid Electrical Vehicles

5.1 Introduction

The application of the design methods proposed in Section 4 and 3 is considered. It will be used to design linear discrete-time robust controllers for a MIMO system. The method given in Section 2 regarding the design of fractional controllers is not suitable to be used here. This is due the fact that we are interested in this section into designing a robust controller, which achieves an overshoot free step response. Moreover, as the controller will be implemented on the real plant, it is always desired to use a low order controller. For this purpose, the LMI based PID controller and the RNN (second order controller) are more appropriate to be used.

The plant under consideration is a hybrid electrical vehicle. Precisely, it is a parallel hybrid electrical vehicle in which the electrical machine and the ICE are placed on the same shaft. This makes a pure electrical drive possible. In this case, the EM drives the vehicle. In some cases, it is necessary to start the ICE. This is achieved through a separation clutch. It enables the coupling and decoupling of the ICE from the rest of the drivetrain. Such a process can be represented as a MIMO system.

In this chapter, the modeling of the PHE test vehicle is considered. It is presented in Section 5.3. Based on measurement signals a continuous-time identification is performed to define a MIMO model of the synchronization process. Thereby, the nonlinear characteristic of the clutch dynamic is considered as an uncertainty in the modeling. In Section 5.4, the design of robust controllers is presented. First, the application of the method based on LMI and MPC is discussed. Then, a linear robust controller is trained in closed-loop to achieve the robustness requirements. At the end of this chapter, simulation as well as experimental results are presented and discussed.

5.2 Classification of HEVs

The aim of this brief summary is to introduce the topic of HEVs to the reader. The mostly used definitions of hybrid electric vehicles are given. Moreover, the classification of HEVs into serial, parallel and power split is provided. Depending on the position of the ICE on the drivetrain, a HEV can be classify into one of these structures.

The discussion provided in this section is based on the definitions, classifications and structures provided in (Reif 2010), (Reif and Noreikat 2012), (Hofmann 2014) and (Ehsani et al. 2009). Figure 5.1 shows an overall description of HEVs. The following configurations are possible:

- **Serial Hybrid**. The main characteristic of this configuration is the presence of three components, namely combustion engine, electrical machine and generator. In this configuration, the ICE is used to charge the battery through the generator.

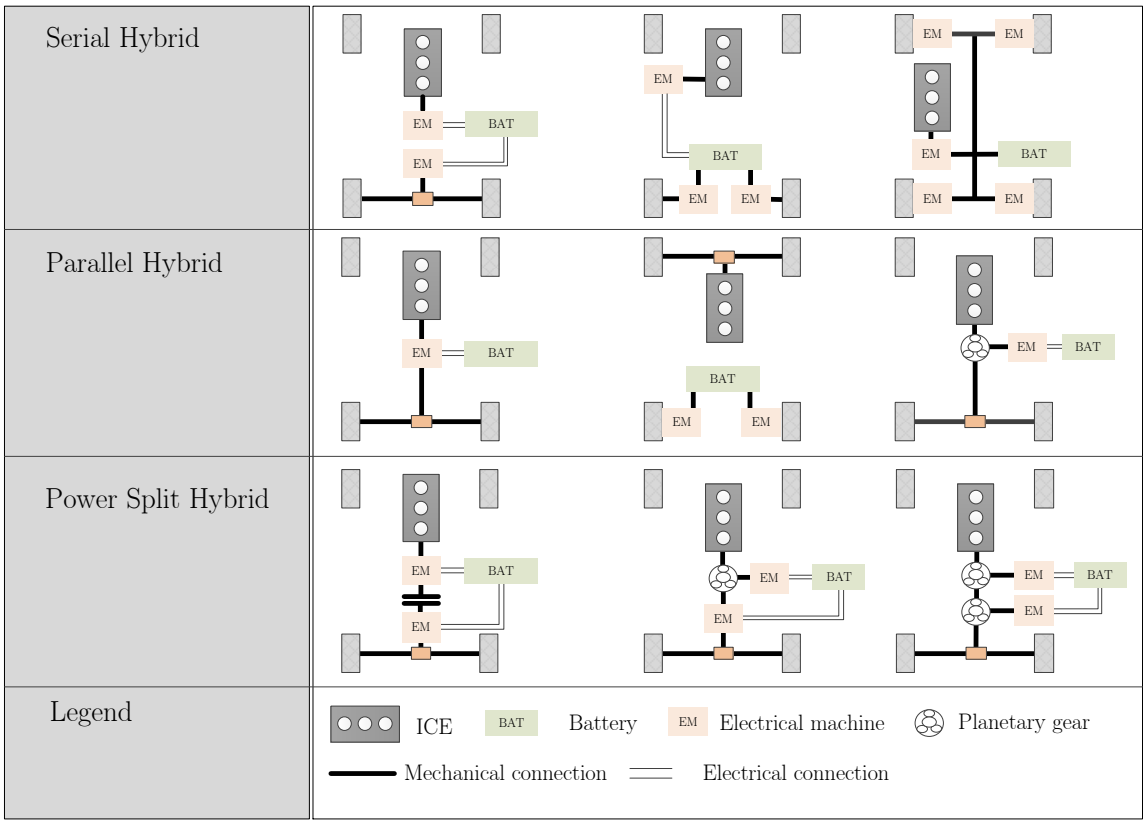

Figure 5.1: Classification of HEVs. Source: (Hofmann 2014, p. 24)

The vehicle is driven using an Electrical Motor (EM). Depending on the number and position of the used EMs, it exists three configurations:

- **One-Motor Hybrid**. This structure consists of one generator used to charge the battery and one motor that drives the vehicle using the energy stored in the battery.
- **Tandem-Motor Hybrid**. In this case, two motors are used to drive the vehicle through the left and right back wheels.
- **Hub-Wheel-Motor Hybrid**. In this configuration, four motors are used to drive the vehicle. All motors are taking the needed energy from the battery. A fifth motor is used as a generator to charge the battery.

- **Parallel Hybrid**. A common characteristic of PHEVs is that both the EM and the ICE can be combined to drive the vehicle. The ICE and the EM are placed on the same shaft. The EM can be used to assist the ICE during high torque phase (for example Boost). Depending on the type of combination, it differs between torque addition, speed addition or both.

  - **Torque-Coupling Hybrid**. In this case, the overall torque of the vehicle is the sum of the torque generated by the ICE and the torque of the EM.
  - **Speed-Coupling Hybrid**. Another combination of both machines is the speed-coupling. The speed of the vehicle is the sum of the ICE speed and EM speed.
  - **Torque-Coupling and Speed-Coupling Hybrid**. In this structure, torque-coupling and speed-coupling are possible.

- **Power Split Hybrid**. Each of the above configurations parallel and serial hybrid vehicles has advantages and disadvantages. In some cases, it is better to have the freedom to use both. This is given by the power split configuration. Within this configuration, it exists three hybrid structures.
  - **Combined**. This structure combines the parallel and serial architecture. In case that the clutch is open, this structure represents the serial hybrid. Closing the clutch makes the transition from serial to parallel hybrid configuration.
  - **1-Mode**. The main aspect of this mode is the use of the planetary gear. It splits the ICE path into two. The machine between the ICE and the battery is used to charge the battery. 1-Mode means that there is only one planetary gear to split the power (also known as input-split).
  - **2-Mode**. This structure contains two planetary gears. With this configuration, two types of power split can be realized, namely input-split and output-split.

The classification presented above is based on the structure of hybrid vehicles. Additionally, another classification depending on the size of the battery is made. It depends on the power available to drive the vehicle fully electrically. Generally, the following hybrid vehicle types are found in the literature:

- **Micro-Hybrid**. This type of hybrid vehicles is the first step towards the hybrid concept. It enables only simple functions as the start-stop and limited energy recuperation.
- **Mild-Hybrid**. The second degree of electrification of hybrid vehicles is the Mild-Hybrid. It enables the same functions as the Micro-Hybrid and limited boost with a middle sized electrical machine. Fully electrical drive is very limited possible.
- **Full-Hybrid**. This hybrid vehicle provides all the electrical hybrid functions such as start-stop, energy recuperation, electrical drive. Generally, vehicle velocity till 50 km/h is pure electrical possible. The range in which an electrical drive is possible is about 2 km.
- **Plug-In-Hybrid**. The extension to Full-Hybrid is given by this vehicle. Plug-In means that the battery can additionally be charged using a home or a commercial charging station. In such a vehicles, the battery is accordingly sized.

5.3 Modeling and Identification of a PHEV

In the last section, the classification of hybrid electrical vehicles were discussed. In this work, a Parallel Hybrid Electric Vehicle is considered. Moreover, it is a Full-Hybrid vehicle. Specifically, Torque-Coupling-Hybrid vehicle is treated here. Within this category, it exists another classification. Depending on the position of the EM on the drive train, it exists three structures, namely P1, P2 and P3. Figure 5.2 shows three parallel

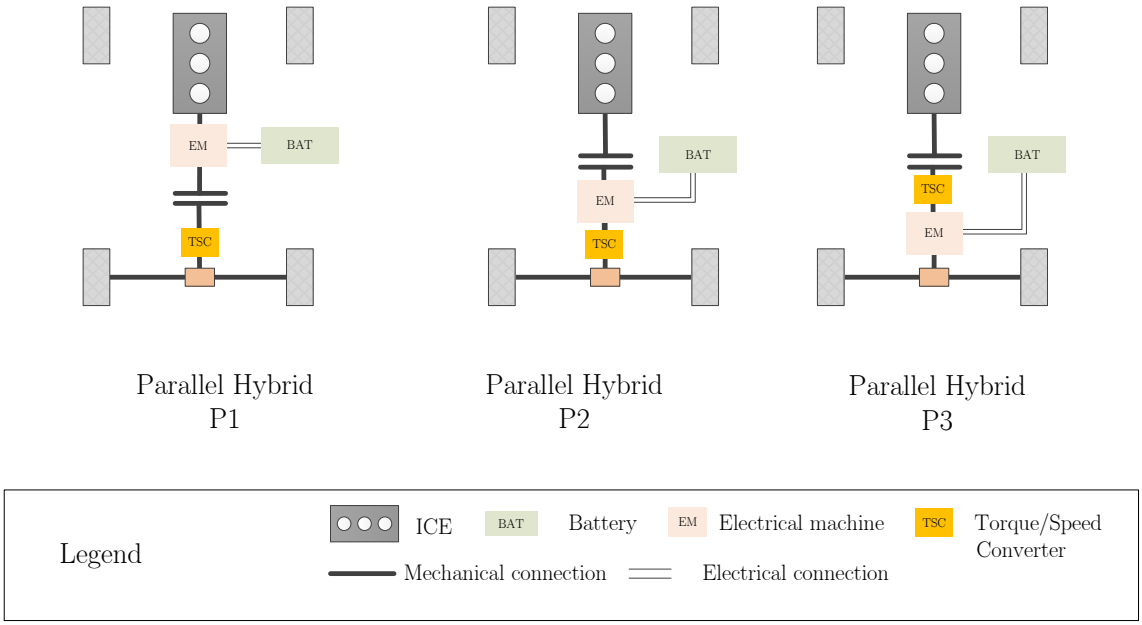

Figure 5.2: Classification of PHEVs

configurations. In (Reif and Noreikat 2012, p. 32), the following definition is given.

- **P-1 Parallel Hybrid**. In this situation, the EM is place between the ICE and the clutch. The EM is used here as a starter. In Boost mode, the EM is used to support the ICE in case of a high torque demand. A pure electrical drive using this configuration is not possible. This structure is denoted as Mild-Hybrid.

- **P-2 Parallel Hybrid**. The EM is placed behind the clutch. The main characteristic of this configuration is that the vehicle can be driven fully electrical. Moreover, all hybrid main functions are achieved with this configuration. It represents a Full-Hybrid vehicle.

- **P-3 Parallel Hybrid**. The EM is placed at the output of the converter. Electrical drive and recuperation with this configuration are possible. A advantage of the P3-PHEV is the preservation of the traction force.

In this work, we are concerned with the design of model-based robust controllers for the P-2 Parallel Hybrid configuration. For this purpose, it is mandatory to perform a system identification of the plant. Two main approaches can be taken into consideration. The first approach is based on the physical laws, differential equations can be set to describe a physical model of the plant. This approach was adopted and explored in (Jarczyk et al. 2009), (Alt et al. 2010) and (Alt et al. 2012). Unfortunately, the model based controller was test only in simulation. Moreover, the physical model and parameters are not experimentally validated. It is worth to be mentioned that the number of physical parameters needed is very high. The second approach, which can be used, is to obtain a model of the plant based on the plant response to some test signals. This method is known as system identification. This approach does not require a physical modeling of the plant. Moreover, one can choose between linear system identification (Ljung 1998), and nonlinear system identification (Nørgård 2000). In this context, one has to pay attention when deciding to use this identification method. An inaccurate choice of the mathematical model needed for system identification may lead to under- or overestimation of the dynamic of interest.

Another approach known as Grey-box model of the plant is to combine the physical informations about the system with the system response to specified input signals. A priori knowledge about the system to be identified is first gathered. Based on the obtained informations, the order and the structure of the model are defined. This is the approach adopted in this work.

A simplified model of the test vehicle drivetrain configuration is shown in Figure 5.3. In case of a pure electrical drive, the separation clutch is open. The unfired ICE is decoupled from the drivetrain. In this case, the EM has to provide the driver desired torque. In case that the driver torque demand is high, the ICE has to be fired and coupled to the drivetrain to provide this torque. This process is achieved by requesting the actuator of the separation clutch to track a predefined position trajectory up to fully closing the clutch. At the same time, the ICE angular velocity is pulled up to the EM angular velocity. The actual and desired clutch actuator position are internally converted into an actual and desired clutch torque. For this reason, the clutch can be seen as a torque source for the ICE. Actually, we are interested in the dynamic that relates the desired clutch torque $M_{CL,Des}$ and the desired EM torque $M_{EM,Des}$ to the angular velocities $\dot{\varphi}_{ICE}$ and $\dot{\varphi}_{EM}$. It is a multi-input multi-output dynamic with four transfer functions.

Combustion Engine Dynamic

The dynamic of the unfired engine is characterized through the transfer function from the actual clutch torque $M_{CL,Act}$ to the angular velocity of the engine $\dot{\varphi}_{ICE}$. Using the Toolchain (ETAS 2009a), (ETAS 2008b) and (ETAS 2009b) the related signals during a synchronization process are measured and recorded.
Figure 5.4 shows the step response of the system under consideration. The left hand side of this Figure shows the actual clutch torque $M_{CL,Act}$. The response to this torque results

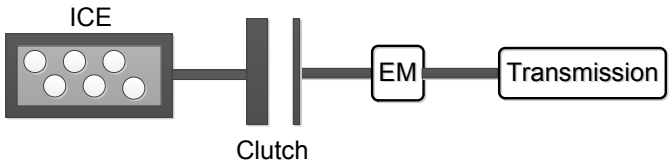

Figure 5.3: Simplified representation of the drivetrain configuration of a PHEV

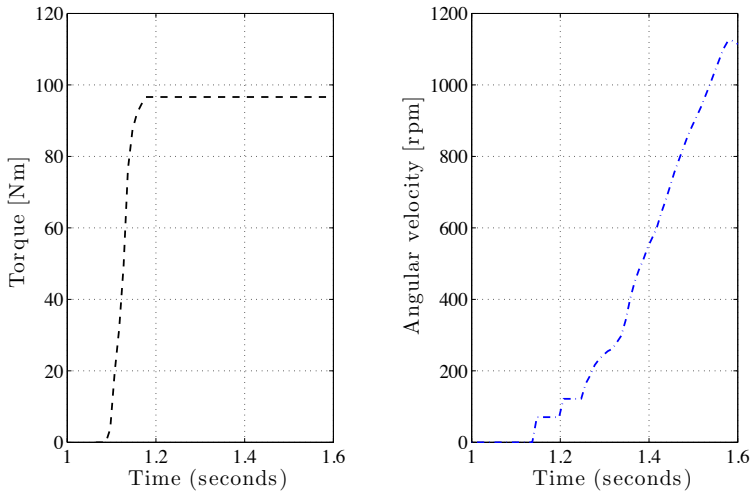

Figure 5.4: Actual clutch torque (left) and the related ICE angular velocity $\dot{\varphi}_{ICE}$ (right)

in pulling up the ICE. This is given by the right hand side of Figure 5.4. To catch the ICE dynamic, a model structure has first to be chosen. An intuitive simplified model that can be used is an integrator. This assumes that the dynamic of the ICE is linear and that the moment of inertia of the ICE plays the major role. In (Jarczyk et al. 2009), a second order model was used to catch this dynamic. It is based on the linearization of the physical model of the PHEV.

Considering now the trajectory of the angular velocity $\dot{\varphi}_{ICE}$, it can be divided into two phases namely, $\dot{\varphi}_{ICE} < 400\,rpm$ and $\dot{\varphi}_{ICE} > 400\,rpm$. In the first phase, the dynamic of the ICE is nonlinear and more complex. Especially, the trajectory of $\dot{\varphi}_{ICE}$ around $1.2\,sec$ have a dynamic similar to systems with delay part. For this reason, a non-minimum phase system (unstable zero / stable pole) is used. In the second phase, the dynamic of the combustion engine is approximately linear. In this case, an integrator with a gain is used. The overall second order transfer function

$$G_{ICE}(s) = \frac{b_{1_{ICE}} s + b_{0_{ICE}}}{a_{2_{ICE}} s^2 + a_{1_{ICE}} s} = \frac{\dot{\varphi}_{ICE}(s)}{M_{CL,Act}(s)} \tag{5.1}$$

is selected to catch the ICE dynamics in the whole range. After identifying the parameters of the transfer function (5.1), the obtained system is validated. Figure 5.5 shows

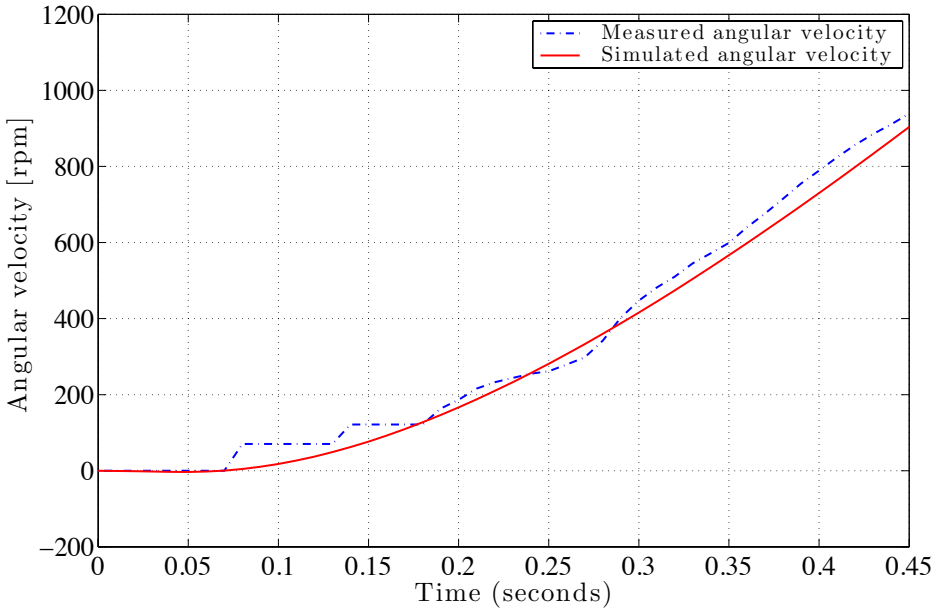

Figure 5.5: Validation of the identified ICE transfer function, measured ICE angular velocity $\dot{\varphi}_{ICE}$ (dashdot blue) and output of the identified function G_{ICE} (red)

the measured angular velocity of the ICE and the output of the identified model. The identified second order model provides about 71% fit. Thereby, the following formula

$$100\frac{(1-\|\mathbf{y}-\mathbf{y}_s\|_2)}{\|\mathbf{y}-y_m\|_2} \tag{5.2}$$

is used to calculate the fit, see the Matlab function *Compare*. Thereby, $\mathbf{y}$ and $\mathbf{y}_s$ are the measured and simulated output, respectively. y_m denotes the mean or average value. Augmenting the order or changing the structure of the transfer function (5.1) does not lead to a better fit. This is due to several nonlinear effects concerning the dynamic of the ICE. Nonlinear identification methods can be used to get a more accurate model. In this work, however this uncertainty in the identification of the system is compensated using a robust controller. This avoids the identification of a complex nonlinear models. Moreover, the linear control methods proposed in Section 3 and 4 can be applied.

Electrical Machine

During a pure electrical drive, the only source of torque is the electrical machine. It provides the desired driver torque. In this case, the driver requests a specified torque through the pedal position, which is internally converted into a desired torque value. This value plays the role of the reference trajectory. The EM torque should track this

torque trajectory. This is ensured using an internal EM torque controller. Figure 5.6 shows the response of the closed-loop system (desired EM torque $M_{EM,Des}$ to actual EM torque $M_{EM,Act}$) using this controller. The goal is to identify the transfer function from the $M_{EM,Des}$ to $M_{EM,Act}$.

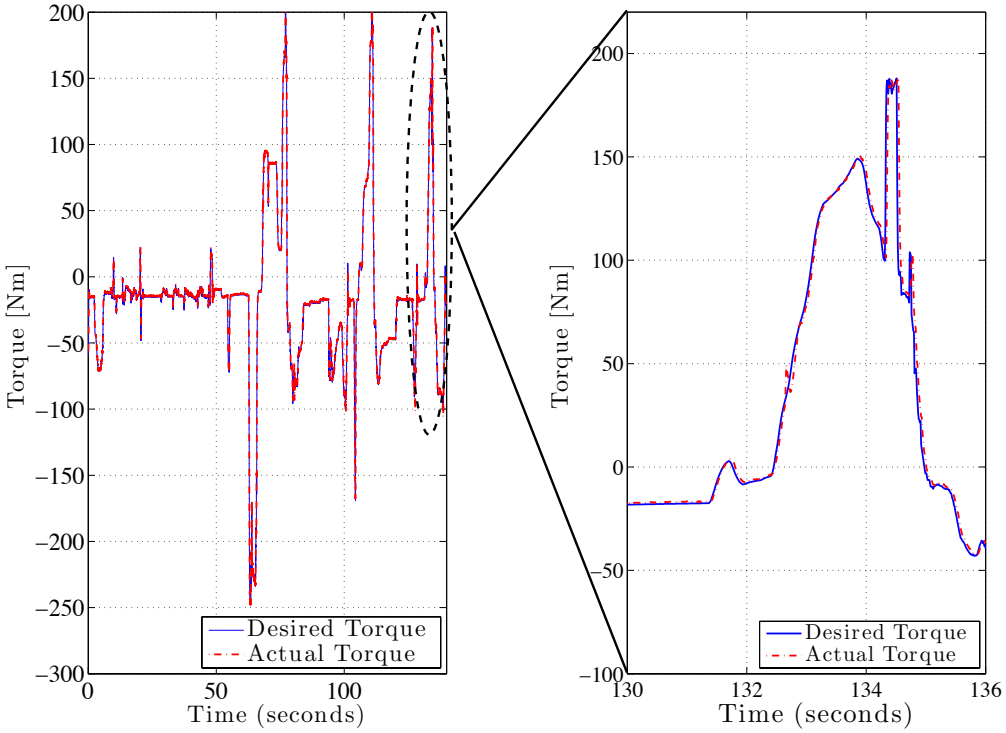

Figure 5.6: Actual and desired torque of the EM, $M_{EM,Act}$ (dashdot red) and $M_{EM,Des}$ (blue)

Now comparing $M_{EM,Des}$ and $M_{EM,Act}$ shown in Figure 5.6, one can notice that the torque of the electrical machine tracks the desired torque for the given trajectory. As it is well known for closed-loop systems, it is also desired to have a good tracking performance up to a specified bandwidth. Based on this assumption, a first order function is used to catch this dynamic. Specifically, a low-pass transfer function is used. Due to the CAN communication this system suffers from a time-delay of about 20 to 30 ms. In this case, the required model is given by

$$G_{EM}(s) = \frac{b_{0_{EM}}}{s + a_{0_{EM}}} e^{-\tau s} = \frac{M_{EM,Act}(s)}{M_{EM,Des}(s)}. \tag{5.3}$$

Now the goal is to identify the parameters $b_{0_{EM}}$, $a_{0_{EM}}$ and τ of the transfer function (5.3). The time-delay is given by the CAN-communication time and varies between 20 ms and 30 ms. The identification problem consists of the identification of the parameters $b_{0_{EM}}$

and $a_{0_{EM}}$. This is achieved using the Matlab Identification Toolbox. In Figure 5.7 the measured and simulated output of the identified transfer function signals are presented. The output of the identified model perfectly fits the measured data. Using the formula (5.2), the obtained fit is 97%.

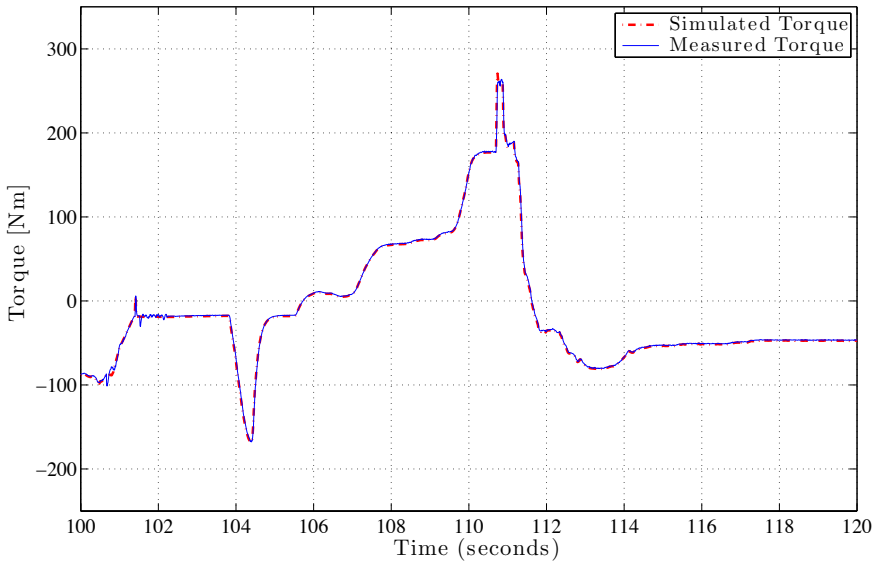

Figure 5.7: Validation of the identified EM transfer function, measured EM torque $M_{EM,Act}$ (blue) and output of the identified function G_{EM} (dashdot red)

Separation Clutch

After identifying the closed-loop dynamic of the EM torque, the torque dynamic of the clutch is considered. During a pure electrical drive, the separation clutch is open. It separates the ICE from the EM. The coupling and the decoupling of the clutch is achieved using an electromechanical actuator. The dependency between the actuator position and the transmitted torque is given by look-up tables. The dynamic of the clutch is mainly governed by the actuator controller. This ensures that the clutch actuator follows the position trajectory. Figure 5.8 shows a typical trajectory for the desired torque of the clutch $M_{CL,Des}$ with the related actual clutch torque $M_{CL,Act}$. In case that $M_{CL,Act}$ is equal to 0 Nm, then the clutch is open. In this case, no torque can be transmitted. In case that the actual clutch torque is equal to 600 Nm, then the clutch is fully closed. The clutch is then locked. The dynamic of interest is the transfer function from the desired clutch torque $M_{CL,Des}$ to the transmitted clutch torque $M_{CL,Act}$. The closed-loop dynamic of the clutch is approximated by a first order transfer function. Due to the CAN communication this system also suffers from a time-delay of 20 to 30 ms.

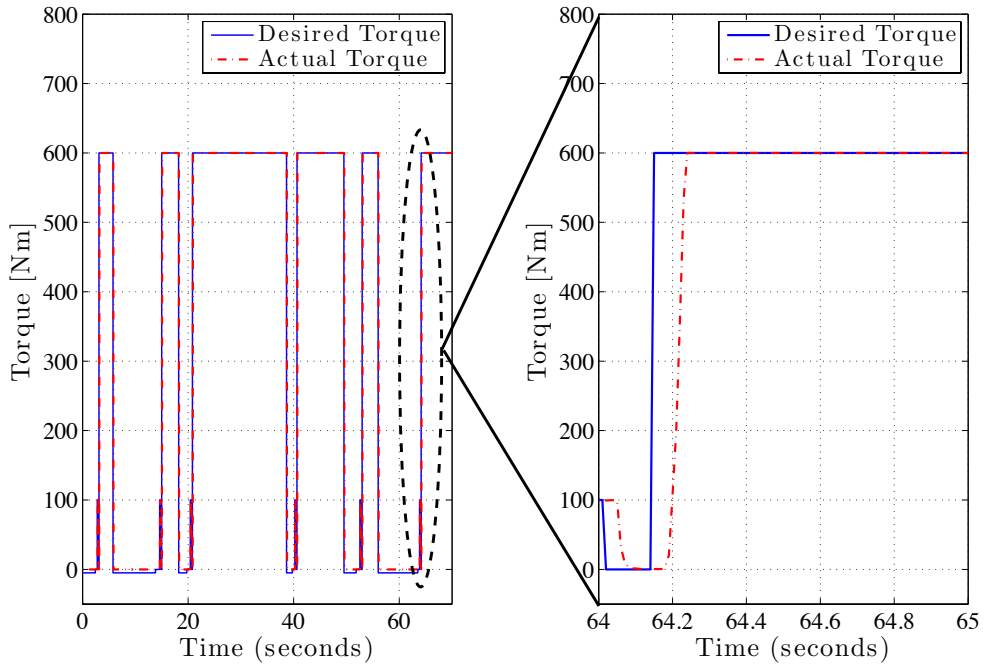

Figure 5.8: Actual and desired torque of the clutch, $M_{EM,Act}$ (dashdot red) and $M_{EM,Des}$ (blue)

The obtained model is given by

$$G_{CL,0}(s) = \frac{b_{0_{CL}}}{s + a_{0_{CL}}} e^{-\tau s} = \frac{M_{CL,Act_R}(s)}{M_{CL,Des}(s)}. \tag{5.4}$$

In Figure 5.9, the measured and simulated signals are presented. The output of the identified model provides a good fit of the measured data. Using the output formula (5.2), the obtained fit is 95%. At this point, we want to mention that the transfer function $G_{CL,0}(s)$ is the nominal transfer function. This is due to the fact that the clutch consists the main source of uncertainty in the modeling of the whole synchronization process. The identification of this model is based on the assumption that the real clutch torque M_{CL,Act_R} is equal to the internally calculated torque $M_{CL,Act}$, which is the output of the transfer function $G_{CL,0}$. Generally, this is not true: it means that $M_{CL,Act_R} = V \cdot M_{CL,Act}$. Replacing this in the expression (5.4) results in the following

$$G_{CL}(s) = V \cdot G_{CL,0}(s) = V \cdot \frac{b_{0_{CL}}}{s + a_{0_{CL}}} e^{-\tau s}. \tag{5.5}$$

It means that the uncertainty in the clutch torque is represented as a multiplicative output uncertainty. This is given by the uncertain static gain V. The representation of the clutch

dynamic by the uncertain function (5.5) is based on the assumption that the variations of the parameter V are very slow. It means that the rate of change given by $\dot{V}(t)$ is negligible.

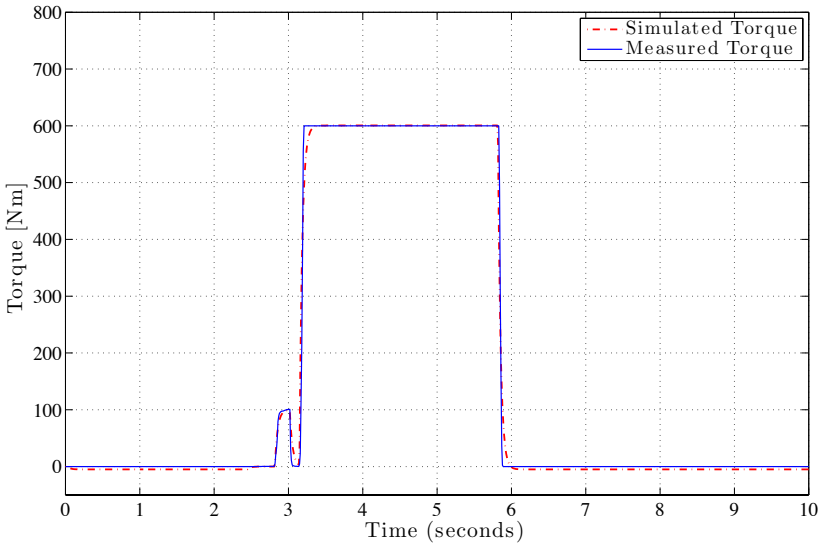

Figure 5.9: Validation of the identified clutch transfer $G_{CL,0}$ function, measured clutch torque $M_{CL,Act}$ (blue) and simulated clutch torque (dashdot red)

Drivetrain Rotational Dynamics

The drivetrain rotational dynamics is responsible for transforming the actual EM torque into a rotation. In case that the separation clutch is open, this dynamic is represented by the transfer function from the actual EM torque $M_{EM,Act}$ to the actual EM angular velocity $\dot{\varphi}_{EM}$. The signals used to identify the model are presented in Figure 5.10. In (Jarczyk et al. 2009), the modeling of the rotational dynamics of the EM have discussed. Based on the nonlinear differential equations of the powertrain of the PHEV, a linearization was performed which leads to a fourth order transfer function. For this purpose, the following transfer function

$$G_{ROT}(s) = \frac{b_{3_{ROT}}s^3 + b_{2_{ROT}}s^2 + b_{1_{ROT}}s + b_{0_{ROT}}}{s^4 + a_{3_{ROT}}s^3 + a_{2_{ROT}}s^2 + a_{1_{ROT}}s} = \frac{\dot{\varphi}_{EM}(s)}{M_{EM,Act}(s)}. \tag{5.6}$$

is used.

In Figure 5.11, the output of the identified model G_{ROT} and the measured angular velocity $\dot{\varphi}_{EM}$ are shown. The output of the identified model approximately fits the measured data. Using the formula (5.2), the obtained fit is 96%. Augmenting the order of the transfer function 5.6 does not lead to a better fit.

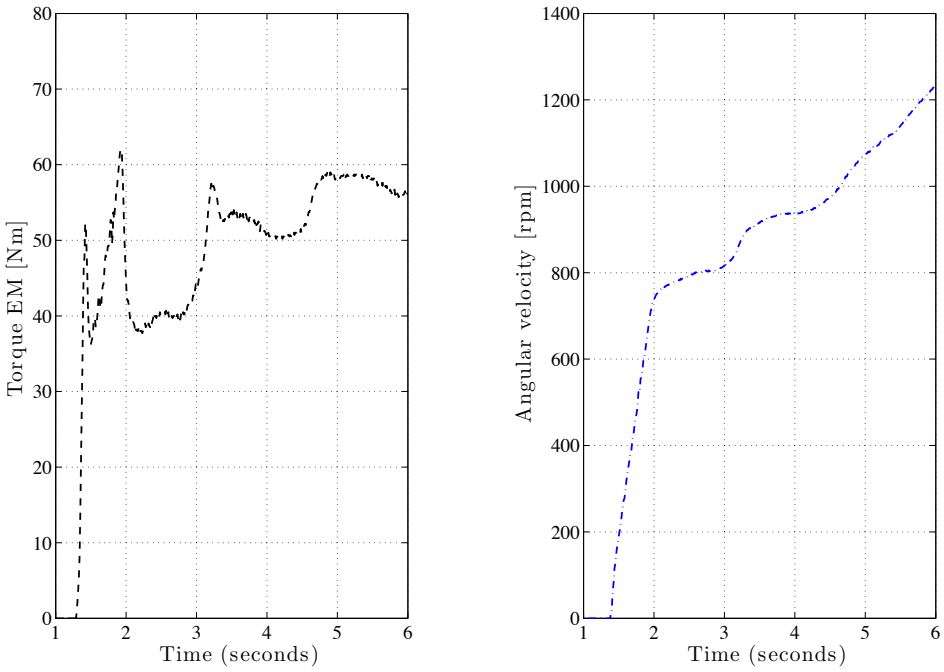

Figure 5.10: Actual EM torque $M_{EM,Act}$ (dash black) and EM angular velocity $\dot{\varphi}_{EM}$ (dashdot)

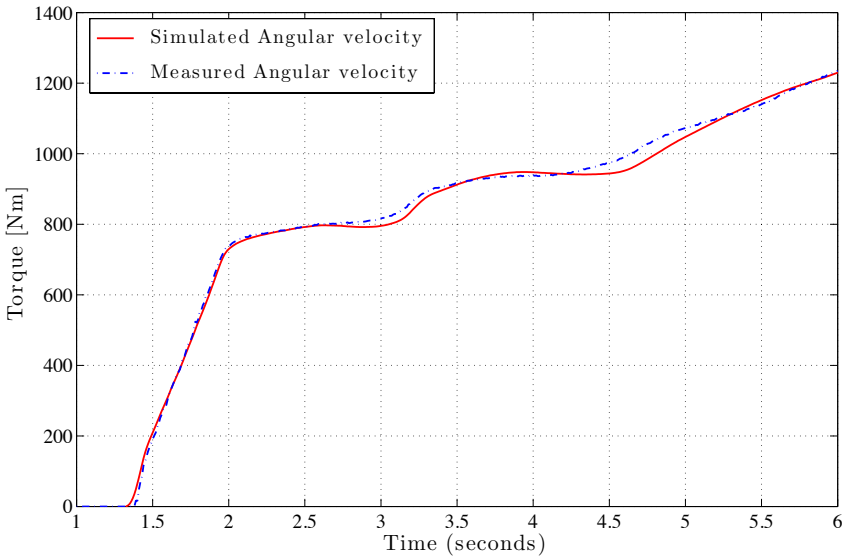

Figure 5.11: Validation of the identified drivetrain transfer function G_{ROT}, measured angular velocity $\dot{\varphi}_{EM}$ (dashdot blue) and simulated angular velocity (red)

Building MIMO Model

After identifying the four transfer functions, the overall MIMO model is built. It has two inputs, the desired EM torque $M_{EM,Des}$ and the desired clutch torque $M_{CL,Des}$ and two outputs, the angular velocities $\dot{\varphi}_{ICE}$ and $\dot{\varphi}_{EM}$. Using the identified transfer functions $G_{CL}(s)$, $G_{ICE}(s)$, $G_{EM}(s)$ and $G_{ROT}(s)$, the MIMO system is constructed. The structure of the model is presented in Figure 5.12, which was also considered in (Alt et al. 2010).

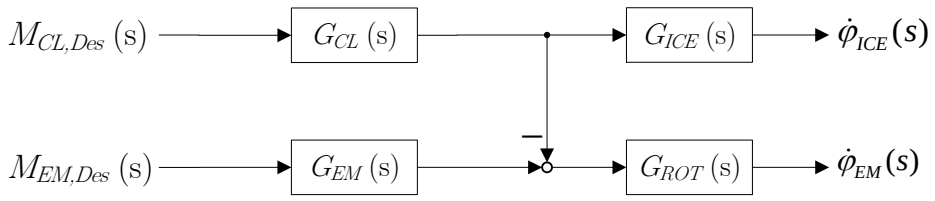

Figure 5.12: MIMO Model with identified transfer functions

The system dynamic is described as follows

$$\begin{aligned} \dot{\varphi}_{EM}(s) &= (M_{CL,Des}(s)G_{CL}(s) - M_{EM,Des}(s)G_{EM}(s))G_{ROT}(s) \\ \dot{\varphi}_{ICE}(s) &= M_{CL,Des}(s)G_{CL}(s)G_{ICE}(s) \end{aligned}$$

or in matrix form

$$\begin{bmatrix} \dot{\varphi}_{EM}(s) \\ \dot{\varphi}_{ICE}(s) \end{bmatrix} = \begin{bmatrix} G_{CL}(s)G_{ROT}(s) & -G_{EM}(s)G_{ROT}(s) \\ G_{CL}(s)G_{ICE}(s) & 0 \end{bmatrix} \begin{bmatrix} M_{CL,Des}(s) \\ M_{EM,Des}(s) \end{bmatrix}. \tag{5.7}$$

This MIMO system is a two by two coupled model. The angular velocity $\dot{\varphi}_{EM}$ depends on both inputs $M_{CL,Des}$ and $M_{EM,Des}$. The angular velocity $\dot{\varphi}_{ICE}$ depends only on the clutch torque $M_{CL,Des}$.

5.4 Design of Robust LTI-Controllers for the Synchronisation Task

In this section, the design of robust controllers to solve the synchronization task in PHEV is discussed. The approaches presented in Section 3 and 4 are considered, namely LMI- and RNN-based methods. Thereby, the MIMO Model (5.7) is used for this task. The requirements to be achieved by the controller are expressed in terms of a desired closed-loop response. Precisely, the goal is to synchronize the angular velocity of the ICE $\dot{\varphi}_{ICE}$ with the angular velocity of EM $\dot{\varphi}_{EM}$ taking into account the variations of the gain V of

the clutch model (5.5). Thereby, any deterioration in the performance of $\dot{\varphi}_{EM}$ should be avoided.

First of all, the controller structure has to be defined. The goal is to control the clutch and the electrical machine torques to minimize the error between the ICE and EM angular velocities. This means that the controller has one input $(\dot{\varphi}_{EM} - \dot{\varphi}_{ICE})$ and generates two outputs $M_{CL,Des}$ and $M_{EM,Des}$. It is a Single-Input Two-Outputs controller. Due to its simple structure, the design of a controller with two $PIDs$ is discussed. For this purpose, the approach presented in Section 3 based on MPC and LMIs is applied. Based on simulation, the performance of this controller is evaluated. To compare the performance of the PID controller, a neural network controller is used as a counterpart. Using the method presented in Section 4 based on RNN, the synchronization task is built as a neural network consisting of the identified MIMO modell and a neural network controller. Thereby, a second order controller with one input $(\dot{\varphi}_{EM} - \dot{\varphi}_{ICE})$ and two outputs, $M_{CL,Des}$ and $M_{EM,Des}$. The controller order is chosen, on one hand to keep the controller complexity low as it will be implemented on the test vehicle, on the other hand the dynamics of the plant can be well catched using second order systems. The optimized controller is compared with the obtained PID controllers in terms of achievement of the performance requirements. Simulation results are presented at the end of this chapter.

5.4.1 PD Controller

Due to the fact the dynamics of the angular velocities $\dot{\varphi}_{ICE}$ and $\dot{\varphi}_{EM}$ given by the transfer functions G_{ICE} and G_{ROT} contain an integrator, the design of PD controllers is considered here. For this reason, the configuration shown in Figure 5.13 is considered. Thereby, R is the reference signal. The input to the controller E is the difference between the angular velocity $\dot{\varphi}_{ICE}$ and the reference R. Y_d consists the desired response used to shape $\dot{\varphi}_{ICE}$. The value of the desired angular velocity of the EM at the time of synchronization is denoted by $\dot{\varphi}_{EM,Des}$.

The goal of the control design is twofold. On one hand, the controller should cope with the uncertainty in the modeling of the separation clutch (5.5). This is given through the variations of the static gain parameter V. On the other hand, the dynamic of the angular velocity $\dot{\varphi}_{ICE}$ and $\dot{\varphi}_{EM}$ should be decoupled. It means that the velocity $\dot{\varphi}_{ICE}$ should follows its reference R without a deterioration of the velocity $\dot{\varphi}_{EM}$.

Note here that the controller design is performed in discrete-time. For this reason, the related discrete-time transfer functions of the identified model (5.7) are used. These are denoted by discrete-time shift operator z. Thereby, the sampling time T_s is 10 ms.

To formulate the control specifications, consider the open-loop function in Figure 5.13

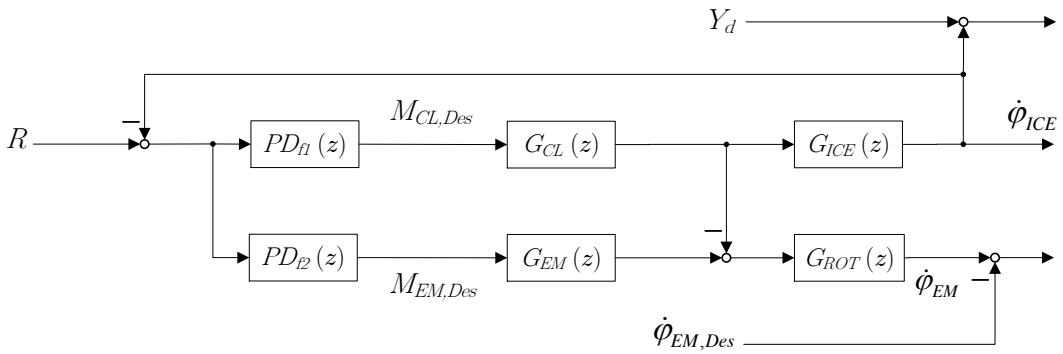

Figure 5.13: Closed-loop structure with MPC-controllers PD_{f1} and PD_{f2}

defined as follows

$$L_{CL}(z) = \frac{\dot{\varphi}_{ICE}(z)}{E(z)} = V \cdot \underbrace{PD_{f1}(z)G_{CL,0}(z)G_{ICE}(z)}_{L_{CL,0}(z)}. \tag{5.8}$$

The robustness in terms of the static gain variations can be achieved through the specification of a desired response for the nominal open-loop function $L_{CL,0}(z)$. As it is presented in Section 3.2, this requirement can be also achieved by shaping the nominal closed-loop response

$$T_{CL,0}(z) = \frac{\dot{\varphi}_{ICE}(z)}{R(z)} = \frac{PD_{f1}(z)G_{CL,0}(z)G_{ICE}(z)}{PD_{f1}(z)G_{CL,0}(z)G_{ICE}(z) + 1} \tag{5.9}$$

to the desired closed-loop response (3.32) given by

$$T_d(z) = \frac{1 - e^{-T_s\omega_c^d}}{z - e^{-T_s\omega_c^d}}, \tag{5.10}$$

which is equivalent to fitting the step response of the closed-loop function (5.9) in terms of $\dot{\varphi}_{ICE}$ to the step response of the desired function (5.10). Thereby, ω_c^d consists the desired bandwidth.

Additionally to the robustness requirement for static gain variations of the clutch model, the controller should also not deteriorate the trajectory of the angular velocity $\dot{\varphi}_{EM}$. During the synchronization process the angular velocity of the electrical machine should not break down to avoid a deterioration of the driver comfort. This is given by requiring $\dot{\varphi}_{EM}$ to be equal to the desired angular velocity $\dot{\varphi}_{EM,Des}$ during the synchronization process. For the seek of simplicity, we assume that the EM rotates with the desired velocity $\dot{\varphi}_{EM,Des}$ before the synchronization starts.

To apply the approach presented in Section 3.4, the problem matrices describing the synchronisation task required for the optimization problem (3.51) should be defined. Let first introduce the discrete-time state-space representation for the nominal case $V = 1$ of the clutch model, which is obtained through the discretization of the identifed MIMO continuous model (5.7) as follows

$$P : \begin{cases} \mathbf{x}_p(k+1) &= \mathbf{A}_p\, \mathbf{x}_p(k) + \mathbf{B}_p \underbrace{\begin{bmatrix} \mathbf{M}_{Des,CL}(k) \\ \mathbf{M}_{Des,EM}(k) \end{bmatrix}}_{\mathbf{M}_{Des}(k)} \\ \begin{bmatrix} \dot{\varphi}_{ICE}(k) \\ \dot{\varphi}_{EM}(k) \end{bmatrix} &= \underbrace{\begin{bmatrix} \mathbf{c}_{ICE}^T \\ \mathbf{c}_{EM}^T \end{bmatrix}}_{\mathbf{C}_P^T} \mathbf{x}_p(k). \end{cases} \tag{5.11}$$

with $\mathbf{A}_p \in \mathbb{R}^{8\times 8}$, $\mathbf{B}_p \in \mathbb{R}^{8\times 2}$, $\mathbf{c}_{ICE}^T \in \mathbb{R}^{1\times 8}$, $\mathbf{c}_{EM}^T \in \mathbb{R}^{1\times 8}$ and the state vector $\mathbf{x}_p \in \mathbb{R}^{8\times 1}$.

The angular velocities $\dot{\varphi}_{ICE}$ and $\dot{\varphi}_{EM}$ at successive time instant k are given by

$$\begin{aligned} \dot{\varphi}_{ICE}(k) = &\quad 0 \\ \dot{\varphi}_{ICE}(k+1) = &\quad 0 + \mathbf{c}_{ICE}^T \mathbf{B}_p\, \mathbf{M}_{Des}(k) \\ \dot{\varphi}_{ICE}(k+2) = &\quad 0 + \mathbf{c}_{ICE}^T \mathbf{A}_p \mathbf{B}_p\, \mathbf{M}_{Des}(k) \;+\; \mathbf{c}_{ICE} \mathbf{B}_p\, \mathbf{M}_{Des}(k+1) \\ \vdots = &\quad \cdots \\ \dot{\varphi}_{ICE}(k-1+N) = &\quad 0 + \mathbf{c}_{ICE}^T \mathbf{A}_p^{N-1} \mathbf{B}_p\, \mathbf{M}_{Des}(k) + \cdots \\ &\quad \cdots + \mathbf{c}_{ICE}^T \mathbf{B}_p\, \mathbf{M}_{Des}(k+N-2) \end{aligned} \tag{5.12}$$

and

$$\begin{aligned} \dot{\varphi}_{EM}(k) = &\quad \dot{\varphi}_{EM,Des} \\ \dot{\varphi}_{EM}(k+1) = &\quad \dot{\varphi}_{EM,Des} + \mathbf{c}_{EM}^T \mathbf{B}_p\, \mathbf{M}_{Des}(k) \\ \dot{\varphi}_{EM}(k+2) = &\quad \dot{\varphi}_{EM,Des} + \mathbf{c}_{EM}^T \mathbf{A}_p \mathbf{B}_p\, \mathbf{M}_{Des}(k) + \mathbf{c}_{EM}^T \mathbf{B}_p\, \mathbf{M}_{Des}(k+1) \\ \vdots = &\quad \cdots \\ \dot{\varphi}_{EM}(k-1+N) = &\quad \dot{\varphi}_{EM,Des} + \mathbf{c}_{EM}^T \mathbf{A}_p^{N-1} \mathbf{B}_p\, \mathbf{M}_{Des}(k) + \;\cdots \\ &\quad \cdots + \mathbf{c}_{EM}^T \mathbf{B}_p\, \mathbf{M}_{Des}(k+N-2). \end{aligned} \tag{5.13}$$

Remark. *The discrete-time state-space model obtained above was based on the identified transfer function with the time delay part. This is due to the fact that the approache proposed in this work considers only linear systems without time delay during the opti-*

mization step. The performance of the controller is tested afterwards based on robustness analysis with the time-delay model.

The parameter N represents the number of past samples of the desired response $y_d(k)$. Using now the following notation

$$\mathbf{M}_{Des} = \begin{bmatrix} \mathbf{M}_{Des,ICE}(k) \\ \mathbf{M}_{Des,EM}(k) \\ \vdots \\ \mathbf{M}_{Des,ICE}(k-1+N) \\ \mathbf{M}_{Des,EM}(k-1+N) \end{bmatrix} \tag{5.14}$$

to denote the control inputs in vector form, the outputs of the model given by $\dot{\varphi}_{ICE}$ and $\dot{\varphi}_{EM}$ in vector form can be presented. These are given as follows

$$\underbrace{\begin{bmatrix} \dot{\varphi}_{EM}(k) \\ \dot{\varphi}_{EM}(k+1) \\ \vdots \\ \dot{\varphi}_{EM}(k-1+N) \end{bmatrix}}_{\mathbf{n}_{EM}} = \mathbf{\Omega}_{EM}\ \mathbf{M} + \dot{\varphi}_{EM,Des} \tag{5.15}$$

and

$$\underbrace{\begin{bmatrix} \dot{\varphi}_{ICE}(k) \\ \dot{\varphi}_{ICE}(k+1) \\ \vdots \\ \dot{\varphi}_{ICE}(k-1+N) \end{bmatrix}}_{\mathbf{n}_{ICE}} = \mathbf{\Omega}_{ICE}\ \mathbf{M}. \tag{5.16}$$

The transition matrices $\mathbf{\Omega}_{ICE}$ and $\mathbf{\Omega}_{EM}$ are given by

$$\mathbf{\Omega}_{ICE} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ \mathbf{c}_{ICE}^T\mathbf{B}_p & 0 & 0 & \cdots & 0 \\ \mathbf{c}_{ICE}^T\mathbf{A}_p\mathbf{B}_p & \mathbf{c}_{ICE}^T\mathbf{B}_p & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 \\ \mathbf{c}_{ICE}^T\mathbf{A}_p^{N-2}\mathbf{B}_p & \mathbf{c}_{ICE}^T\mathbf{A}_p^{N-1}\mathbf{B}_p & \cdots & \mathbf{c}_{ICE}^T\mathbf{B}_p & 0 \end{bmatrix}$$

and

$$\mathbf{\Omega}_{EM} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ \mathbf{c}_{EM}^T\mathbf{B}_p & 0 & 0 & \cdots & 0 \\ \mathbf{c}_{EM}^T\mathbf{A}_p\mathbf{B}_p & \mathbf{c}_{ICE}^T\mathbf{B}_p & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 \\ \mathbf{c}_{EM}^T\mathbf{A}_p^{N-2}\mathbf{B}_p & \mathbf{c}_{EM}^T\mathbf{A}_p^{N-1}\mathbf{B}_p & \cdots & \mathbf{C}_{EM}^T\mathbf{B}_p & 0 \end{bmatrix}.$$

Note that the deviation of the angular velocity $\dot{\varphi}_{EM}$ is taken into consideration by minimizing the 2-norm

$$\|\dot{\varphi}_{EM,Des} - \mathbf{n}_{EM}\|_2^2 = \|\dot{\varphi}_{EM,Des} - \mathbf{\Omega}_{EM}\ \mathbf{M} - \dot{\varphi}_{EM,Des}\|_2^2, \tag{5.17}$$

which is equivalent to the norm $\|\mathbf{\Omega}_{EM}\ \mathbf{M}\|_2^2$. Before proceeding to the main result, the controller structure is discussed. It is given as follows

$$\tilde{\mathbf{M}}_{ICE,Des} = K_{P_1}(\mathbf{r} - \mathbf{y}_d) + K_{D_1}\mathbf{\Omega}_D(\mathbf{r} - \mathbf{y}_d)_f \tag{5.18}$$

and

$$\tilde{\mathbf{M}}_{EM,Des} = K_{P_2}(\mathbf{r} - \mathbf{y}_d) + K_{D_2}\mathbf{\Omega}_D(\mathbf{r} - \mathbf{y}_d)_f. \tag{5.19}$$

It is given by two PD_f controllers. These are defined through the parameters K_{P_1}, K_{D_1}, K_{P_2} and K_{D_2}. The vector $\mathbf{y}_d$ is the discrete-time step response of the reference transfer function (5.10). The vector $\mathbf{r}$ denotes the step command in vector notation. Additionally to the robust performance imposed on the PD_f controllers, the closed-loop response should be fast. This is achieved by specifying a value for the desired bandwidth ω_c^d. For values smaller than $4\,rad/s$, the LMI-based method provides a PD_f controller that achieves the required performance. For $\omega_c^d = 4\,rad/s$, the method fails to compute a robust controller. This was the motivation to use neural network based controllers, which achieves robustness in case of static gain variations for $\omega_c^d = 4\,rad/s$. For higher values, the neural network appraoch did not find any controller with an acceptable performance. In this section, the obtained performance with both controllers is presented.

Now, the proposed approach can be applied to optimize the controller parameters K_{P_1}, K_{D_1}, K_{P_2} and K_{D_2} the optimal input vector $\mathbf{M}_{Des}$.

The overall optimization problem in LMI form is given by

Problem 5.1 :

$$
\begin{aligned}
&\underset{K_{P_1},K_{D_1},K_{P_2},K_{D_2},\mathbf{M}_{Des}}{\text{minimize}} && (\gamma_1+\gamma_2+\gamma_3+\gamma_4)\\
&\text{subject to} && \begin{bmatrix} -\gamma_1 & (\mathbf{y}_d-\mathbf{\Omega}_{ICE}\mathbf{M}_{Des})^T \\ (\mathbf{y}_d-\mathbf{\Omega}_{ICE}\mathbf{M}_{Des}) & \mathbf{I}_N \end{bmatrix} < 0,\\
& && \begin{bmatrix} -\gamma_2 & (\Omega_{EM}\mathbf{M}_{Des})^T \\ (\Omega_{EM}\mathbf{M}_{Des}) & \mathbf{I}_N \end{bmatrix} < 0,\\
& && \begin{bmatrix} -\gamma_3 & (\mathbf{M}_{Des_{EM}}-\tilde{\mathbf{M}}_{Des_{EM}})^T \\ (\mathbf{M}_{Des_{EM}}-\tilde{\mathbf{M}}_{Des_{EM}}) & -\mathbf{I}_N \end{bmatrix} < 0,\\
& && \begin{bmatrix} -\gamma_4 & (\mathbf{M}_{Des_{CL}}-\tilde{\mathbf{M}}_{Des_{CL}})^T \\ (\mathbf{M}_{Des_{CL}}-\tilde{\mathbf{M}}_{Des_{CL}}) & -\mathbf{I}_N \end{bmatrix} < 0
\end{aligned}
\tag{5.20}
$$

The desired torques for the EM and the clutch in vector notation are given by

$$
\tilde{\mathbf{M}}_{Des,ICE} = \begin{bmatrix} \mathbf{M}_{Des,ICE}(k) \\ \mathbf{M}_{Des,ICE}(k+1) \\ \vdots \\ \mathbf{M}_{Des,ICE}(k-1+N) \end{bmatrix}, \tilde{\mathbf{M}}_{Des,EM} = \begin{bmatrix} \mathbf{M}_{Des,EM}(k) \\ \mathbf{M}_{Des,EM}(k+1) \\ \vdots \\ \mathbf{M}_{Des,EM}(k-1+N) \end{bmatrix} \tag{5.21}
$$

Problem (5.20) consists of computing two optimal input trajectories $\tilde{\mathbf{M}}_{Des,CL}$ and $\tilde{\mathbf{M}}_{Des,EM}$, which brings the angular velocity of the ICE up to the desired value given by the desired response $\mathbf{y}_d$. Thereby, the angular velocity $\dot{\varphi}_{EM}$ should not deteriorate. To solve this problem, the optimization software (Labit et al. 2002) is used. In this case, the computation of the optimal solution takes 18 *seconds*.

Figure 5.14 shows the closed-loop response with the obtained controller. Thereby, a constant desired angular velocity is set as a reference, namely 3000 *rpm*. The step response in terms of the angular velocity $\dot{\varphi}_{ICE}$ shows a maximal overshoot of 80 *rpm* in case of variations of the gain V. The robustness requirements are approximately achieved. On the right hand side of Figure 5.14, the step response in terms of the angular velocity $\dot{\varphi}_{EM}$ is given. A small deterioration in this velocity is shown. It is about 290 U/min or 10% for $V = 0.85$, 2% for $V = 1.0$ and -4% for $V = 0.85$. It is acceptable. Generally, the requirements imposed on the closed-loop system are partially satisfied.

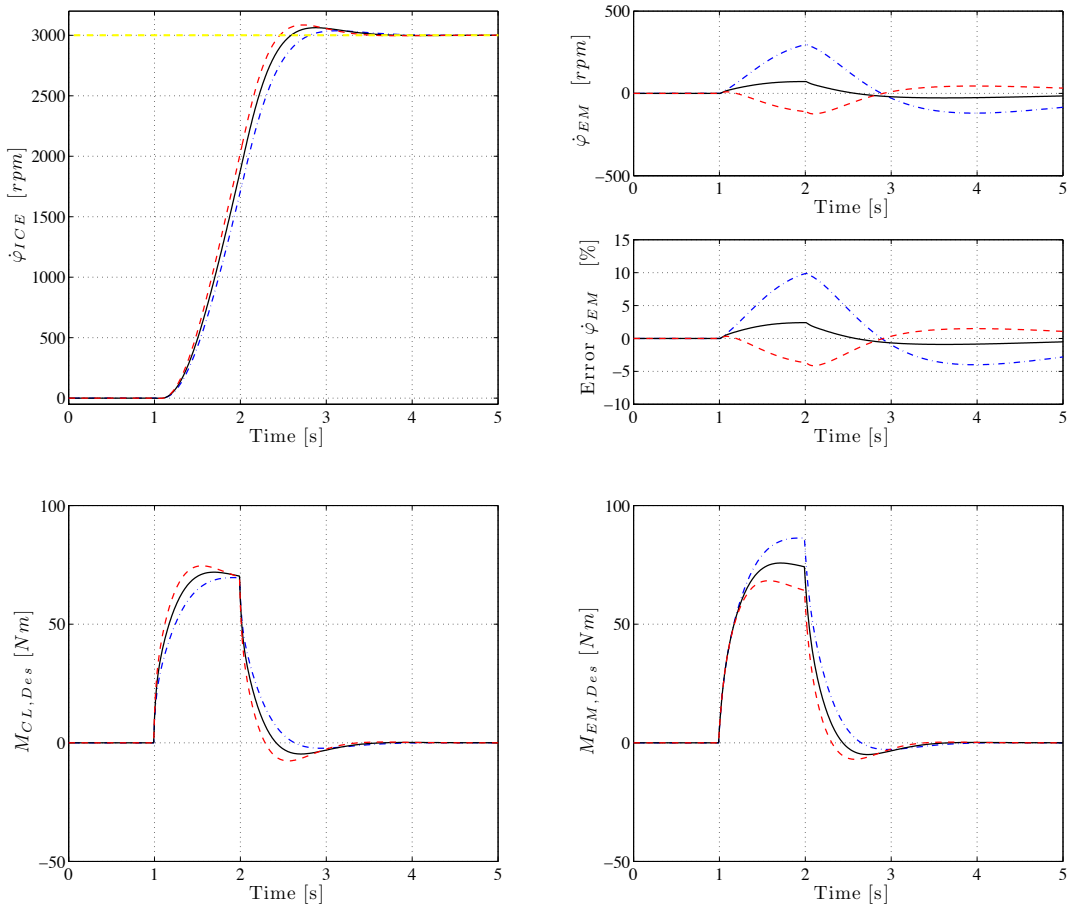

Figure 5.14: Step response with the *PD* SIMO controller for $V = 0.85$ (blue dashdot), $V = 1.0$ (black) and $V = 1.15$ (red dashed), (top, left) ICE angular velocity $\dot{\varphi}_{ICE}$ and desired ICE speed (dash yellow), (top, right) EM angular velocity $\dot{\varphi}_{EM}$ and its deviation in [%], (bottom left) desired clutch torque $M_{Des,CL}$, (bottom right) desired EM torque $M_{Des,EM}$

5.4.2 LTI-Neural Network Controller

In this section the design and the optimization of a NN controller to solve the synchronization problem is presented. Based on the discrete-time model (5.11), a structured closed-loop recurrent neural network is built. It consists of two recurrent networks. The first RNN represents the identified plant in discrete-time. Their weights are set to the plant matrices and will not be updated, this correponds to an already well identified plant behaviour. The second RNN is dedicated to the controller. The weights of this network consists the optimization parameters. These are updated through a specified cost function.

The structure used to optimize the controller is presented in Figure 5.15. Thereby, all

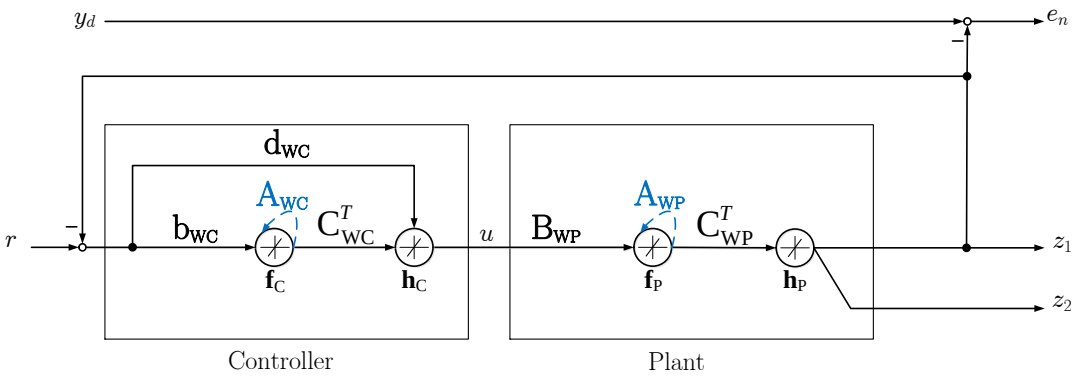

Figure 5.15: Closed-loop structure with recurrent neural network controller

activation functions are set linear. It means that

$$\mathbf{f}_C(\mathbf{x}) = \mathbf{f}_P(\mathbf{x}) = \mathbf{h}_C(\mathbf{x}) = \mathbf{h}_P(\mathbf{x}) = \mathbf{x} \tag{5.22}$$

with $\mathbf{x}$ denotes the input vector. In this case, the linear recurrent neural network describing the plant is given by

$$\begin{aligned} \mathbf{z}_{fP}(k+1) &= \mathbf{A}_{WP} \cdot \mathbf{z}_{fP}(k) + \mathbf{B}_{WP} \cdot \mathbf{u}(k) \\ \mathbf{z}(k) &= \mathbf{C}^T_{WP} \cdot \mathbf{z}_{fP}(k). \end{aligned} \tag{5.23}$$

The matrices $\mathbf{A}_{WP}$, $\mathbf{B}_{WP}$ and $\mathbf{C}^T_{WP}$ consist the plant state-space matrices. This RNN defines a MIMO discrete-time system. It can be used to represent the identified PHEV model given by the expression (5.11). In this case, the matrices $\mathbf{A}_{WP}$, $\mathbf{B}_{WP}$ and $\mathbf{C}^T_{WP}$ are set to the matrices $\mathbf{A}_P$, $\mathbf{B}_P$ and $\mathbf{C}^T_P$, respectively. Moreover, the input and output notation is as follows

$$\mathbf{u} = \begin{bmatrix} M_{CL,Des} \\ M_{EM,Des} \end{bmatrix} \qquad \text{and} \qquad \mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} \dot{\varphi}_{ICE} \\ \dot{\varphi}_{EM} \end{bmatrix} . \tag{5.24}$$

The second RNN denotes the controller. the structure of this network has to be defined. From Figure 5.15, the controller has one input e and two outputs $M_{ICE,Des}$ and $M_{EM,Des}$. It is a SIMO controller, which is defined as follows

$$\begin{aligned} \mathbf{z}_{fC}(k+1) &= \mathbf{A}_{WC} \cdot \mathbf{z}_{fC}(k) + \mathbf{b}_{WC} \cdot (r(k) - \dot{\varphi}_{ICE}(k)) \\ \begin{bmatrix} M_{Des,ICE} \\ M_{Des,EM} \end{bmatrix} &= \mathbf{C}^T_{WC} \cdot \mathbf{z}_{fC}(k) + \mathbf{d}_{WC} \cdot (r(k) - \dot{\varphi}_{ICE}(k)). \end{aligned} \tag{5.25}$$

This results in the controller matrices $\mathbf{A}_{WC} \in \mathbb{R}^{2\times2}$, $\mathbf{b}_{WC} \in \mathbb{R}^{2\times1}$, $\mathbf{C}^T_{WC} \in \mathbb{R}^{2\times2}$ and

$\mathbf{d}_{WC} \in \mathbb{R}^{2\times 1}$.

As discussed in the previous section, the goal is to find a robust controller to cope with the static uncertainty of the separation clutch given by the function (5.5). This is defined through the desired transfer function (5.10). The goal of the NN controller optimization is to minimize the mean square error between the output of the desired model y_d and the output of the model $\dot{\varphi}_{ICE}$. Thereby, the same desired crossover frequency $\omega_c^d = 4\ rad/s$ is used. At the same time the value of the angular velocity $\dot{\varphi}_{EM}$ should not break down. As it is the case for the PD controller, we are assuming that the EM rotates with a specified constant desired value $\dot{\varphi}_{EM}$. The goal is to keep $\dot{\varphi}_{ICE}$ in this level during the synchronization phase. This objective is taken into consideration through the minimization of the mean square error norm between the angular velocity $\dot{\varphi}_{EM,Des}$ and $\dot{\varphi}_{EM}$. Before presenting the optimization problem, the following notation for the controller matrix

$$\mathbf{W}_C = \begin{bmatrix} \mathbf{A}_{WC} & \mathbf{b}_{WC} \\ \mathbf{C}_{WC}^T & \mathbf{d}_{WC} \end{bmatrix} \tag{5.26}$$

which have to be optimized and the constant fixed plant weights $\mathbf{A}_{WP}$, $\mathbf{b}_{WP}$ and $\mathbf{c}_{WP}^T$ given by

$$\mathbf{W}_P = \begin{bmatrix} \mathbf{A}_{WP} & \mathbf{B}_{WP} \\ \mathbf{C}_{WP}^T & \mathbf{0}_{2\times 2} \end{bmatrix}. \tag{5.27}$$

is considered. The representations of the controller and plant matrices (4.36) and (4.37) are introduced to make the understanding easier. The goal is to fit the network output z to a given desired output y_d. To achieve this goal, the following mean square representation

Problem 5.2 :

$$\underset{\mathbf{W}_C,\mathbf{W}_P^0}{\text{minimize}} \quad \underbrace{\frac{1}{N}\sum_{k=0}^{N} \overbrace{\left(y_d(k) - \dot{\varphi}_{ICE}(k,\mathbf{W}_C,\mathbf{W}_P^0)\right)^2}^{e_1^2} + \overbrace{\left(\dot{\varphi}_{EM,Des}(k) - \dot{\varphi}_{EM}(k,\mathbf{W}_C,\mathbf{W}_P^0)\right)^2}^{e_2^2}}_{J(\mathbf{W}_C,\mathbf{W}_P^0)=e^2(k)} \tag{5.28}$$

is used. The above optimization problem consists of minimizing the sum of the square errors $e_1^2(k)$ and $e_2^2(k)$ over the sampling step k. Thereby, the plant matrices denoted by $\mathbf{W}_P^0$ will not be updated. On the other hand, the controller matrix $\mathbf{W}_C$ is updated using the method discussed in Section 4.2.3 which is given as follows

$$\begin{bmatrix} \mathbf{W}_C^{k+1} \\ \mathbf{W}_P^0 \end{bmatrix} = \begin{bmatrix} \mathbf{W}_C^k \\ \mathbf{W}_P^0 \end{bmatrix} - \left(\begin{bmatrix} g(\mathbf{W}_C^k) \\ \mathbf{0}_{10\times 10} \end{bmatrix} \right). \tag{5.29}$$

In this case, the computation of the update function function $g(\mathbf{W}_C^k)$ is performed using the Levenberg-Marquadt algorithm discussed in the Appendix (D). After defining the structure of the recurrent neural network given by Figure 5.15, the network which represents the plant given by the expression (5.23), the network which represents the controller given by the expression (5.25) and the cost function given by the expression (5.28), the control problem can be solved. For this purpose, the whole approach is programmed using the Matlab Neural Network Toolbox.

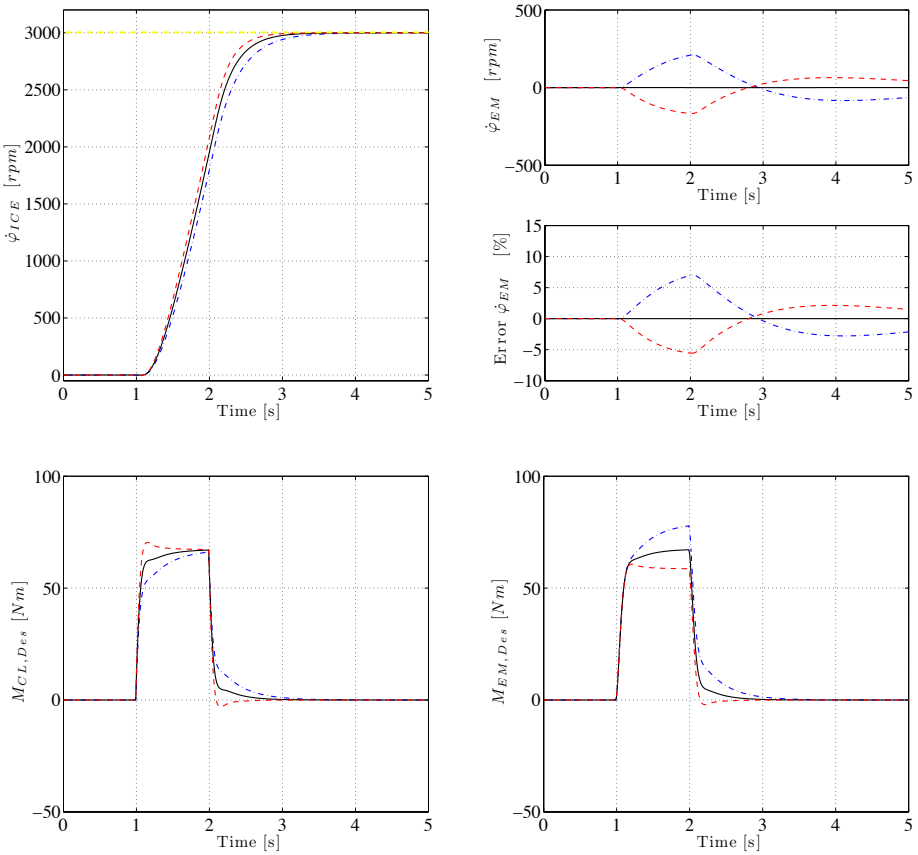

Figure 5.16: Step response with the neural network controller for $V = 0.85$ (blue dash-dot), $V = 1.0$ (black) and $V = 1.15$ (red dashed), (top, left) ICE angular velocity $\dot{\varphi}_{ICE}$ and desired ICE speed (dash yellow), (top, right) EM angular velocity $\dot{\varphi}_{EM}$ and its deviation in [%], (bottom left) desired clutch torque $M_{Des,CL}$, (bottom right) desired EM torque $M_{Des,EM}$

Figure 5.16 shows the obtained performance. The overshoot of the response in terms of the angular velocity $\dot{\varphi}_{ICE}$ for all parameter variations is approximately zero. In this case, the robustness requirement is achieved. Regarding the angular velocity $\dot{\varphi}_{EM}$, the deterioration for the nominal case is zero. For the minimal and maximal cases, it is

about $\mp 210\ rpm$, which corresponds to $\mp 7\%$. It shows a more reduced variation then the performance achieved by the SIMO PD controller.

Remark. *Comparing the performance of both controllers, it is clear that the NN controller outperforms the PD controller. For this reason, the NN controller is used to be implemented on the real plant. Although, an improvement of the NN controller performance in terms of the deterioration of $\dot{\varphi}_{EM}$ can be achieved. This fact is discussed in the next section in details.*

5.4.3 Simulation of the PHEV

In this section, the performance analysis of the obtained NN controller is presented under a more realistic simulation environment. Thereby, three aspects are investigated. First, the interval in which the static gain V varies is augmented. Now it is given by $\mp 20\%$. Another important aspect is considered, namely the time-delay of the system. For this purpose, a variation of the nominal value given by 20 ms is considered. Moreover, up to 250% is assumed. The third aspect is dedicated to the performance of the EM angular velocity $\dot{\varphi}_{EM}$ during a synchronization phase. Additionally to these issues, a more practical assumption is considered. In the previous sections, a reference trajectory for the angular velocity $\dot{\varphi}_{ICE}$ was generated indepently of the velocity $\dot{\varphi}_{EM}$. Unfortunately, this is not actually true in practice. This due to the fact that the goal of the synchronization is to bring $\dot{\varphi}_{EM}$ up to $\dot{\varphi}_{EM}$. In this case, the EM velocity plays the role of the reference trajectory for $\dot{\varphi}_{ICE}$.

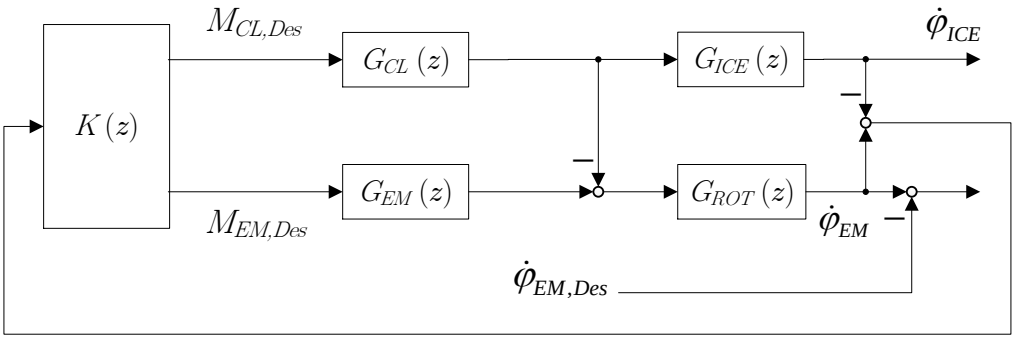

Figure 5.17: Synchronization closed-loop structure with neural network controller $K(z)$

Time Delay

The time delay in the modeling of the PHEV is due to the CAN communication. In Section 5.3, a constant time delay was assumed. It is 20 ms. The behaviour of the real plant shows that this value is not constant. For this reason, it is necessary to test the performance of the controller in case of variations of this parameter. Moreover, we assume a 250% variation. It means that the time-delay varies from 20 to 50 ms.

Figure 5.18 shows the obtained performance for all variations of the time delay. The step response in terms of the angular velocity $\dot{\varphi}_{ICE}$ does not show any overshoot. The impact of the time delay variations is approximately absent. Thereby, this parameter takes the minimal, nominal and maximal values 20 ms, 30 ms and 50 ms, respectively. This means that the performance of the controller is ensured up to 250% variation of the assumed value 20 ms.

Static Gain Variations

Additionally to the time delay variations, the impact of the static gain V is considered. The simulation results are shown in Figure 5.19. The angular velocity $\dot{\varphi}_{ICE}$ reaches the desired value, which is the actual EM velocity $\dot{\varphi}_{EM}$, without any overshoot. This due to the fact that the NN controller is robust in the presence of static gain variations. These are given by $\mp$20% of the nominal value. Additionally, the performance of the EM velocity is considered. Thereby, the deviation for all variations is smaller then 50 rpm. In terms of the relative error, it is less then 2%. This value is acceptable for the performance of the controller.

Actually, the range in which the static gain parameter V varies is not known. For this purpose, we want to find out up to which value of V, the robustness of the neural network controller is guaranteed. To achieve this goal, we assume larger variations of this parameter, given by $\mp$50 %. The simulation results are shown in Figure 5.20. The synchronization of the ICE and EM angular velocities show an overshoot for $V = 1.5$. Moreover, the performance of the EM angular velocity is deteriorated. In this case, the robustness performance is only partially achieved. This means that the robustness performance, in case that the variation of the parameter V is larger than 20 %, is not guaranteed.

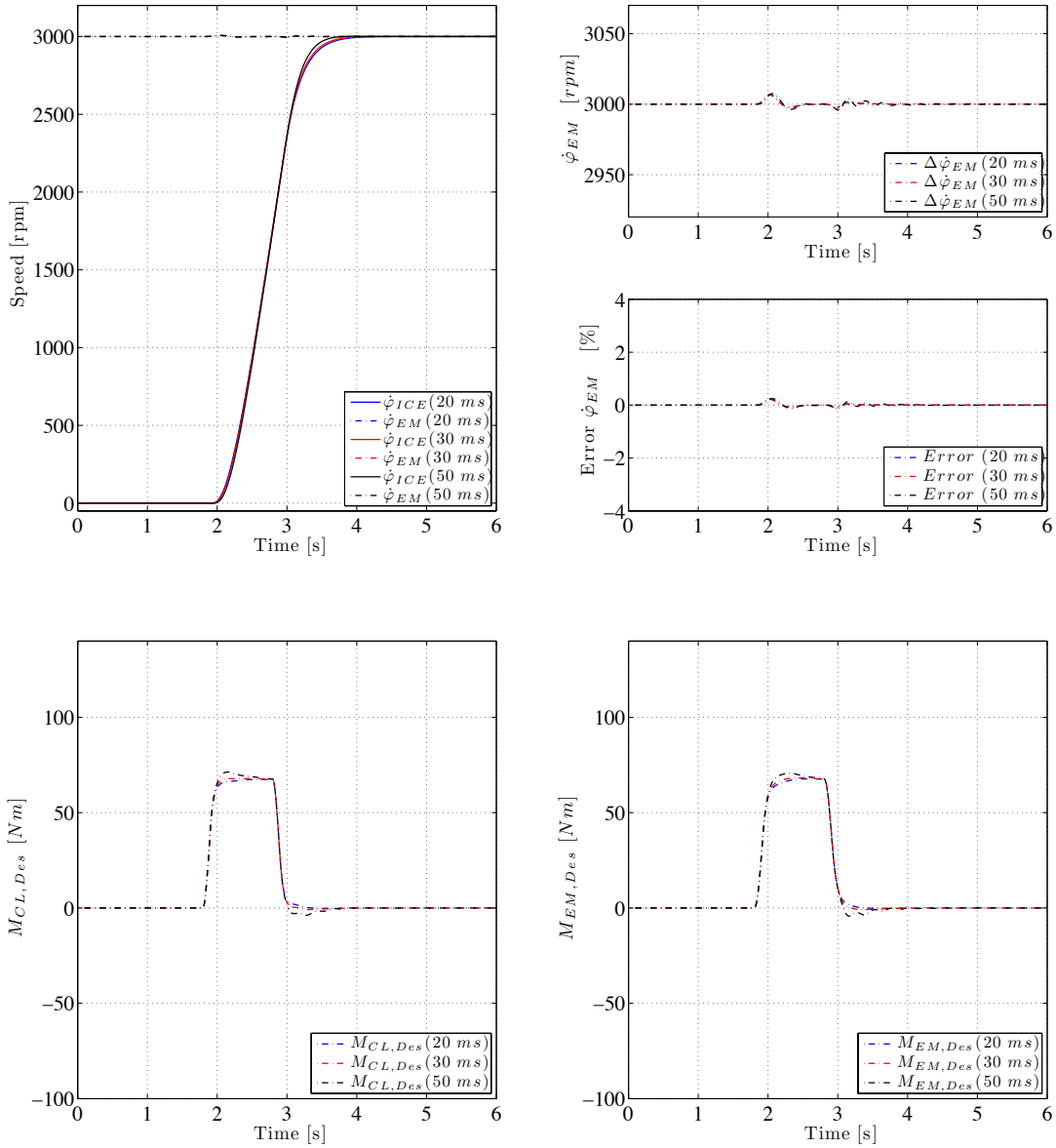

Figure 5.18: Synchronization of ICE speed with EM speed, comparison of the nominal delay 20 ms and increasing delay 30 ms and 50 ms, (top, left) ICE and EM angular velocities $\dot{\varphi}_{ICE}$ and $\dot{\varphi}_{EM}$, (top, right) EM angular velocity $\dot{\varphi}_{EM}$ and its deviation in [%], (bottom left) desired clutch torque $M_{Des,CL}$, (bottom right) desired EM torque $M_{Des,EM}$

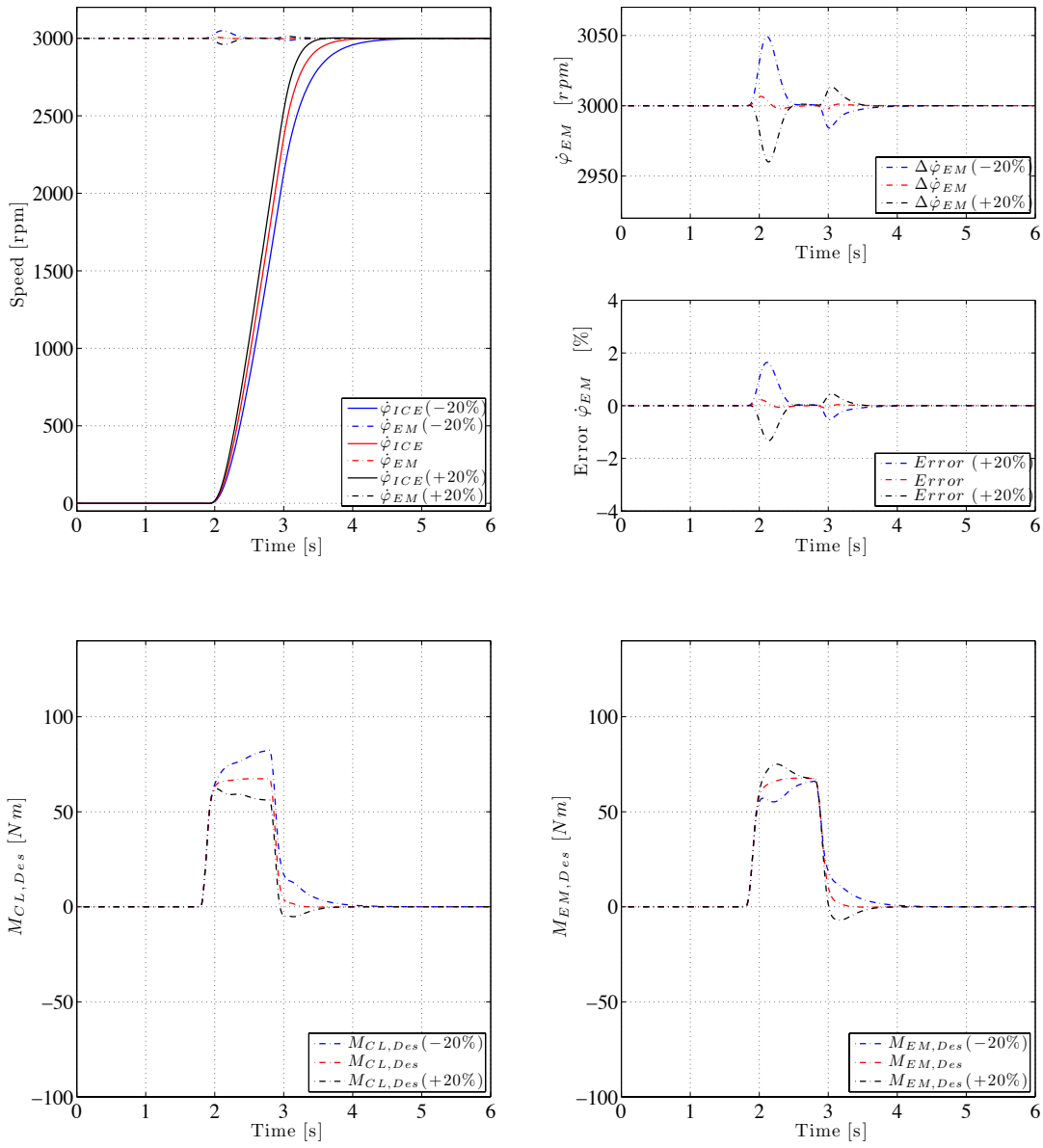

Figure 5.19: Synchronization of ICE speed with EM speed with respect to static gain variations $V = 0.8$, $V = 1.0$ and $V = 1.2$, (top, left) ICE and EM angular velocities $\dot{\varphi}_{ICE}$ and $\dot{\varphi}_{EM}$, (top, right) EM angular velocity $\dot{\varphi}_{EM}$ and its deviation in [%], (bottom left) desired clutch torque $M_{Des,CL}$, (bottom right) desired EM torque $M_{Des,EM}$

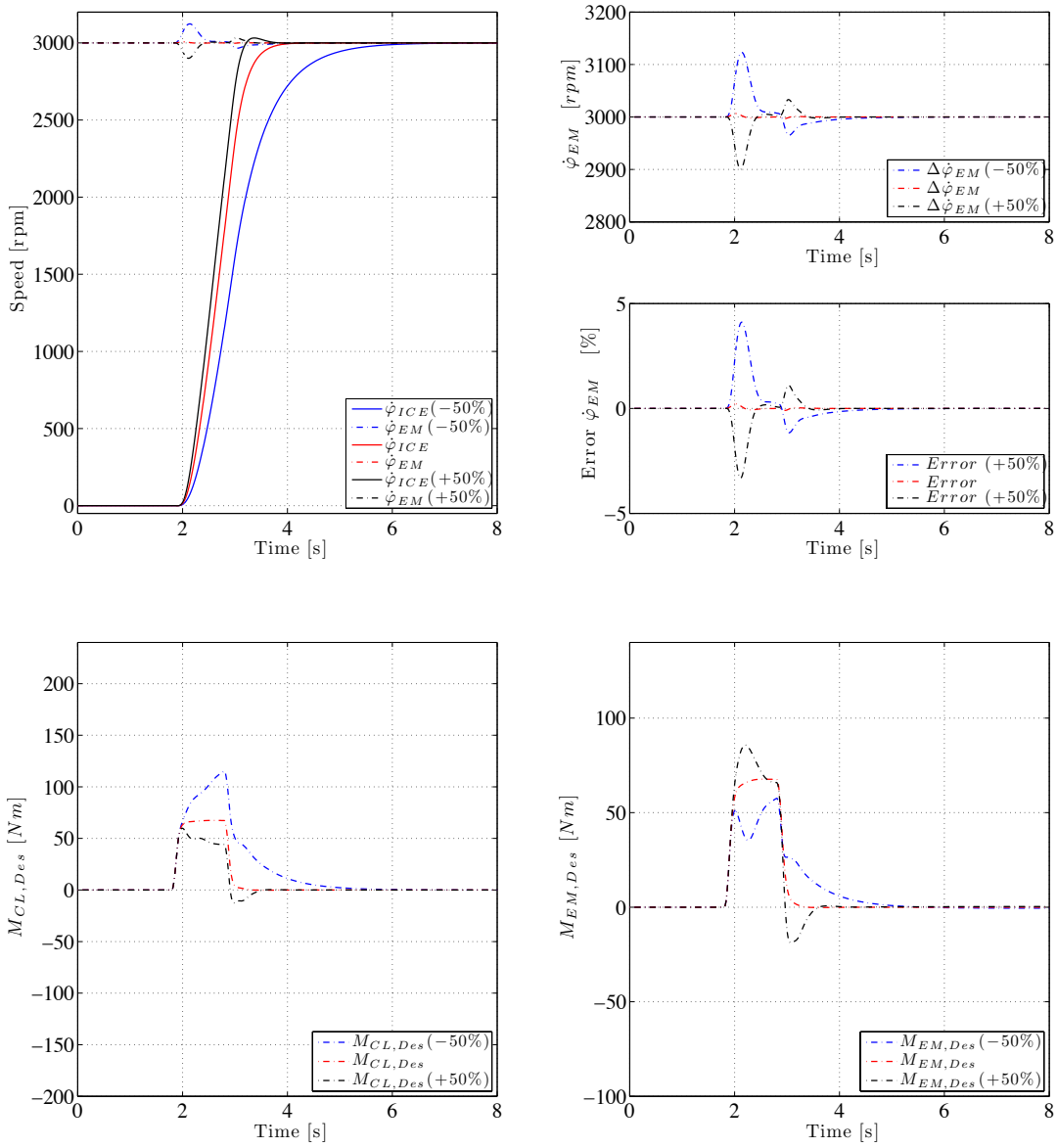

Figure 5.20: Synchronization of ICE speed with EM speed with respect to static gain variations $V = 0.5$, $V = 1.0$ and $V = 1.5$, (top, left) ICE and EM angular velocities $\dot{\varphi}_{ICE}$ and $\dot{\varphi}_{EM}$, (top, right) EM angular velocity $\dot{\varphi}_{EM}$ and its deviation in [%], (bottom left) desired clutch torque $M_{Des,CL}$, (bottom right) desired EM torque $M_{Des,EM}$

5.5 Controller Implementation

After having confirmed in the previous section the capabilities of the robust NN controllers in simulations for gain and time-delay variations, the implementation of this controller on the test PHEV is considered. Moreover, experimental results are presented. For this reason, the Toolchain based on *ETAS* hardware and software is used. It is based on external bypassing the ECU signals.

Before presenting the experimental results, it is important to describe the situation we are considering. It is as follows:

- Initially, the separation clutch is open. The PHEV is driven electrically.
- A synchronization process is activated. This could happen for some of the following reasons:
  - Desired driver torque is higher then the maximal EM torque
  - Desired velocity is higher then the maximal velocity which can be provided by the EM
  - The level of the battery is low. In this case, the ICE has to be started to charge the battery.
- The controller is activated to bring the ICE angular velocity up to the EM angular velocity (synchronization).
- When both velocities are approximately equal, the separation clutch is completely closed.

The main goal which has to be achieved by the controller during the synchronization is to pull up the unfired ICE without any deterioration in the EM velocity. Moreover, the ICE velocity should follow a specified desired trajectory. As a parameter to be used to check this performance, the clutch temperature can be considered. Nevertheless, due to the fact that this parameter is not available on the test vehicle and the estimation for such temperature goes beyond the scope of this work, the ICE temperature is used instead. It serves as an auxiliary variable that helps to evaluate the controller performance. Regarding the ICE velocity $\dot{\varphi}_{ICE}$, the dynamic of the ICE for $\dot{\varphi}_{ICE} < 600\ rpm$ is highly nonlinear so that the controller performance is evaluated for $\dot{\varphi}_{ICE} > 600\ rpm$.

Remark. *The total clutch engagement time (the time it takes to synchronize the angular velocities $\dot{\varphi}_{ICE}$ and $\dot{\varphi}_{EM}$) achieved in Simulation is about two seconds. During the controller implementation, a desired trajectory with a slower increase is used such that the performance of our controller can be compared to existing methods (Dolcini et al. 2005). The desired engagement time $4\,sec$ is used.*

Synchronization Task

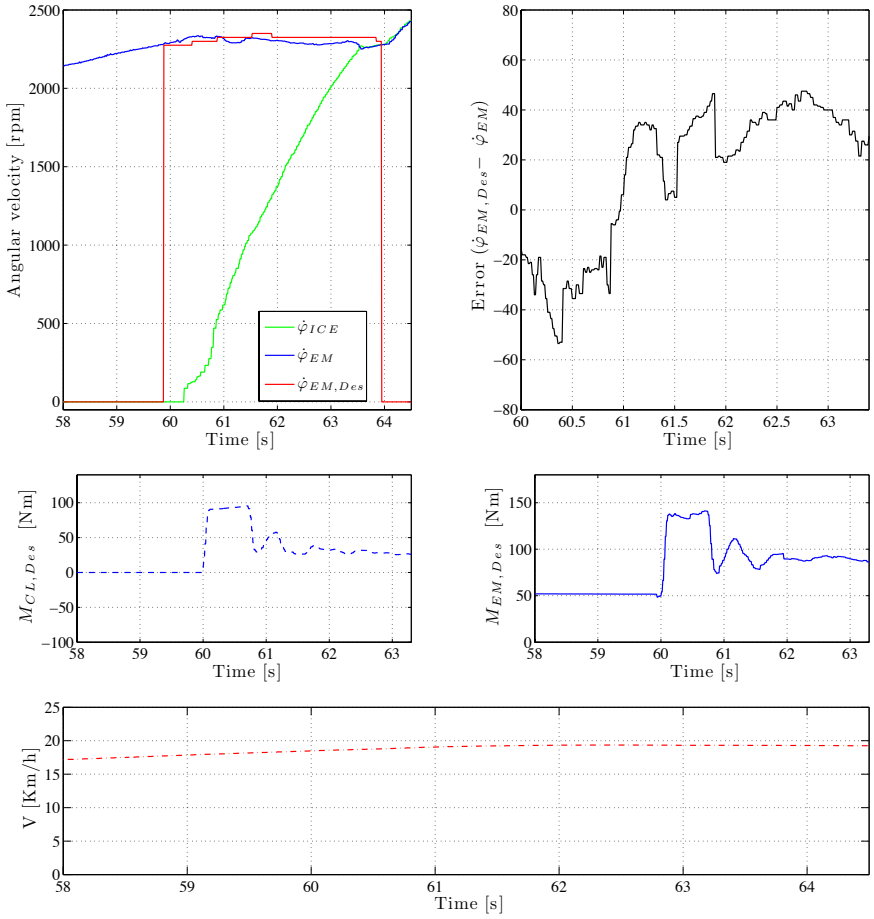

Figure 5.21: Synchronization of the EM and ICE angular velocities (experimental results test vehicle): (top, left) ICE, EM and desired EM angular velocities $\dot{\varphi}_{ICE}$ (green), $\dot{\varphi}_{EM}$ (blue) and $\dot{\varphi}_{EM,Des}$ (red), (top, right) error EM angular velocity $\dot{\varphi}_{EM,Des} - \dot{\varphi}_{EM}$, (centre left) desired clutch torque $M_{CL,Des}$, (centre right) desired EM torque $M_{EM,Des}$, (bottom) vehicle velocity V

Figure 5.21 shows the experimental results of a synchronization task with the implemented neural network controller. During this process, the angular velocity $\dot{\varphi}_{EM}$ follows the desired trajectory $\dot{\varphi}_{EM,Des}$ without any significant deterioration. Thereby, the error $\Delta\dot{\varphi}_{EM}$ is bounded by -60 rpm and $+60$ rpm, which corresponds to -2.3 % and $+2.3$ %. This experiment was performed at the vehicle velocity $V \approx 18$ Km/h and the motor temperature $T_{Eng} = 80°\text{C}$. Note here that the desired clutch torque $M_{CL,Des}$ has to be provided by the EM additionally to the driver desired torque. This is shown in the figure above (centre right) by the increasing of the EM torque by $M_{CL,Des}$ at 60 s.

Synchronization Task: Engine temperature $T_{Eng} = 67.5°$

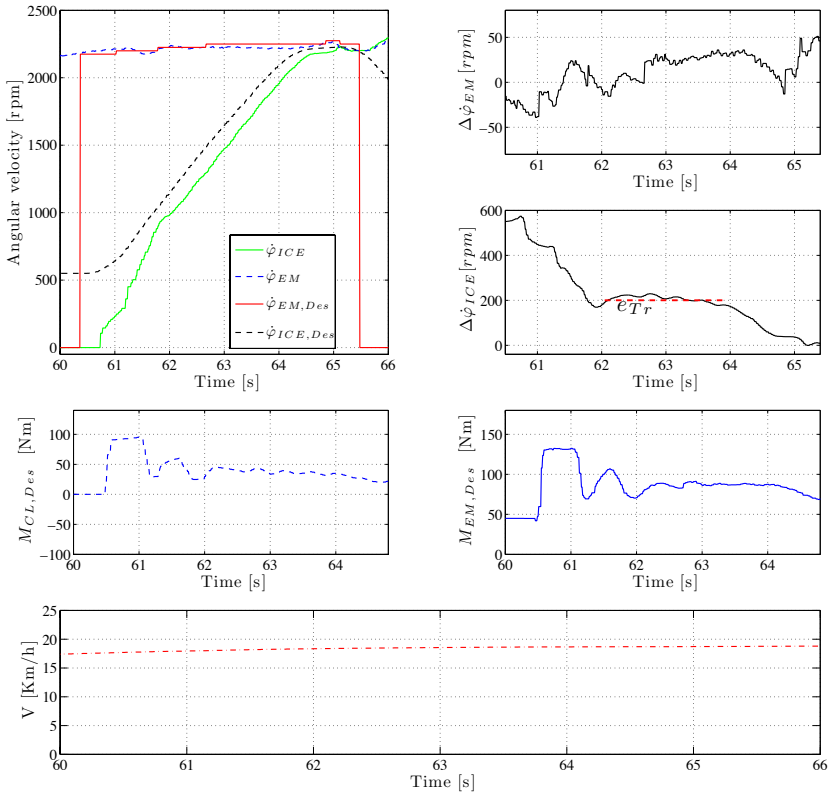

Figure 5.22: Synchronization of the EM and ICE speeds (experimental results test vehicle): (top, left) ICE, EM and desired ICE and EM angular velocities $\dot{\varphi}_{ICE}$ (green), $\dot{\varphi}_{EM}$ (dash blue), $\dot{\varphi}_{EM,Des}$ (red) and $\dot{\varphi}_{ICE,Des}$ (dash black), (top, right) variations of the EM and ICE angular velocities $\Delta\dot{\varphi}_{EM}$ and $\Delta\dot{\varphi}_{ICE}$, (centre left) desired clutch torque $M_{CL,Des}$, (centre right) desired EM torque $M_{EM,Des}$, (bottom) vehicle velocity V

Figure 5.22 shows the experimental results of a synchronization task with a reference trajectory $\dot{\varphi}_{ICE,Des}$ for the angular velocity $\dot{\varphi}_{ICE}$. Thereby, the velocity $\dot{\varphi}_{EM}$ follows its reference $\dot{\varphi}_{EM,Des}$. The deviation $\Delta\dot{\varphi}_{EM}$ is approximately between -39 rpm and 51 rpm, which corresponds respectively to -1.7 % and 2.2%. To analyse the performance of the controller, we used the tracking error e_{Tr} shown for the ICE speed variation $\Delta\dot{\varphi}_{ICE}$, which is the smallest error between $\dot{\varphi}_{ICE,Des}$ and $\dot{\varphi}_{ICE}$ as soon as both velocities run parallel. This is approximately equal to 200 rpm for the parallel section. This experiment was performed at the motor temperature $T_{Eng} = 67.5°$C.

The rest of the experimental results is given in Appendix A.

5.6 Summary

In this section, the design and optimization of the controllers developed in Section 3 and 4 for the synchronization task in a PHEV have been considered. First of all, a MIMO model of the plant has been identified using a rapid control prototyping system. This model consists of four transfer functions describing respectively the dynamics of the ICE, EM, clutch and powertrain. Validation results have shown that the identified linear MIMO system provides a very good fit of the measured data. Based on this model, two controllers have been considered, namely a SIMO PD controller and a second order state-space neural network. By means of robustness analysis, the NN controller outperformed the PD controller.

Table 5.1: PHEV Controller Performance

T_{Eng} $[°C]$	67.5	69.6	74.15	77.8	79.6	83.1
$e_{Tr}[rpm]$	200	123	110	32	103	85
$\Delta\dot{\varphi}_{EM}[rpm]$	$[-39\ 51]$	$[-45\ 51]$	$[-30\ 48]$	$[-40\ 56]$	$[-32\ 42]$	$[-36\ 66]$
$\Delta\dot{\varphi}_{EM}[\%]$	$[-1.7\ 2.2]$	$[-2.1\ 2.3]$	$[-1.53\ 2.2]$	$[-1.9\ 2.4]$	$[-1.5\ 1.9]$	$[-1.6\ 2.9]$
sae_N	227.5	224.4	169.8	135.3	163.6	143.3

Additionally to the tracking error e_{Tr} and the angular velocity variations $\Delta\dot{\varphi}_{ICE}$ and $\Delta\dot{\varphi}_{EM}$, the following error

$$sae_N = \sum_{k=0}^{N} \frac{|\dot{\varphi}_{ICE,Des} - \dot{\varphi}_{ICE}|}{N} = \sum_{k=0}^{N} \frac{|e(k)|}{N} \tag{5.30}$$

is used to evaluate the performance at each temperature T_{Eng}, see (Unbehauen 2008, p. 191). The parameter N consists the number of samples needed for a synchronization process. The experimental results of the NN controller are summarized in Table 5.1. In all cases, the variations of the angular velocity $\Delta\dot{\varphi}_{EM}$ is less than 3 %. Thereby, the tracking error e_{Tr} is varying depending on the temperature T_{Eng}, which is an indicator that the system gain is also changing. The NN controller provides a robust performance.

6 Conclusion and Outlook

The main goal of this thesis consists in exploring new methods and techniques to design robust controllers. The work focus was put on their practicability in terms of the implementation in a real environment. Thereby, the controller should provide a specified performance in the whole parameter range of interest. Three controllers have been used to achieve this task, namely fractional as well as integer order PID controllers and then neural network structures.

Due to the fact that fractional operators are not well known in the control community, an introduction to this field has been given in Appendix B. Magnitude and phase Bode plot of fractional derivatives and integrals are discussed. The advantage of this operator is that the slope of the magnitude and phase are not restricted respectively to a whole multiple of 20 dB and 90° any more. This fact is very important in the design of robust controllers as it opens new possibilities in tuning the controller for uncertain plants. This later class of plant consists of a nominal transfer function which is constant and a static gain parameter, which varies in a specified range. In this context, an uncertain model of the electronic throttle is introduced to show the impact of parameter variations on the frequency response given by the Bode plot. It has been shown that this consists of a shifting of the magnitude plot; the phase plot is not affected by theses variations.

In Section 2, the optimization of the robust fractional order controllers $(PI^{\alpha}D^{\beta})$ and $(PID)^n$ has been treated in detail. Thereby, the robustness requirements are given in terms of a desired crossover frequency and a desired phase margin for all parameter variations of systems with static gain variations. It has been shown that we can formulate these requirements as a desired fractional order integrator for the open-loop transfer function. The case in which a maximal overshoot is determined instead of phase margin has been also considered. For this reason, a method has been given to represent the maximal overshoot for the closed-loop step response as a desired phase margin. Two approaches have been developed to optimize the fractional controllers parameters, which are based on shaping the open-loop in the frequency domain response for the $PI^{\alpha}D^{\beta}$ controller or the closed-loop response for the $(PID)^n$. To solve the related optimization problem, an algorithm based on the recently developed nonsmooth techniques and the Steepest-Descent method was presented.

The fractional order optimization problems (2.52) and (2.58) in Section 2.3.1 are nonsmooth and nonconvex, which motivates us to design a novel Fractional Order PID Controller (FOPID) Matlab Toolbox. It aims to enable the user to tune the controller parameters further in case that the obtained parameter does not satisfy the robustness requirements. In all the examples which have been considered, it was not necessary to use this feature. The obtained controllers already satisfy the imposed specifications. Moreover, the comparison to other existing methods in the literature has shown that our FOPID provides the best performance.

In case that the robustness specifications to be achieved are given in terms of an overshoot

free step response, classical integrator can be used to formulate this requirement. In Section 3, the optimization of robust controllers that achieve this goal has been discussed. Thereby, as the mostly used controllers type $PIDs$ are considered. Due to its capability to represent constraints on systems, Model Predictive Control (MPC) has been applied to formulate the related optimization problem. Open-loop as well as closed-loop MPC problems are treated. Moreover, we propose a method to formulate these problems in terms of Linear Matrix Inequalities (LMIs). The advantage of this formulation is that LMIs correspond to convex problems, which can be solved efficiently using existing LMI solvers. This has been shown through the computation of an optimal input trajectory for the electronic throttle. Thereby, constraints on the input and output were imposed. As we are interested in the optimization of robust controllers, the formulation of the closed-loop problem is discussed. In turns out that the related optimization problem is nonconvex given in terms of a Bilinear Matrix Inequality (BMI) which is a nonconvex problem, and therefore hard to solve. For this reason, we proposed a method to transform this problem into a set of LMIs based on the Schur complement. The effectiveness of our proposed method has been shown through the optimization of a robust PD controller for the electronic throttle. Simulation results are given that showed the achieved performance.

Although, PID controllers are able to provide a satisfactory performance in many applications, it is preferable to have other alternatives in case that they fail. For this reason, the design of a more sophisticated and modern methods has been explored in Section 4, by applying Neural Network to the control problem. Firstly, the classification of NNs into feedforward and recurrent network have been given. The capability of nonlinear networks to approximate a given nonlinear static function is demonstrate with a numerical example. Regarding dynamic systems, Recurrent Neural Networks (RNNs) provides a more appropriate network structure to be used. In this context, we proposed a novel RNN representation, which represents the closed-loop system. Thereby, we proposed to use this structure to formulate the closed-loop problem in which whether the controller is known and the plant has to be identified or a controller has to be optimized for a given known plant. In both cases, we introduced an algorithm to optimize the system parameters given by the weights of the related RNN.To demonstrate the effectiveness of our approach, the optimization of robust controller for the uncertain model of the electronic throttle has been presented. Thereby, the same robustness as in Section 3 are considered. Simulation results have shown that the RNN controller always provides the imposed robustness specifications.

Both proposed approaches in Section 3 and 4 are suitable to optimize robust controllers for uncertain systems. To compare both methods, the design of a Single-Input Multiple-Output (SIMO) controller has been considered in Section 5. The system under consideration is Multi-Input Multi-Output model of the synchronisation of the Internal Combustion Engine (ICE) and Electric Machine (EM) velocities in a Parallel Hybrid Electrical Vehicle (PHEV). This consists a very interesting plant to be explored as it is a very active field of research. With the help of a Rapid Control Prototyping (RCP) system, we identified

a MIMO model that describes the dynamics of the synchronisation task from the torque of the EM and the clutch to the angular velocity of the ICE and the EM. Additionally to imperfection of the identification, the main source of uncertainty corresponds to the separation clutch. Thereby, we proposed to represent this uncertainty as variations in a multiplicative static gain of the clutch transfer function. Using our method, the uncertainties related to the transmitted torque by the clutch can be handled. This fact has been confirmed for the optimized RNN and LMI based robust controllers. Simulation results have shown that the RNN controller provides the imposed requirements given in terms of an overshoot free step response of the ICE angular velocity for all variations of the clutch model static gain. Morever, it has been shown that this RNN outperformed the LMI based controller. For this reason, the implementation of RNN in a real PHEV have been investigated. To round off this work, we applied the RNN controller to solve synchronisation task. Experimental results are presented to shown the achieved robust performance.

Outlook

Further research activities are proposed here:

- Fractional Order Controllers:
  - In this work, the optimization of the fractional controllers $PI^{\alpha}D^{\beta}$ and $(PID)^{n}$ for Single-Input Single-Output (SISO) uncertain systems has been treated. A future work would be to extend the proposed method to Multi-Input Multi-Output (MIMO) systems. It will be also very interesting to explore the application to nonlinear systems. Additionally, another point which can be explored is the optimization of fractional controller to handle not only static gain variations but general parameter variations.
- Optimization of PID controllers using MPC and LMIs:
  - Regarding the optimization of PID controllers, it has been shown that MPC can be combined with LMI to produce an efficient method to optimize the parameters of these controllers. The stability of the controller can be checked afterwards using standard classical methods. A future work would be to incorporate a priori stability conditions into the set of LMIs. This can be achieved using for example Lyapunov stability condition. Another point which can be explored is the application of our method to nonlinear systems. As we are using the MPC formulation, this can be achieved in a straightforward manner. Another important point to be explored is the application of our method to optimize adaptive PID controllers for slow parameter varying systems.
- Artificial Neural Network Controller:
  - We have shown in this work that neural network can be applied to design robust controllers achieving a specified performance. We have also demonstrate that

such networks, when well structured, can be applied to solve practical problems. The closed-loop structure presented here can be extended to nonlinear systems as well as nonlinear controllers. Moreover, an interesting issue will be to explore adaptive RNNs and real-time tuning of the neural network weights.

- Experimental implementation on a Parallel Hybrid Electrical Vehicle
  - Regarding the synchronization task of the test PHEV, it was shown that the RNN neural network provides a better performance than the LMI based controller. Moreover, experimental results were very promising as well as many ideas have been raised related to the synchronization task in PHEVs. These are summarized here:
    1. To design a robust controller, it is firstly necessary to specify the range in which the uncertain parameter V varies. Otherwise, the performance can not be guaranteed. For this reason, the estimating of the real transmitted clutch torque can be explored. To achieve this goal, one can use linear or nonlinear observer techniques.
    2. The model used to identify the ICE was linear. An important point would be to explore nonlinear identification methods to describe the nonlinear dynamics of the ICE.
    3. Due to the influence of the temperature on the clutch dynamics, another important point which should be explored is to estimate its value by means of a temperature observer.
    4. Contrary to robust controllers, parameter varying controllers can be adapted in real-time to achieve a more specific performance in terms of some physical parameters as temperature, transmitted clutch torque, etc.

7 References

Abbas, H. and **H. Werner**. 2008. Polytopic Quasi-LPV Model Based on Neural State-Space Models and Application to Air Charge Control of a SI Engine. *Proceeding of the 17th IFAC World Congress, Seoul, Korea.* pp. 6466–6471.

Abbas, H., **M. Ali** and **H. Werner**. 2010. Linear Recurrent Neural Network for Open- and Closed-loop Consistent Identification of LPV Models. *Proceeding of the 49th IEEE Conference on Decision and Control, Atlanta, USA*. pp. 6851–6856.

Abel, D. and **A. Bollig**. 2006. *Rapid Control Prototyping.* Springer.

Abi Zeid Daou, R., **X. Moreau** and **C. Francis**. 2011. Control of Hydro-Electromechanical System Using the Generalized PID and the Crone Controllers. *Proceeding of the 18th IFAC World Congress, Milan, Italy.* pp. 15037–15042.

Ahmed, M., **N. Lachhab** and **F. Svaricek**. 2012. Non-Model Based Adaptive Control of Electro-Hydraulic Servo Systems using Prefilter Inversion. *Proceeding of the 9th International Multi-Conference on Systems, Signals and Devices, Chemnitz, Germany.* pp. 1–6.

Akpan, V. A. and **G. Hassapis**. 2009. Adaptive Predictive Control using Recurrent Neural Network Identification. *Proceeding of 17th Mediterranean Conference on Control and Automation, Thessaloniki, Greece.* pp. 61–66.

Alt, B., **J. Blath**, **F. Svaricek** and **M. Schultalbers**. 2010. Self-Tuning Control Design Strategy for an Electronic Throttle with Experimental Robustness Analysis. *Proceeding of the American Control Conference, Baltimore, Maryland, USA*. pp. 6127–6132.

Alt, B., **F. Antritter** and **F. Svaricek**. 2012. Flatness based Control for Electric and Hybrid Electric Vehicle Drivetrains. *Proceeding of the 20th Mediterranean Conference on Control & Automation, Barcelona, Spain.* pp. 915–920.

Alt, B., **F. Antritter**, **F. Svaricek**, **J. P. Blath** and **M. Schultalbers**. 2010. Improved Performance for the Synchronization of the Angular Velocity in Hybrid Electric Vehicles using a Feedforward Strategy. *Proceeding of the 6th IFAC Symposium on Advances in automotive control, Munich, Germany.* pp. 530–535.

Amari, R., **M. Alamir**, **P. Tona** et al. 2008. Unified MPC Strategy for Idle-Speed Control, Vehicle Start-Up and Gearing Applied to an Automated Manual Transmission. *Proceeding of the 17th IFAC World Congress, Seoul, South Korea.*

Amari, R., **P. Tona** and **M. Alamir**. 2009. Experimental evaluation of a hybrid MPC strategy for vehicle start-up with an Automated Manual Transmission. *Proceeding of the European Control Conference, Budapest, Hungary.*

Apkarian, P. and **D. Noll**. 2006. Nonsmooth H_∞ Synthesis. *IEEE Transactions on Automatic Control* 51(1). pp. 71–86.

Apkarian, P., **J.-M. Biannic** and **P. Gahinet**. 1995. Self-Scheduled H_∞-Infinity Control of Missile via Linear Matrix Inequalities. *Journal of Guidance, Control, and Dynamics* 18(3). pp. 532–538.

Apkarian, P. and **P. Gahinet**. 1995. A Convex Characterization of Gain-Scheduled H_∞ Controllers. *IEEE Transactions on Automatic Control* 40(5). pp. 853–864.

Apkarian, P., **P. Gahinet** and **G. Becker**. 1995. Self-Scheduled H_∞ Control of Linear Parameter-Varying Systems: A Design Example. *Automatica* 31(9). pp. 1251–1261.

Apkarian, P., **P. Gahinet** and **C. Buhr**. 2014. Multi-model, Multi-objective Tuning of Fixed-Structure Controllers. *Proceeding of the European Control Conference, Strasbourg*. p. 856–861.

Arsie, I., **S. Di Iorio**, **C. Pianese**, **G. Rizzo** and **M. Sorrentino**. 2008. Recurrent Neural Networks for Air-Fuel Ratio Estimation and Control in Spark-Ignited Engines. *Proceeding of the 17th IFAC World Congress, Seoul, Korea*. pp. 6–11.

Åström, K. and **T. Hägglund**. 2001. The Future of PID Control. *Control Engineering Practice* 9(11). pp. 1163–1175.

Balas, G., **R. Chiang**, **A. Packard** and **M. Safonov**. 2011. Robust Control Toolbox User's Guide.

Banos, A., **J. Cervera**, **P. Lanusse**, **J. Sabatier** et al. 2008. Bode optimal loop shaping with CRONE compensators. *Proceeding of the 14th IEEE Mediterranean Electrotechnical Conference, Beirut, Lebanon*.

Beck, R., **F. Richert**, **A. Bollig**, **D. Abel**, **S. Saenger**, **K. Neil**, **T. Scholt** and **K.-E. Noreikat**. 2005. Model Predictive Control of a Parallel Hybrid Vehicle Drivetrain. *Proceeding of the 44th IEEE Conference on Decision and Control, European Control Conference, Seville, Spain*. pp. 2670–2675.

Beck, R., **S. Saenger-Zetina**, **A. Bollig** and **K. Neiß**. 2007. Robuste Modellprädiktive Regelung für Hybridgetriebe. *at Automatisierungstechnik*.

Bemporad, A. 2006. Model predictive control design: New trends and tools. *Proceedings of the 45th IEEE Conference on Decision and Control, San Diego, USA*. pp. 6678–6683.

Bemporad, A., **M. Morari**, **V. Dua** and **E. N. Pistikopoulos**. 2002. The explicit linear quadratic regulator for constrained systems. *Automatica* 38(1). pp. 3–20.

Bode, H. W. 1945. *Network analysis and feedback amplifier design.* D. Van Nostrand New York.

Borgeest, K. 2010. *Elektronik in der Fahrzeugtechnik.* Springer.

Boyali, A., **M. Demirci**, **T. Acarman**, **L. Guvenc**, **O. Tur**, **H. Ucarol**, **B. Kiray** and **E. Ozatay**. 2006. Modeling and Control of a Four Wheel Drive Parallel Hybrid Electric Vehicle. *Proceeding of the IEEE International Symposium on Intelligent Control, Computer Aided Control System Design, International Conference on Control Applications, Munich, Germany.* pp. 155–162.

Boyd, S. and **L. Vandenberghe**. 2004. *Convex Optimization.* Cambridge university press.

Boyd, S., **L. El Ghaoui**, **E. Feron** and **V. Balakrishnan**. 1994. Linear Matrix Inequalities in System and Control Theory.

Boyd, S. P., **C. H. Barratt**, **S. P. Boyd** and **S. P. Boyd**. 1991. *Linear Controller Design: Limits of Performance.* Prentice Hall Englewood Cliffs, NJ.

Chen, Y., **C. Hu** and **K. L. Moore**. 2003. Relay Feedback Tuning of Robust PID Controllers With Iso-Damping Property. *Proceeding of the 42nd IEEE Conference on Decision and Control, Hawaii, USA.* pp. 2180–2185.

Chilali, M., **P. Gahinet** and **P. Apkarian**. 1999. Robust Pole Placement in LMI Regions. *IEEE Transactions on Automatic Control* 44(12). pp. 2257–2270.

Clarke, F. H. 1990. *Optimization and Nonsmooth Analysis.* SIAM.

de Oliveira, M. C., **J. Bernussou** and **J. C. Geromel**. 1999. A new discrete-time robust stability condition. *Systems & control letters* 37(4). pp. 261–265.

Deguchi, Y., **H. Itoyama** and **Y. Kitajima**. 2000. Hybrid drive system for vehicle with clutch control. US Patent 6,083,139.

Dias, F. M. and **A. M. Mota**. 2001. Comparison between Different Control Strategies using Neural Networks. *Proceeding of the 9th Mediterranean Conference on Control and Automation, Dubrovnik, Croatia.*

Dolcini, P., **H. Béchart** and **C. Canudas de Wit**. 2005. Observer-based optimal control of dry clutch engagement. *Proceeding of the 44th IEEE Conference on Decision and Control, European Control Conference, Seville, Spain.* pp. 440–445.

Ehsani, M., **Y. Gao** and **A. Emadi**. 2009. *Modern Electric, Hybrid Electric, and Fuel Cell Vehicles: Fundamentals, Theory, and Design.* CRC press.

Eisenwerth, K., **M. Schnitzer**, **F. Iacona** and **M. Klymenko**. 2013. Adaption des Kupplungsmomentes einer Trennkupplung eines Parallelhybrid-Antriebsstrangs eines Fahrzeugs. DE Patent App. DE201,110,089,676.

ETAS 2008a. *INCA V6.2 Schnelleinstieg.*

ETAS 2008b. *INCA V6.2 Tutorial.*

ETAS 2009a. *ES 910.3 Rapid Prototyping Module User's Guide.*

ETAS 2009b. *INTECRIO V4.1 User's Guide.*

Farag, A. and **H. Werner**. 2006. Fixed-Structure μ-Synthesis An Evolutionary Approach. *Proceedings of the American control conference, Minneapolis, MN, USA*.

Fliess, M., **C. Join**, **S. Riachy** et al. 2011. Revisiting some practical issues in the implementation of model-free control. *Proceeding of the 18th IFAC World Congress, Milano, Italy.*

Fliess, M., **C. Join** et al. 2008. Intelligent PID Controllers. *Proceeding of the 16th Mediterrean Conference on Control and Automation, Ajaccio, Corsica, France.*

Fueger, T., **N. Lachhab** and **F. Svaricek**. 2013. Parameter Reduction for Disturbance Attenuation with a Discrete-Time LPV Controller. *Proceeding of the 5th IFAC Symposium on System Structure and Control, Grenoble, France*. pp. 791–796.

Fukushima, H. and **R. R. Bitmead**. 2003. Robust constrained model predictive control using closed-loop prediction. *Proceedings of the American Control Conference, Denver, Colorado*. IEEE. 2511–2516.

Gahinet, P. and **P. Apkarian**. 1994. A Linear Matrix Inequality Approach to H_∞ Control. *International Journal of Robust and Nonlinear Control* 4(4). pp. 421–448.

Gahinet, P. and **P. Apkarian**. 2011. Structured H_∞ Synthesis in MATLAB. *Proceeding of the 18th IFAC World Congress, Milan, Italy.*

Gahinet, P., **A. Nemirovski**, **A. Laub** and **M. Chilali**. 1995. LMI Control Toolbox User's Guide. 1995. *The Mathworks, Massachusetts.*

Gasper, R., **F. Hesseler** and **D. Abel**. 2011. Control of a Parallel Hybrid Drivetrain: A Flatness Based Approach. *Proceeding of the 18th IFAC World Congress, Milano, Italy.* pp. 2943–2948.

Gasper, R., **B. Ralf**, **D. Peter** and **A. Dirk**. 2009. Feedforward Control of a Parallel Hybrid Launch Clutch. *Proceedings of the European Control Conference 2009, Budapest, Hungary.* pp. 4264–4270.

Gavin, H. 2011. The levenberg-marquardt method for nonlinear least squares curve-fitting problems. *Department of Civil and Environmental Engineering, Duke University* 1–15.

Gil, P., **J. Henriques**, **A. Dourado** and **H. Duarte-Ramos**. 2006a. On State-Space Neural Networks for Systems Identification: Stability and Complexity. *Proceeding of the IEEE Conference on Cybernetics and Intelligent Systems, Singapore*. pp. 1–5.

Gil, P., **J. Henriques**, **A. Dourado** and **H. Duarte-Ramos**. 2006b. On State-Space Neural Networks for Systems Identification: Stability and Complexity. *Proceeding of the IEEE Conference on Cybernetics and Intelligent Systems, Bangkok, Thailand*. pp. 1–5.

Hofmann, P. 2014. Hybridfahrzeuge. *Hybridfahrzeuge:, ISBN 978-3-7091-1780-4. Springer-Verlag Vienna, 2014*.

Huang, C. Y., **T. C. Chen** and **C. L. Huang**. 1999. Robust Control of Induction Motor with a Neural-Network Load Torque Estimator and a Neural-Network Identification. *IEEE Transactions on Industrial Electronics,* 46(5). pp. 990–998.

Hunt, K. and **D. Sbarbaro**. 1991. Neural networks for nonlinear internal model control. *Proceeding of the IEE Control Theory and Applications*. IET. pp. 431–438.

Isermann, R. and **N. Müller**. 2001. Modelling and Adaptive Control of Combustion Engines with Fast Neural Networks. *Proceeding of the European Symposium on Intelligent Technologies, Puerto de la Cruz, Spain*.

Jarczyk, J., **B. Alt**, **J. Blath**, **F. Svaricek** and **M. Schultalbers**. 2009. Decoupling Control for the Speed Synchronization Task in the Powertrain of a Hybrid Electric Vehicle. *Proceeding of the IEEE International Conference on Control and Automation, Christchurch, New Zealand*. 2154–2159.

Kapernick, B. and **K. Graichen**. 2014. The Gradient Based Nonlinear Model Predictive Control Software GRAMPC. *Proceeding of the European Control Conference, Strasbourg*. p. 1170–1175.

Karmarkar, N. 1984. A NEW POLYNOMIAL-TIME ALGORITHM FOR LINEAR PROGRAMMING. *Proceeding of the sixteenth annual ACM symposium on Theory of computing, New York, USA*. ACM. pp. 302–311.

Khatibi, H., **A. Karimi** and **R. Longchamp**. 2008. Fixed-Order Controller Design for Polytopic Systems using LMIs. *IEEE Transactions on Automatic Control* 53(1). pp. 428–434.

Kothare, M. V., **V. Balakrishnan** and **M. Morari**. 1996. Robust constrained model predictive control using linear matrix inequalities. *Automatica* 32(10). 1361–1379.

Kuwata, Y., **T. Schouwenaars**, **A. Richards** and **J. How**. 2005. Robust constrained receding horizon control for trajectory planning. *Proceedings of the AIAA Guidance, Navigation, and Control Conference*. Citeseer.

Labit, Y., **D. Peaucelle** and **D. Henrion**. 2002. SEDUMI INTERFACE 1.02: a tool for solving lmi problems with SEDUMI. *Proceeding of the IEEE International Symposium on Computer Aided Control System Design, Glasgow, Scotland, UK*. pp. 272–277.

Lachhab, N., **H. Abbas** and **H. Werner**. 2008. A Neural-Network Based Technique for Modelling and LPV Control of an Arm-Driven Inverted Pendulum. *Proceeding of the 47th IEEE Conference on Decision and Control, Cancun, Mexico*. pp. 3860–3865.

Lachhab, N., **F. Svaricek**, **F. Wobbe** and **H. Rabba**. 2013. Fractional Order PID Controller (FOPID)-Toolbox. *Proceedings of the European Control Conference, Zurich, Switzerland*. pp. 3694–3699.

Lanusse, P., **A. Oustaloup** and **B. Mathieu**. 1993. Third Generation CRONE Control. *Proceeding of the International Conference on Systems, Man and Cybernetics, Le Touquet, France*. pp. 149–155.

Lhomme, W., **R. Trigui**, **P. Delarue**, **B. Jeanneret**, **A. Bouscayrol** and **F. Badin**. 2008. Switched Causal Modeling of Transmission with Clutch in Hybrid Electric Vehicles. *IEEE Transactions on Vehicular Technology* 57(4). pp. 2081–2088.

Li, H., **Y. Luo** and **Y. Chen**. 2010. A Fractional Order Proportional and Derivative (FOPD) Motion Controller: Tuning Rule and Experiments. *IEEE Transactions on Control Systems Technology* 18(2). pp. 516–520.

Liu, J. and **H. Peng**. 2008. Modeling and Control of a Power-Split Hybrid Vehicle. *IEEE Transactions on Control Systems Technology* 16(6). pp. 1242–1251.

Ljung, L. 1998. *System Identification*. Springer.

Lofberg, J. 2004. YALMIP: A toolbox for modeling and optimization in MATLAB. *Proceeding of the IEEE International Symposium on Computer Aided Control Systems Design, Taipei, Taiwan*. pp. 284–289.

Luo, Y. and **Y. Q. Chen**. 2009. Fractional-Order [Proportional Derivative] Controller for Robust Motion Control: Tuning Procedure and Validation. *Proceeding of the American Control Conference, St. Louis, Missouri, USA*. pp. 1412–1417.

Maciejowski, J. M. 1989. Multivariable Feedback Design. *Electronic Systems Engineering Series, Wokingham, England: Addison-Wesley,1989*.

Maciejowski, J. M. 2002. *Predictive Control: with Constraints*. Pearson education.

Mark, B., **H. Martin** and **D. Howard**. 2011. *Neural Network Toolbox User's Guide*.

Mayne, D. Q., **S. Raković**, **R. Findeisen** and **F. Allgöwer**. 2006. Robust output feedback model predictive control of constrained linear systems. *Automatica* 42(7). 1217–1222.

Melchior, P., **P. Lanusse**, **O. Cois**, **F. Dancla** and **A. Oustaloup**. 2002. Crone Toolbox for Matlab: Fractional Systems Toolbox. *Proceeding of the 41st IEEE Conference on Decision and Control, Las Vegas, Nevada, USA*. pp. 9–13.

Mohagheghi, S., **G. K. Venayagamoorthy** and **R. G. Harley**. 2005. A Dynamic Recurrent Neural Network for Wide Area Identification of a Multimachine Power System with a Facts Device. *Proceedings of the 13th IEEE International Conference on Intelligent Systems Application to Power Systems, Arlington, USA*.

Monje, C., **B. Vinagre**, **Y. Chen**, **V. Feliu**, **P. Lanusse** and **J. Sabatier**. 2004. Proposals for Fractional $pi\lambda d\mu$ Tuning. *Proceeding of The First IFAC Symposium on Fractional Differentiation and its Applications, Bordeaux, France*. pp. 115–120.

Monje, C., **B. Vinagre**, **V. Feliu** and **Y. Chen**. 2008. Tuning and auto-tuning of fractional order controllers for industry applications. *Control Engineering Practice* 16(7). pp. 798–812.

Motosugi, J., **K. Ito**, **K. Adachi** and **H. Ashizawa**. 2007. Clutch engagement control apparatus for hybrid vehicle. US Patent App. 11/753,321.

Naderi, P., **M. Mirsalim**, **M. T. Bathaee** and **R. Chini**. 2009. Fuzzy Controller Design for Parallel Hybrid Vehicle Analysis Using Forward Simulation. *Proceeding of the Conference on Vehicle Power and Propulsion, Dearborn, MI, USA*. pp. 234–241.

Nemirovskii, A. and **P. Gahinet**. 1994. The Projective Method for Solving Linear Matrix Inequalities. *Proceeding of the American Control Conference, Baltimore, Maryland, USA*. pp. 840–844.

Nørgård, M. 2000. *Neural Networks for Modelling and Control of Dynamic Systems: A Practitioner's Handbook*. Springer.

Nørgård, P. M., **O. Ravn**, **L. K. Hansen** and **N. K. Poulsen**. 1996. The NNSYD Toolbox a Matlab Toolbox for System Identification with Neural Networks. *Proceeding of the IEEE Symposium on Computer-Aided Control System Design, Dearborn, Michigan USA*. pp. 374–379.

Oh, J. J. and **S. B. Choi**. 2014. Design of Hybrid Position/Force Engagement Controller for Dry Dual Clutch Transmission without Diaphragm Spring. *Proceeding of the American Control Conference, Portland, Oregon, USA*. pp. 2618–2623.

Oustaloup, A., **B. Bluteau** and **M. Nouillant**. 1993. First Generation Scalar CRONE Control: Application to a Two DOF Manipulator and Comparison with non Linear Decoupling Control. *Proceeding of the International Conference on Systems, Man and Cybernetics, Le Touquet, France*. pp. 453–458.

Oustaloup, A., **B. Mathieu** and **P. Lanusse**. 1993. Second Generation CRONE Control. *Proceeding of the International Conference on Systems, Man and Cybernetics, Le Touquet, France*. pp. 136–142.

Padhee, S., **A. Gautam**, **Y. Singh** and **G. Kaur**. 2011. A Novel Evolutionary Tuning Method for Fractional Order PID Controller. *International Journal of soft computing and Engineering*.

Petras, I. 2011. *Fractional-Order Nonlinear Systems: Modeling, Analysis and Simulation*. Springer.

Podlubny, I. 1999. Fractional-order systems and $PI^{\alpha}D^{\beta}$-controllers. *IEEE Transactions on Automatic Control* 44(1). pp. 208–214.

Popov, A., **A. Farag** and **H. Werner**. 2005. Tuning of a PID controller Using a Multiobjective Optimization Technique Applied to A Neutralization Plant. *Proceeding of the 44th IEEE Conference on Decision and Control, European Control Conference, Seville, Spain*. 7139–7143.

Reif, K. 2010. *Konventioneller Antriebsstrang und Hybridantriebe*. Springer.

Reif, K. 2011. Bosch Autoelektrik und Autoelektronik. *Bordnetze, Sensoren und elektronische Systeme. Vieweg+ Teubner Verlag, sechste überarbeitete und erweiterte Auflage Auflage* 20.

Reif, K. and **K. E. Noreikat**. 2012. *Kraftfahrzeug-Hybridantriebe: Grundlagen, Komponenten, Systeme, Anwendungen*. Springer DE.

Sadeghpour, M., **V. De Oliveira** and **A. Karimi**. 2012. A Toolbox for Robust PID Controller Tuning Using Convex Optimization. *Proceedings of the IFAC Conference on Advances in PID Control, Brescia, Italy*. p. 158–163.

Sarangapani, J. 2006. *Neural Network Control of Nonlinear Discrete-Time Systems*. CRC press.

Scherer, C., **P. Gahinet** and **M. Chilali**. 1997. Multiobjective Output-Feedback Control via LMI Optimization. *IEEE Transactions on Automatic Control* 42(7). pp. 896–911.

Schnitzer, M., **F. Iacona**, **K. Eisenwerth** and **M. Klymenko**. 2013. Parallelhybrid-Antriebsstrang eines Fahrzeugs und Verfahren zur Steuerung desselben. DE Patent App. DE201,110,089,678.

Sciarretta, A., **M. Back** and **L. Guzzella**. 2004. Optimal Control of Parallel Hybrid Electric Vehicles. *IEEE Transactions on Control Systems Technology* 12(3). pp. 352–363.

Shah, G. and **S. Engell**. 2011. Tuning MPC for Desired Closed-Loop Performance for MIMO systems. *Proceeding of the American Control Conference, San Francisco, California, USA*. pp. 4404–4409.

Shah, P. and **S. Agashe**. 2016. Review of fractional PID controller. *Mechatronics* 38. pp. 29–41.

Shimabukuro, E., **S. Ibaraki**, **T. Shirasaka**, **M. Kishida**, **Y. Tamagawa** and **T. Hasebe**. 2003. Hybrid vehicle control device. US Patent 6,638,022.

Sjöberg, J., **Q. Zhang**, **L. Ljung**, **A. Benveniste**, **B. Delyon**, **P.-Y. Glorennec**, **H. Hjalmarsson** and **A. Juditsky**. 1995. Nonlinear Black-Box Modeling in System Identification: a Unified Overview. *Automatica* 31(12). pp. 1691–1724.

Skogestad, S. and **I. Postlethwaite**. 2007. *Multivariable Feedback Control: Analysis and Design*. Wiley New York.

Sontag, E. 1997. Recurrent Neural Networks: Some Systems-Theoretic Aspects. *Dealing with complexity: A neural network approach* pp. 1–12.

Streif, S., **M. Kögel**, **T. Bäthge** and **R. Findeisen**. 2014. Robust nonlinear model predictive control with constraint satisfaction: A relaxation-based approach. *Proceedings of the 19th IFAC World Congress on International Federation of Automatic Control*. 11073–11079.

Sturm, J. F. 1999. USING SEDUMI 1.02, A MATLAB TOOLBOX FOR OPTIMIZATION OVER SYMMETRIC CONES. *Optimization methods and software* 11(1-4). pp. 625–653.

Svaricek, F. and **N. Lachhab**. 2016. H_∞-Entwurf fraktionaler PID-Regler. *at-Automatisierungstechnik* 64(6). 407–417.

Tenoutit, M., **N. Maamri** and **J. Trigeassou**. 2011. An output feedback approach to the design of robust fractional PI and PID controllers. *Proceedings of the 18th IFAC World Congress of the International Federation of Automatic Control, Milan, Italy*.

Tenoutit, M., **N. Maamri** and **J. Trigeassou**. 2011a. A TIME MOMENTS APPROACH TO THE DESIGN OF ROBUST FRACTIONAL PID CONTROLLERS. *Proceeding of the 8th International Multi-Conference on Systems, Signals and Devices, Sousse, Tunisia*. pp. 1–7.

Tenoutit, M., **N. Maamri** and **J. Trigeassou**. 2012. Tuning of a new class of robust fractional–order proportional–integral–derivative controllers. *Proc. IMechE, Part I: J Systems and Control Engineering* 226. 486–496.

Triess, B., **C. P. Müller**, **F. C. Mößner** et al. 2007. Entwicklung und Applikation von Motor-und Getriebesteuerungen: mit der ETK-Steuergeräteschnitt Stelle. *ATZ-Automobiltechnische Zeitschrift* 109(1). pp. 32–39.

Unbehauen, H. 2007. *Regelungstechnik II Zustandsregelung, digitale und nichtlineare Regelsysteme*. Braunschweig/Wiesbaden: Vieweg.

Unbehauen, H. 2008. *Regelungstechnik I Klassische Verfahren zur Analyse und Synthese linearer kontinuierlicher Regelsysteme, Fuzzy-Regelsysteme*. Braunschweig/Wiesbaden: Vieweg.

Valério, D. and **J. da Costa**. 2005. Time-domain implementation of fractional order controllers. *Control Theory and Applications*. IET. pp. 539–552.

Van Berkel, K., **F. Veldpaus**, **T. Hofman**, **B. Vroemen** and **M. Steinbuch**. 2014. Fast and Smooth Clutch Engagement Control for a Mechanical Hybrid Powertrain. *IEEE Transactions Control Systems Technology* 22. pp. 1241–1254.

Van Der Heijden, A., **A. Serrarens**, **M. Camlibel** and **H. Nijmeijer**. 2007. Hybrid optimal control of dry clutch engagement. *International Journal of Control* 80(11). pp. 1717–1728.

Vieira, J., **F. M. Dias** and **A. Mota**. 2004. Artificial neural networks and neuro-fuzzy systems for modelling and controlling real systems: a comparative study. *Engineering Applications of Artificial Intelligence* 17(3). pp. 265–273.

Wallentowitz, H. and **K. Reif**. 2010. *Handbuch Kraftfahrzeugelektronik: Grundlagen-Komponenten-Systeme-Anwendungen*. Springer.

Wältermann, P., **H. Schütte** and **K. Diekstall**. 2004. Hardware-in-the-loop Testing of Distributed Electronic Systems. *ATZ worldwide* 106(5). pp. 6–10.

Wang, Y. and **S. Boyd**. 2010. Fast Model Predictive Control using Online Optimization. *IEEE Transactions on Control Systems Technology* 18(2). pp. 267–278.

Wang, Y. and **S. P. Boyd**. 2008. Fast Model Predictive Control Using Online Optimization. *Proceedings of the 17th IFAC World Congress, Seoul, Korea*. pp. 6974–6979.

Weinmann, A. 1991. *Uncertain Models and Robust Control*. Springer.

Werner, H. 2004. *Neural and Genetic Computing in Control*. Arbeitsbereich Regelungstechnik, Technische Universität Hamburg-Harburg.

Werner, H., **P. Korba** and **T. C. Yang**. 2003. Robust Tuning of Power System Stabilizers using LMI-Techniques. *IEEE Transactions on Control Systems Technology* 11(1). pp. 147–152.

Widrow, B. and **M. Bilello**. 1993. Adaptive Inverse Control. *Proceeding of the IEEE International Symposium on Intelligent Control, Chicago, IL, USA*. pp. 1–6.

Wilamowski, B. M. and **J. D. Irwin**. 2011. *Intelligent Systems*. CRC Press.

Witt, J., **S. Boonto** and **H. Werner**. 2007. Approximate Model Predictive Control of a 3-DOF Helicopter. *Proceeding of the 46th IEEE Conference on Decision and Control, New Orleans, LA, USA*. pp. 4501–4506.

Xu, J. X., **J. Donne** and **U. Ozguner**. 1991. Synthesis of Feedback Linearization and Variable Structure Control with Neural Net Compensation. *Proceedings of the IEEE International Symposium on Intelligent Control, Arlington, USA*. pp. 184–189.

A Experimental Results

Synchronization Task: Engine temperature $T_{Eng} = 69.6°$

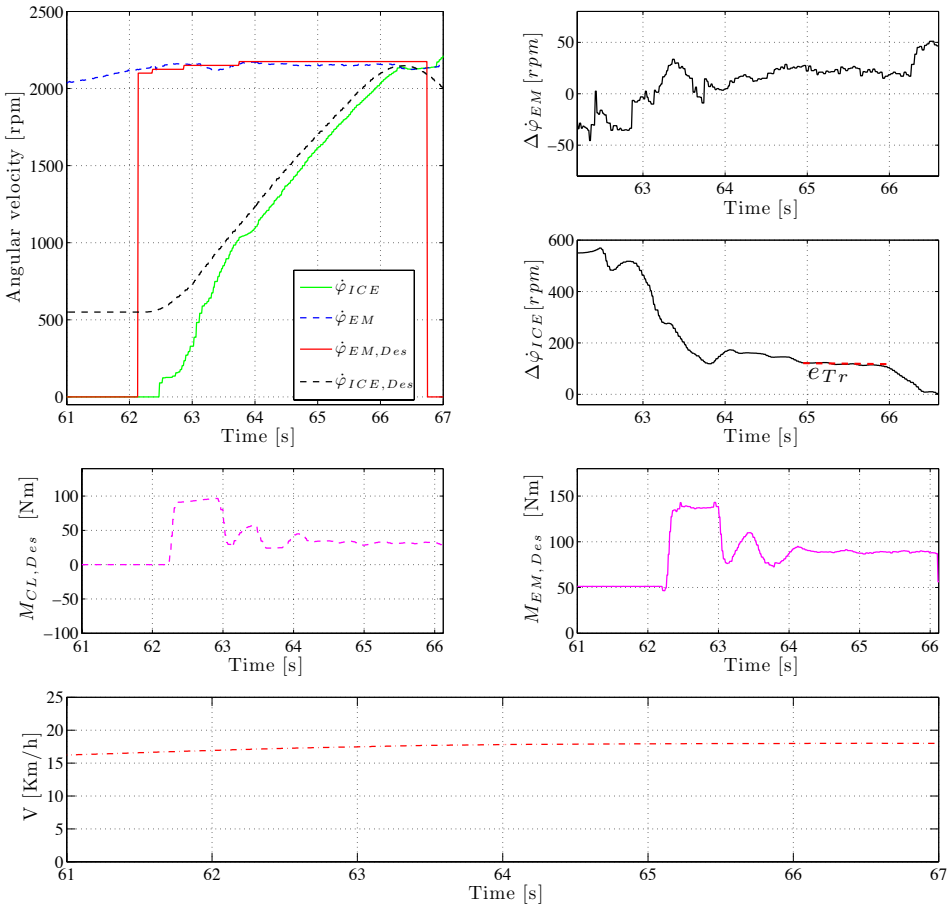

Figure A.1: Synchronization of the EM and ICE speeds (experimental results test vehicle), (top, left) ICE, EM and desired ICE and EM angular velocities $\dot{\varphi}_{ICE}$ (green), $\dot{\varphi}_{EM}$ (dash blue), $\dot{\varphi}_{EM,Des}$ (red) and $\dot{\varphi}_{ICE,Des}$ (dash black), (top, right) variations of the EM and ICE angular velocities $\Delta\dot{\varphi}_{EM}$ and $\Delta\dot{\varphi}_{ICE}$, (centre left) desired clutch torque $M_{CL,Des}$, (centre right) desired EM torque $M_{EM,Des}$, (bottom) vehicle velocity V

Figure A.1 shows the experimental results of a synchronization task for $T_{Eng} = 69.6°$. As in the previous case, there is no break down of the angular velocity $\dot{\varphi}_{EM}$. Thereby, the velocity $\dot{\varphi}_{ICE}$ follows its reference $\dot{\varphi}_{ICE,Des}$ with a smaller tracking error 123 rpm. Compared to the previous case shown in Figure 5.22, a change in this parameter is an indicator that the gain of the system is also changing. However, due to the robustness of the controller, there is no deterioration in the performance by means of overshoots for the angular velocity $\dot{\varphi}_{EM}$.

Synchronization Task: Engine temperature $T_{Eng} = 74.15°$

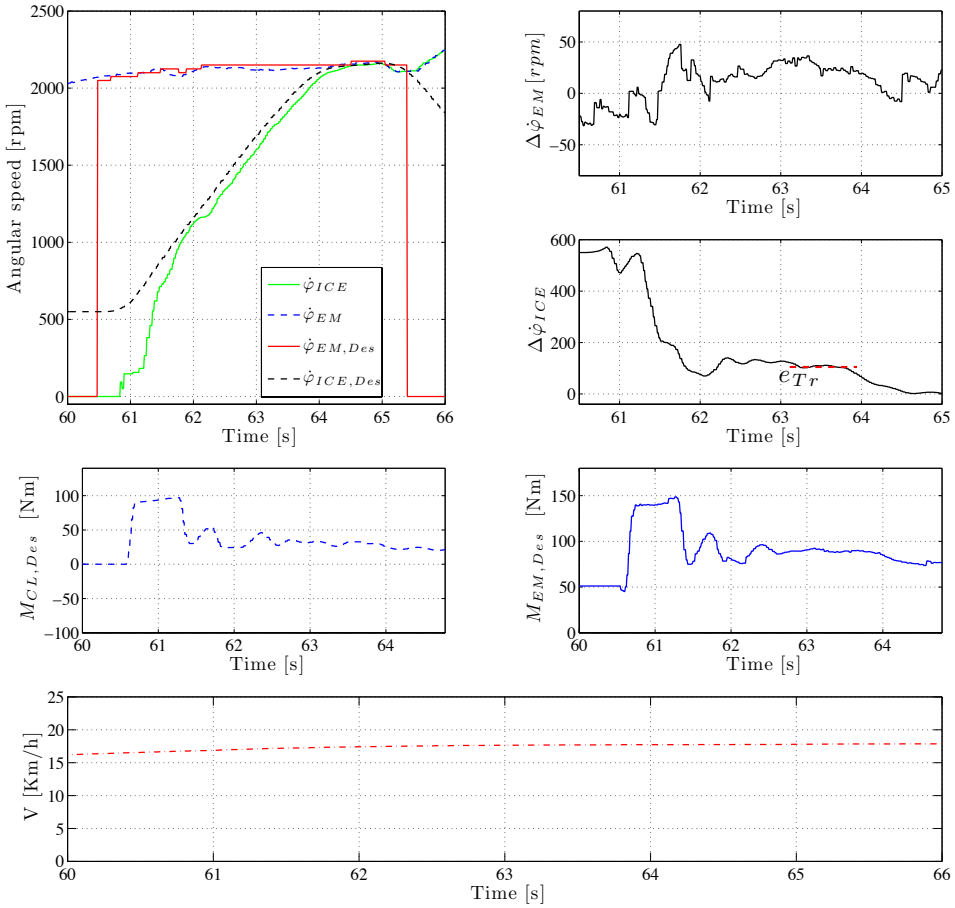

Figure A.2: Synchronization of the EM and ICE speeds (experimental results test vehicle), (top, left) ICE, EM and desired ICE and EM angular velocities $\dot{\varphi}_{ICE}$ (green), $\dot{\varphi}_{EM}$ (dash blue), $\dot{\varphi}_{EM,Des}$ (red) and $\dot{\varphi}_{ICE,Des}$ (dash black), (top, right) variations of the EM and ICE angular velocities $\Delta\dot{\varphi}_{EM}$ and $\Delta\dot{\varphi}_{ICE}$, (centre left) desired clutch torque $M_{CL,Des}$, (centre right) desired EM torque $M_{EM,Des}$, (bottom) vehicle velocity V

The experimental results of a synchronization task for $T_{Eng} = 74.15°$ are shown in Figure A.2. The variation of the angular velocity $\Delta\dot{\varphi}_{EM}$ is below 50 rpm. As in the previous cases, the controller provides the required performance (no breakdown of the EM angular velocity). The tracking error is here $e_{Tr} = 110$ rpm.

Synchronization Task: Engine temperature $T_{Eng} = 77.8°$

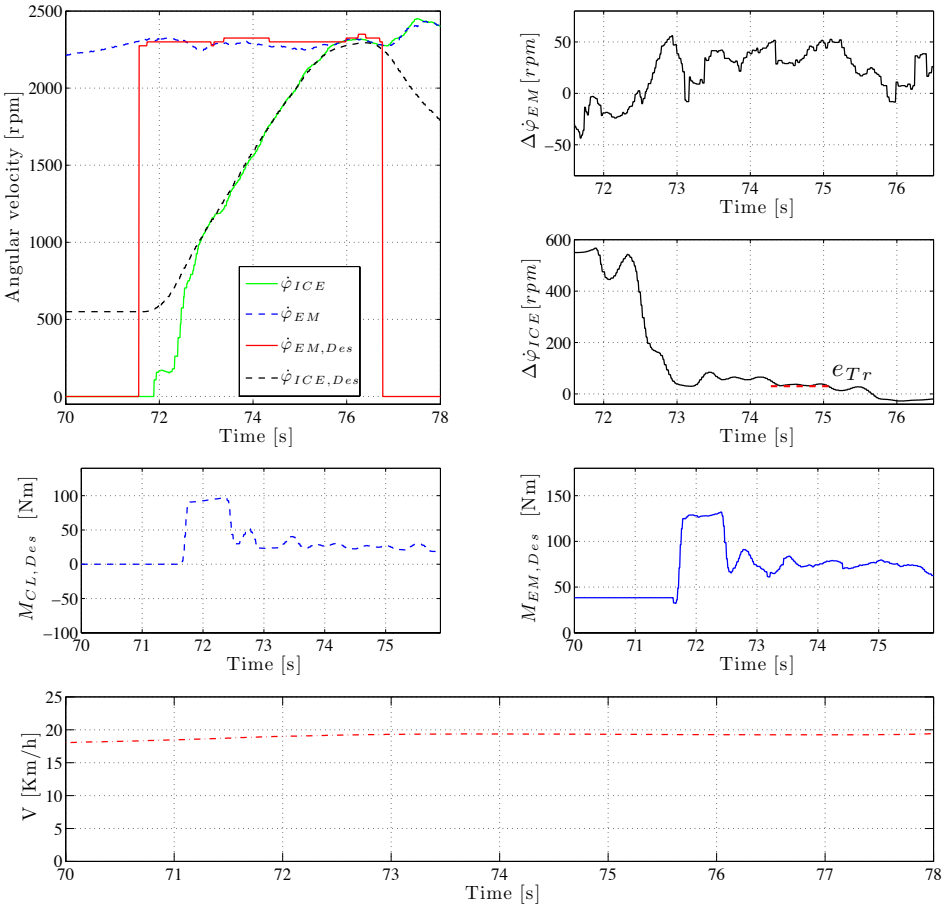

Figure A.3: Synchronization of the EM and ICE speeds (experimental results test vehicle), (top, left) ICE, EM and desired ICE and EM angular velocities $\dot{\varphi}_{ICE}$ (green), $\dot{\varphi}_{EM}$ (dash blue), $\dot{\varphi}_{EM,Des}$ (red) and $\dot{\varphi}_{ICE,Des}$ (dash black), (top, right) variations of the EM and ICE angular velocities $\Delta\dot{\varphi}_{EM}$ and $\Delta\dot{\varphi}_{ICE}$, (centre left) desired clutch torque $M_{CL,Des}$, (centre right) desired EM torque $M_{EM,Des}$, (bottom) vehicle velocity V

Figure A.3 shows the experimental results of a synchronization task for $T_{Eng} = 77.8°$. This time, the desired trajectory for the ICE angular velocity $\dot{\varphi}_{ICE}$ with the tracking error $e_{Tr} = 32\ rpm$. Also for this case, there is no deterioration of the angular velocity $\dot{\varphi}_{EM}$, which is confirmed through a variation $\Delta\dot{\varphi}_{EM}$ below 56 rpm is achieved.

Synchronization Task: Engine temperature $T_{Eng} = 79.6°\mathbf{C}$

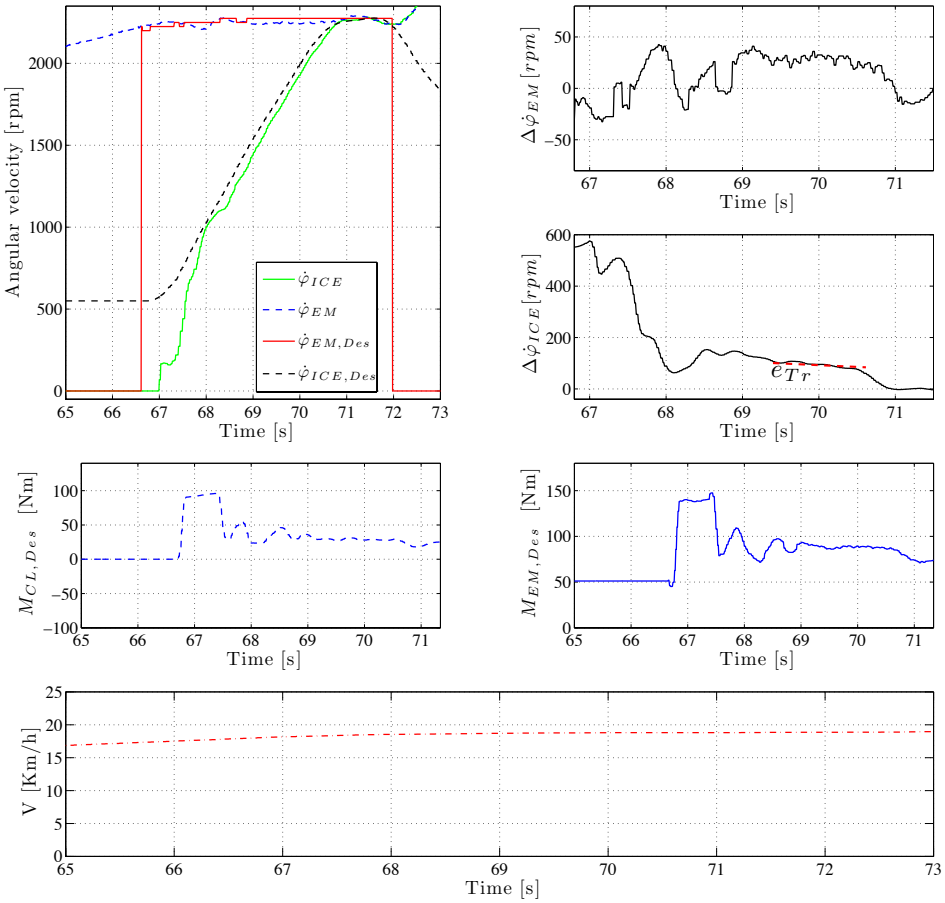

Figure A.4: Synchronization of the EM and ICE speeds (experimental results test vehicle), (top, left) ICE, EM and desired ICE and EM angular velocities $\dot{\varphi}_{ICE}$ (green), $\dot{\varphi}_{EM}$ (dash blue), $\dot{\varphi}_{EM,Des}$ (red) and $\dot{\varphi}_{ICE,Des}$ (dash black), (top, right) variations of the EM and ICE angular velocities $\Delta\dot{\varphi}_{EM}$ and $\Delta\dot{\varphi}_{ICE}$, (centre left) desired clutch torque $M_{CL,Des}$, (centre right) desired EM torque $M_{EM,Des}$, (bottom) vehicle velocity V

In Figure A.4, the ICE temperature T_{Eng} is about 80°C. The controller still achieves a low deviation $\Delta\dot{\varphi}_{EM}$, which is smaller than 50 rpm. Thereby, the ICE velocity $\dot{\varphi}_{ICE}$ follows its reference $\dot{\varphi}_{ICE,Des}$ with an achieved tracking error of about $e_{Tr} = 103$ rpm.

Synchronization Task: Engine temperature $T_{Eng} = 83.1°\mathbf{C}$

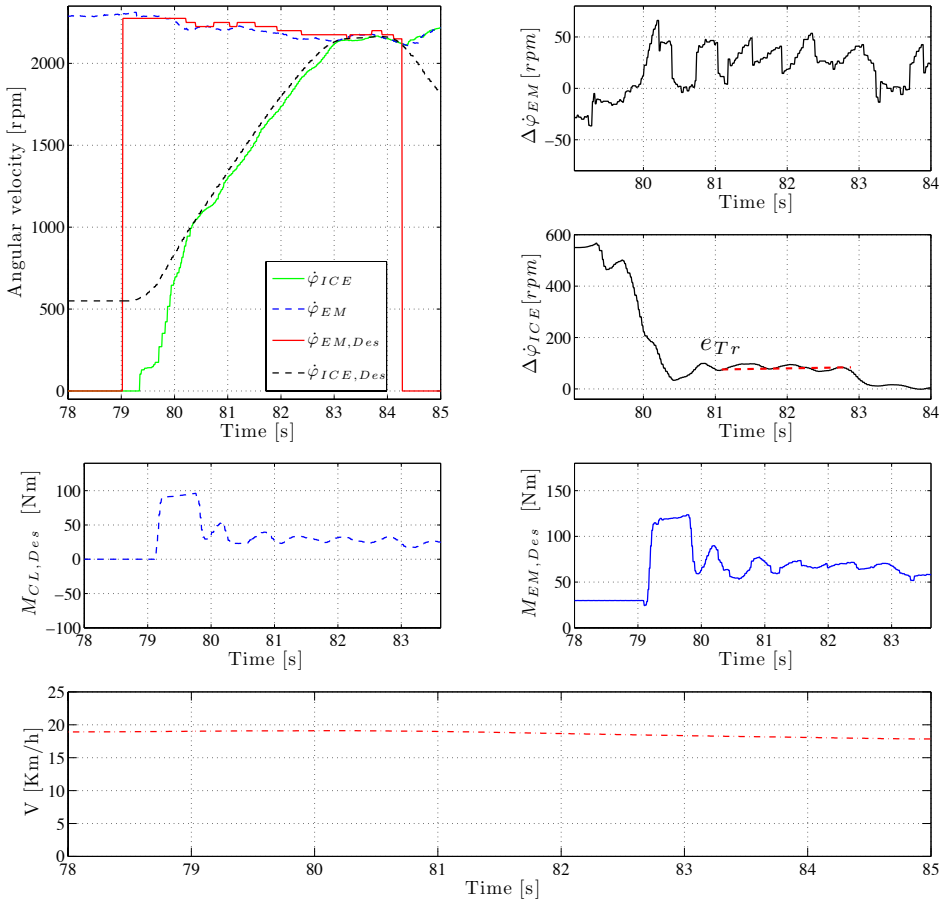

Figure A.5: Synchronization of the EM and ICE speeds (experimental results test vehicle), (top, left) ICE, EM and desired ICE and EM angular velocities $\dot{\varphi}_{ICE}$ (green), $\dot{\varphi}_{EM}$ (dash blue), $\dot{\varphi}_{EM,Des}$ (red) and $\dot{\varphi}_{ICE,Des}$ (dash black), (top, right) variations of the EM and ICE angular velocities $\Delta\dot{\varphi}_{EM}$ and $\Delta\dot{\varphi}_{ICE}$, (centre left) desired clutch torque $M_{CL,Des}$, (centre right) desired EM torque $M_{EM,Des}$, (bottom) vehicle velocity V

The last experiment was performed at the ICE temperature $T_{Eng} = 83.1°$C. In this case, we remark a deterioration of the EM velocity $\dot{\varphi}_{EM}$, which is acceptable. In this case, the robustness performance is not fully achieved. The simulation results for the case that the static gain V varies more than 50% given in Figure 5.20 shows that in this case the robustness performance is not guaranteed. For this reason, the variation of the static gain related to the results in Figure A.5 is probably higher than 50%, which is outside the robustness range of the NN controller.

B Mathematical Preliminaries and Robust Control

In this section, the main mathematical tools used in this work are presented. Only definitions and theorems, which contribute to a better understanding are presented and discussed. For more details, the reader can see the related references or the Appendix. When necessary, some definitions are accompanied with examples to demonstrate their practical aspects. Basic mathematical definitions are presented in Section B.1. In Section B.2, linear matrix inequalities are discussed. The introduction of fractional calculus and the related fractional order functions are discussed in Section B.3. In Section B.4, the definition of the class of uncertainty considered in this work is presented. At the end of this Chapter, robust control is briefly revised.

B.1 Linear Algebra

First, the notation used along this thesis is given. Scalars, vectors and matrices are denoted by lowercase letters x, boldface lowercase letters $\mathbf{x}$ and boldface capital letters $\mathbf{X}$, respectively. Time signals are denoted whether by lowercase letters $x(t)$ for the scalar case or by $\mathbf{X}(t)$ for the vector case. Laplace transform is denoted by capital letters $X(s)$ for scalar signals and boldface capital letters $\mathbf{X}(s)$ for vector signals. The following notations are used to denote the definiteness of matrices. A positive definite matrix $\mathbf{A}$ is denoted by $\mathbf{A} > 0$, semi-positive definite by $\mathbf{A} \geq 0$, negative definite by $\mathbf{A} < 0$ and semi-negative definite by $\mathbf{A} \leq 0$. An identity matrix $\in \mathbb{R}^{n \times n}$ is denoted by $\mathbf{I}_n$. A n by m matrix with zeros entries is denoted by $\mathbf{0}_{n \times m}$. The transposed of a matrix $\mathbf{A}$ is denoted by $\mathbf{A}^T$ and the conjugate transposed (Hermitian) by $\mathbf{A}^H$. $\sigma(\mathbf{A})$ denotes the singular values of a given matrix $\mathbf{A}$ and $\bar{\sigma}(\mathbf{A})$ denotes the maximal singular value.

Vector 2-norm (Skogestad and Postlethwaite 2007, p. 550)

The vector 2-norm is used to define the cost function of the optimization problem in Section 3. Therefore, the definition of this norm is presented here. It is as follows

$$\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{m} |\mathbf{x}_i|^2} \tag{B.1}$$

or in vector form

$$\|\mathbf{x}\|_2 = \sqrt{\mathbf{x}^H \mathbf{x}} \tag{B.2}$$

for a given vector $\mathbf{x} \in \mathbb{C}^m$, where $\mathbf{x}^H$ denotes the conjugate transpose of $\mathbf{x}$. For real vectors, $\mathbf{x}^H$ can be replaced by the transposed $\mathbf{x}^T$.

Signal 2-norm (Skogestad and Postlethwaite 2007, p. 558)

Due to its use to define the H_∞ norm, the signal 2-norm is presented. It is defined as

follows

$$\|y(t)\|_2 = \sqrt{\int_{-\infty}^{+\infty} |y(\tau)|^2 d\tau} \tag{B.3}$$

with $y(t)$ is a given scalar time signal. The lower bound can be replaced with zero as $y(t) = 0$ for $t < 0$. For a vector time signal $\mathbf{y}(t)$, the 2-norm turns out to be

$$\|\mathbf{y}(t)\|_2 = \sqrt{\int_{0}^{+\infty} \sum_i |\mathbf{y}_i(\tau)|^2 d\tau}. \tag{B.4}$$

H_∞-norm (Skogestad and Postlethwaite 2007, p. 163-166)

The H_∞ norm plays an important role in control theory. Especially, it is known for its use to derive the H_∞ controller problem. In this work, the H_∞ norm is used to formulate the fractional order controller problem in Section 2. For this reason, it is necessary to present its definition. Moreover, a brief interpretation for SISO systems is given.

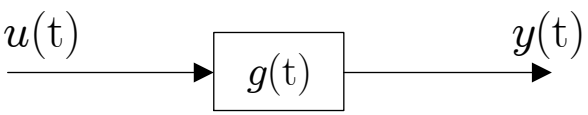

Figure B.1: LTI-configuration with input $u(t)$, output $y(t)$ and impulse response $g(t)$

Figure B.1 shows a proper linear stable system with impulse response $g(t)$, input $u(t)$ and output $y(t)$. The first definition of the H_∞ norm is given in terms of the time domain response of the plant. It is as follows

$$\|G(s)\|_\infty = \max_{u(t)\neq 0} \frac{\|y(t)\|_2}{\|u(t)\|_2} = \max_{\|u(t)\|_2=1} \|y(t)\|_2. \tag{B.5}$$

Another definition or interpretation of the H_∞ norm is given in terms of the frequency response of the transfer function $G(s)$. It is as follows

$$\|G(s)\|_\infty = \sup_\omega |(G(j\omega)|. \tag{B.6}$$

This definition can be interpreted as the maximal magnitude of the system over all frequencies. Figure B.2 shows an example of the magnitude plot of the system $G(s)$. Thereby, the H_∞ norm is the maximal magnitude over all frequencies. It is achieved at the frequency ω_T.

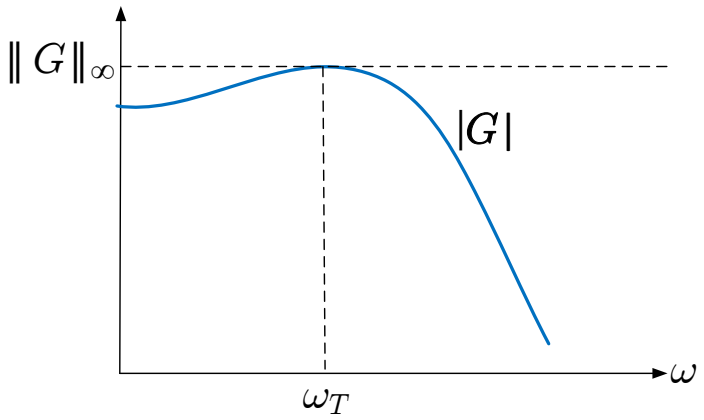

Figure B.2: H_∞ norm of the SISO system G, or maximal gain over ω

For MIMO systems, the magnitude $|G|$ shown in Figure B.2 is replaced by the maximal singular value $\bar{\sigma}(G)$.

Schur complement (Boyd et al. 1994, p. 7-8)

Another important mathematical tool used in this work is the Schur complement. It can be used to transform a nonlinear matrix inequality into a linear matrix inequality. This idea is applied on the control problem in Section 3. For this reason, consider the two inequalities

$$\mathbf{Q}(x) > 0 \text{ and } \mathbf{Q}(x) - \mathbf{S}(x)\mathbf{R}(x)^{-1}\mathbf{S}(x)^T > 0. \tag{B.7}$$

The symmetric matrices $\mathbf{Q}(x)$ and $\mathbf{R}(x)$ and the matrix $\mathbf{S}(x)$ depend affine on the parameter x. The inequality on the right hand side of the expression (B.7) is nonlinear. This is due to the term $\mathbf{R}(x)^{-1}$ and the left and right multiplication with $\mathbf{S}(x)$ and $\mathbf{S}(x)^T$, respectively. The two inequalities in the above expression are equivalent to the condition

$$\begin{bmatrix} \mathbf{Q}(x) & \mathbf{S}(x) \\ \mathbf{S}^T(x) & \mathbf{R}(x) \end{bmatrix} > 0. \tag{B.8}$$

This expression becomes linear in the parameter x. It defines a linear matrix inequality. The Schur complement was used in many contributions to turn nonlinear problems into LMIs. It plays an important role in deriving LMI conditions to design H_∞ controllers, (Apkarian and Gahinet 1995). In (de Oliveira et al. 1999, Theorem 1-2), the authors use the Schur complement to derive stability and robust stability conditions for linear discrete-time systems. For a better understanding of this mathematical tool, consider the following least-square problem (Boyd and Vandenberghe 2004, p. 301)

Problem B.1 :

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} && \underbrace{\|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2^2}_{f(\mathbf{x})} \\ &\text{subject to} && \mathbf{x} > 0 \end{aligned} \tag{B.9}$$

with a given matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, a given vector $\mathbf{b} \in \mathbb{R}^{m\times 1}$ and the parameter vector $\mathbf{x} \in \mathbb{R}^{n\times 1}$. The expression (B.9) is a constrained least-square problem. The objective function

$$f(\mathbf{x}) = (\mathbf{A}\mathbf{x}-\mathbf{b})^T(\mathbf{A}\mathbf{x}-\mathbf{b}) = \mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x} - 2\mathbf{x}^T\mathbf{b} + \mathbf{b}^T\mathbf{b}. \tag{B.10}$$

is quadratic in the parameter vector $\mathbf{x}$. Using the Schur complement, the constrained optimization problem (B.9) can be transformed into a LMI. This is achieved by considering the equivalent problem

Problem B.2 :

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} && \gamma \\ &\text{subject to} && \underbrace{(\mathbf{A}\mathbf{x}-\mathbf{b})^T(\mathbf{A}\mathbf{x}-\mathbf{b})}_{f(\mathbf{x})} < \gamma \ \text{ and } \ \mathbf{x} > 0 \end{aligned} \tag{B.11}$$

with the scalar cost function γ. Now introducing the matrices $\mathbf{Q}$, $\mathbf{R}$ and $\mathbf{S}$ as follows

$$\underbrace{(\gamma)}_{\mathbf{Q}} - \underbrace{(\mathbf{A}\mathbf{x}-\mathbf{b})^T}_{\mathbf{S}}\underbrace{\mathbf{I}_N}_{\mathbf{R}^{-1}}\underbrace{(\mathbf{A}\mathbf{x}-\mathbf{b})}_{\mathbf{S}^T} > 0 \tag{B.12}$$

and then applying the Schur complement (B.8) on the inequality (B.12) leads to

$$\begin{bmatrix} \gamma & (\mathbf{A}\mathbf{x}-\mathbf{b})^T \\ (\mathbf{A}\mathbf{x}-\mathbf{b}) & \mathbf{I}_N \end{bmatrix} > 0. \tag{B.13}$$

The quadratic matrix inequality (B.12) is transformed into the linear matrix inequality (B.13) in $\mathbf{x}$. The overall least square optimization problem is now given in terms of the following LMI problem

Problem B.3 :

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} && \gamma \\ &\text{subject to} && \begin{bmatrix} \gamma & (\mathbf{A}\mathbf{x}-\mathbf{b})^T \\ (\mathbf{A}\mathbf{x}-\mathbf{b}) & \mathbf{I}_N \end{bmatrix} > 0, \\ & && \mathbf{x} > 0. \end{aligned} \tag{B.14}$$

Problem (B.14) consists of minimizing a linear objective given by γ and two linear matrix

inequalities describing the constraints. This LMI problem can be solved using existing LMI solvers.

B.2 Linear Matrix Inequalities

The benefit of representing optimization problems in terms of LMIs is that the resulting problem is convex. Using modern algorithms such as (Labit et al. 2002) or (Gahinet et al. 1995), LMIs can be solved numerically very efficient. The number of books and papers dealing with LMIs is considerable high. For example, a good survey is given by (Boyd and Vandenberghe 2004) and (Boyd et al. 1994). In this section, a very brief review on LMIs is presented. Moreover, a control example is presented to show the field of their application.

Linear matrix inequality is a special form of inequalities, which appears in control theory. In this case, the optimization parameter enters the matrix inequality linearly. As it is mentioned in (Boyd et al. 1994), the first contribution was made by Lyapunov (about 1880). He presented the so-called Lyapunov theory as a tool to check the stability of dynamic systems. For this purpose, consider the following system

$$\dot{\mathbf{x}}(t) = \mathbf{A}\,\mathbf{x}(t) \tag{B.15}$$

with $\mathbf{x} \in \mathbb{R}^n$ and and $\mathbf{A} \in \mathbb{R}^{n\times n}$ denote the state vector and system matrix, respectively. Many approaches exist to check the stability of this system. One approach is given in terms of the so-called Lyapunov matrix. In this context, it is well known that the system (B.15) is stable (in the sense that all trajectories converge to zero) if and only if a positive definite matrix $\mathbf{P}$ exists such that the following LMI condition (Boyd et al. 1994, p. 2)

$$\mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} < 0 \tag{B.16}$$

holds. One way to solve problem (B.16) is to transform it into a linear matrix equality (Boyd et al. 1994, p. 25). This can be done using any auxiliary symmetric positive definite matrix $\mathbf{Q}$ and then solve the following equality

$$\mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} = -\mathbf{Q} \tag{B.17}$$

in the matrices $\mathbf{P}$ and $\mathbf{Q}$.

In general, the matrix inequality (B.16) does not have an analytical solution. Especially, for high dimensional problems it becomes very difficult to solve it by hand. Thanks to the work of Karmarkar (Karmarkar 1984) in the early 1980, the introduction of projection methods (Nemirovskii and Gahinet 1994) and interior point methods (Boyd and Vandenberghe 2004), LMIs become more popular. This is due to the fact that algorithms based on these methods solve LMI problems efficiently and in polynomial time, see (Labit et al. 2002) and (Gahinet et al. 1995).

Actually, the general inequality form which defines LMI problems is as follows (Boyd et al. 1994, p. 7)

$$\mathbf{S}_0 + p_1\mathbf{S}_1 + p_2\mathbf{S}_2 + \cdots + p_N\mathbf{S}_N > 0, \tag{B.18}$$

with $\mathbf{p} = [p_0 \quad p_1 \quad p_2 \quad \cdots \quad p_N]^T$ represents the vector of decision variables. The matrices $\mathbf{S}_0$ to $\mathbf{S}_N$ are given symmetric matrices. A more general LMI form is given by

Problem B.4 :

$$\begin{aligned} \underset{\mathbf{p}}{\text{minimize}} \quad & f(\mathbf{p}) = \mathbf{c}^T\mathbf{p} \\ \text{subject to} \quad & \mathbf{S}_0 + p_1\mathbf{S}_1 + \cdots + p_N\mathbf{S}_N > 0. \end{aligned} \tag{B.19}$$

The scalar function $f(\mathbf{p})$ is the objective function, which is linear in the parameter vector $\mathbf{p} \in \mathbb{R}^{n\times 1}$. The vector $\mathbf{c} \in \mathbb{R}^{n\times 1}$ defines the objective, which will be minimized. The optimization problem (B.19) represents a minimization of a linear objective subject to a linear matrix inequality. In this case, the resulting optimization problem is convex.

Many control problems can be formulated as LMI problems. Mostly, the original problem is nonlinear. Using linear algebra, the nonlinear problem can be converted into a LMI form. For a better understanding, a simple example is given. In control theory, one of the common problems is the computation of a stabilizing state feedback gain for linear systems. For this purpose, consider the following LTI system (Boyd et al. 1994, p. 100)

$$\dot{\mathbf{x}}(t) = \mathbf{A}\,\mathbf{x}(t) + \mathbf{b}\,u(t). \tag{B.20}$$

with $\mathbf{A} \in \mathbb{R}^{2\times 2}$, $\mathbf{x} \in \mathbb{R}^{2\times 1}$ and $\mathbf{b} \in \mathbb{R}^{2\times 1}$. The goal is to compute a state feedback vector $\mathbf{f} \in \mathbb{R}^{2\times 1}$ with $u(t) = \mathbf{f}^T\,\mathbf{x}(t)$ that stabilizes the plant (B.20). Applying now the Lyapunov stability condition, the state feedback problem is defined as follows

$$\mathbf{P}(\mathbf{A} + \mathbf{b}\mathbf{f}^T)^T + (\mathbf{A} + \underbrace{\mathbf{b}\mathbf{f}^T}_{Bilinear})\mathbf{P} < 0. \tag{B.21}$$

Due to the multiplication between the vector $\mathbf{f}^T$ and the Lyapunov matrix $\mathbf{P}$, the inequality (B.21) is not a LMI. It is a Bilinear Matrix Inequality (BMI). Now introducing $\mathbf{y} = \mathbf{f}^T\mathbf{P}$ and substituting it in the BMI (B.21), the following

$$\mathbf{P}\mathbf{A}^T + \mathbf{y}^T\mathbf{b}^T + \mathbf{A}\mathbf{P} + \mathbf{b}\mathbf{y} < 0, \tag{B.22}$$

defines the state feedback problem in LMI form. Inequality (B.22) is a LMI in the auxiliary variable $\mathbf{y}$ and the Lyapunov matrix $\mathbf{P}$. The general form (B.18) of the LMI (B.22) can

be obtained using

$$\mathbf{P} = p_1 \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} + p_2 \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + p_3 \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \tag{B.23}$$

and

$$\mathbf{y} = y_1 \begin{bmatrix} 1 & 0 \end{bmatrix} + y_2 \begin{bmatrix} 0 & 1 \end{bmatrix}. \tag{B.24}$$

Substituting the expressions (B.23) and (B.24) in the inequality (B.22) leads to the following general LMI form

$$\begin{aligned} p_1 \overbrace{\mathbf{sym}\left(\mathbf{A}\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\right)}^{\mathbf{S_1}} + p_2 \overbrace{\mathbf{sym}\left(\mathbf{A}\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}\right)}^{\mathbf{S_2}} + p_3 \overbrace{\mathbf{sym}\left(\mathbf{A}\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}\right)}^{\mathbf{S_3}} \\ + p_4 \underbrace{\mathbf{sym}\left(\mathbf{b}\begin{bmatrix} 1 & 0 \end{bmatrix}\right)}_{\mathbf{S_4}} + p_5 \underbrace{\mathbf{sym}\left(\mathbf{b}\begin{bmatrix} 0 & 1 \end{bmatrix}\right)}_{\mathbf{S_5}} < 0, \end{aligned} \tag{B.25}$$

with $p_4 = y_1$ and $p_5 = y_2$. The term $\mathbf{sym}(\cdot)$ is defined as follows

$$\mathbf{sym}(\mathbf{M}) = \mathbf{M} + \mathbf{M}^T,$$

for a given matrix $\mathbf{M}$. After solving the LMI (B.25), the state feedback vector $\mathbf{f}^T$ is equal to $\mathbf{y}\mathbf{P}^{-1}$. Due to the fact that the Lyapunov matrix is positive definite, the inverse of $\mathbf{P}$ exists. To solve LMIs, the modeling language YALMIP (Lofberg 2004) and the optimization algorithm SeDuMi provided by (Labit et al. 2002) are used.

B.3 Fractional Order Derivative and Integral

In this section, a brief introduction to fractional order operators is given. Especially, definitions of fractional order derivative and integral are discussed. Moreover, the frequency domain interpretation and the comparison to integer order integral and derivative are provided. In this context, the Euler's Gamma function is introduced. It is defined as follows (Petras 2011, p. 7)

$$\Gamma(n) = \int_0^\infty t^{(n-1)} e^{-t} dt. \tag{B.26}$$

The definition above consists a generalization of the classical factorial

$$\Gamma(n) = (n-1)! \tag{B.27}$$

for integer numbers n. Fractional derivatives and integrals generalize differentiation and integration to noninteger orders. Now giving a fractional order $\alpha \in \mathbb{R}$, the integral and

differential operator are defined as follows (Petras 2011, p. 9)

$$ {}_aD_t^\alpha = \begin{cases} \frac{d^\alpha}{dt^\alpha}, & \alpha > 0 \\ 1, & \alpha = 0 \\ \int_a^t (d\tau)^\alpha \quad , & \alpha < 0 \end{cases} \tag{B.28} $$

with a and t define the bounds of the integral. It exists several definitions of fractional operators. In this brief introduction, the currently used definitions are presented.

Grünwald-Letnikov definition (Petras 2011, p. 9)

Before presenting this definition, recall first the definition of the integer order n-derivative of a function $f(t)$ as follows

$$ \frac{d^n}{dt^n} f(t) = f^{(n)}(t) = \lim_{h \to 0} \frac{1}{h^n} \sum_{j=0}^{n} (-1)^j \begin{pmatrix} n \\ j \end{pmatrix} f(t - jh) \tag{B.29} $$

with the binomial coefficient

$$ \begin{pmatrix} n \\ j \end{pmatrix} = \frac{n(n-1)(n-2)\cdots(n-j+1)}{j!} = \frac{n!}{j!(n-j)!}. \tag{B.30} $$

Now generalizing the order n to real numbers, the fractional order derivative can be written as

$$ D_t^\alpha f(t) = \lim_{h \to 0} \frac{1}{h^\alpha} \sum_{j=0}^{\infty} (-1)^j \begin{pmatrix} \alpha \\ j \end{pmatrix} f(t - jh) \tag{B.31} $$

with α being a real number. To calculate the binomial coefficients for α, the following expression

$$ \begin{pmatrix} \alpha \\ j \end{pmatrix} = \frac{\alpha!}{j!(\alpha - i)!} = \frac{\Gamma(\alpha + 1)}{\Gamma(j+1)\Gamma(\alpha - j + 1)} \tag{B.32} $$

is used.

Riemann-Liouville definition (Petras 2011, p. 11)

The Riemann-Liouville definition provides another method to compute frational integrals and derivatives. Contrary to the previous definition, an integral based expression is given. The following defines fractional integration of a given order α for the function $f(t)$ (Petras 2011, p. 11)

$$ {}_aI_t^\alpha f(t) = {}_aD_t^{-\alpha} = \frac{1}{\Gamma(-\alpha)} \int_a^t \frac{f(\tau)}{(t-\tau)^{\alpha+1}} d\tau \tag{B.33} $$

with α and $a \in \mathbb{R}$ and $\alpha < 0$. On the other hand, the fractional derivation with Riemann-Liouville is given by

$$ {}_aD_t^{\alpha} = \frac{1}{\Gamma(n-\alpha)} \frac{d^n}{dt^n} \int_a^t \frac{f(\tau)}{(t-\tau)^{\alpha-n+1}} d\tau. \tag{B.34} $$

Laplace Transform (Petras 2011, p. 12-15)

The definitions presented above are suitable for analysis purposes. For the seek of control, it is more convenient and preferable to have an insight into the frequency response of fractional operators. For this reason, the Laplace transform of fractional orders is considered. For the case of the fractional derivative of a function $f(t)$ and with the assumption that all initial conditions are zero, this is given by

$$ L\{{}_0D_t^{\alpha} f(t); s\} = s^{\alpha} F(s) \tag{B.35} $$

for $\alpha > 0$. Comparing the Laplace transform of the fractional order α with the Laplace transform of the n-derivative of the function $f(t)$ given by

$$ L\{f^n(t); s\} = s^n F(s) \tag{B.36} $$

with $n \in \mathbb{N}$, the fractional order generalizes the use of the Laplace operator to fractional derivatives s^{α}. The counterpart to definition (B.35) is the Laplace transform of the fractional integral. This is given by

$$ L\{{}_0D_t^{\beta} f(t)\} = s^{\beta} F(s) \tag{B.37} $$

with $\beta < 0$.

Implementation of Fractional Orders

It exists many formulas to implement fractional orders. Generally, they can be divided into two main approaches, those which are based on continuous-time approximations and others which are based on discrete-time approximations. In (Valério and da Costa 2005), this issue is discussed in details. A summary of the main results is given in the Appendix C. In this work, the continuous-time approximation given by (Melchior et al. 2002) is used. It is known as the CRONE approximation. The main benefit of using this method is the possibility to specify the frequency range in which the approximation of fractional orders should be valid. This prevents unnecessary high order approximations. Due to its importance for this work, the CRONE approximation is briefly discussed. Given now a

fractional function s^{α}, the following representation

$$\prod_{i=1}^{N} \frac{1+\frac{s}{\omega_i^{'}}}{1+\frac{s}{\omega_i}}, \qquad \omega_i^{'}, \omega_i \quad \in \mathbb{R} \tag{B.38}$$

can be used to approximate the order α in the given frequency range $[\omega_{min}\ \omega_{max}]$. The frequencies $\omega_i^{'}$ and ω_i are computed by first determining v and η as follows

$$v = \left(\frac{\omega_{max}}{\omega_{min}}\right)^{\left(\frac{\alpha}{N}\right)}, \quad \eta = \left(\frac{\omega_{max}}{\omega_{min}}\right)^{\left(\frac{1-\alpha}{N}\right)} \tag{B.39}$$

and then compute

$$\omega_i = \omega_i^{'} v \ \text{ and } \ \omega_{i+1}^{'} = \omega_i \eta. \tag{B.40}$$

The order of the approximation N can be chosen depending on the frequency range given by ω_{min} and ω_{max} in which the fractional order approximation is valid.

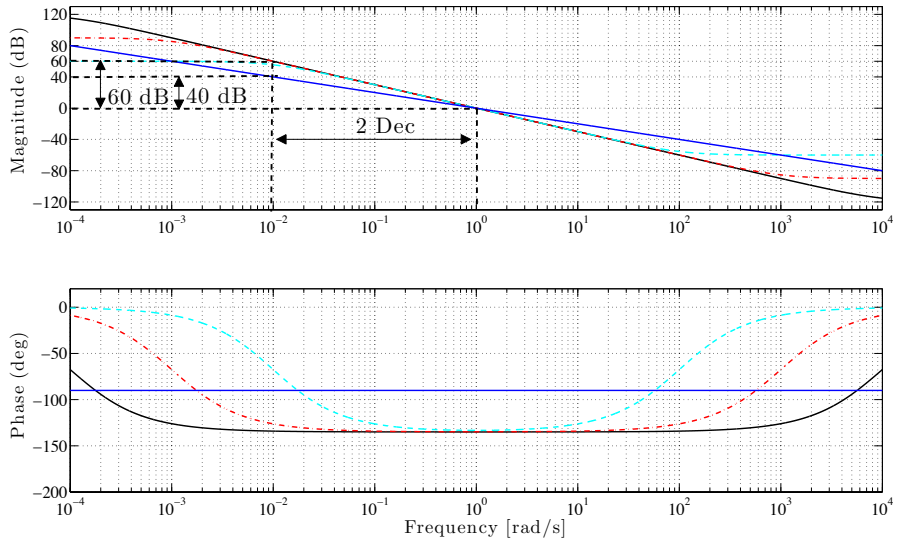

Figure B.3: Bode plot of the CRONE approximation of the fractional integrator $\frac{1}{s^{1.5}}$ for the frequency ranges, $[10^{-4}\ 10^{4}]\,rad/s$ (black), $[10^{-3}\ 10^{3}]\,rad/s$ (dashed red) and $[10^{-2}\ 10^{2}]\,rad/s$ (dashdot cyan), in comparison Bode plot of the integrator $\frac{1}{s}$ (blue)

To get an insight into the frequency response of fractional orders, the Bode diagram of the following fractional integral

$$\frac{1}{s^{\alpha}} \approx \prod_{i=1}^{N} \frac{1+\frac{s}{\omega_i^{'}}}{1+\frac{s}{\omega_i}}, \qquad \omega_i^{'}, \omega_i \quad \in \mathbb{R} \tag{B.41}$$

with $\alpha = 1.5$ is considered. Moreover, three frequency ranges are considered. The results are shown in Figure B.3. Considering the magnitude plot, the slope of the classical integrator is given by $-20\ dB/dec$. For the fractional integrator, the slope is a function of the fractional order. Precisely, it is given by the order α multiplied by $-20\ dB/dec$. For the example above, it is $-1.5 \times 20\ dB/dec$. Now consider the phase plot, the phase of the classical integrator is $-90°$ over all frequencies. In contrast, the phase of the fractional integral is $-1.5 \times 90°$. Changing the fractional order from 1 to 2 results in a variation of the slope of the magnitude from $-20\ dB/dec$ to $-40\ dB/dec$. Thereby, the constant phase varies between $-90°$ and $-180°$.

To this point, we want to mention that the magnitude and phase characteristics depends on the chosen frequency range, see Figure B.3. A rule of thumb is to use one order for each frequency decade. For example, the frequency range $[10^{-2}\ 10^{2}]\ rad/s$ can be approximated using a forth order function ($N = 4$).

Now, the approximation of the fractional derivative

$$s^{\beta} \approx \prod_{i=1}^{N} \frac{1 + \frac{s}{\omega_i'}}{1 + \frac{s}{\omega_i}}, \qquad \omega_i', \omega_i \quad \in \mathbb{R} \tag{B.42}$$

using the same method is considered. The results are shown in Figure B.4.

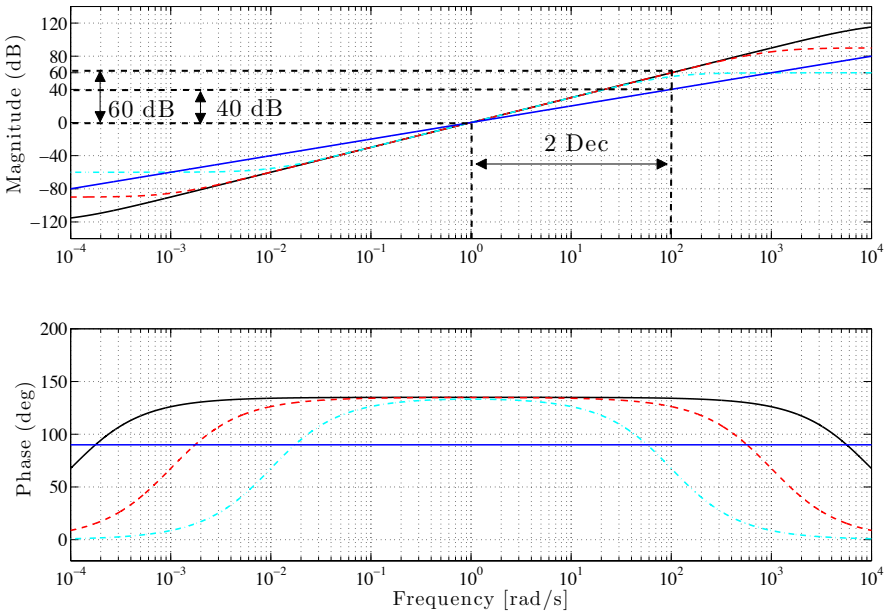

Figure B.4: Bode plot of the CRONE approximation of the fractional integrator $s^{1.5}$ for the frequency ranges, $[10^{-4}\ 10^{4}]\,rad/s$ (black), $[10^{-3}\ 10^{3}]\,rad/s$ (dashed red) and $[10^{-2}\ 10^{2}]\,rad/s$ (dashdot cyan), in comparison bode plot of the integrator $\frac{1}{s}$ (blue)

It is well known that the slope of the magnitude of an ideal differentiator s is 20 dB/dec, see Figure B.4. The magnitude slope of the fractional derivative is given by 20 dB/dec multiplied by the fractional order β, namely 1.5×20 dB/dec. The phase of the classical derivative is 90° over all frequencies. In contrast, the phase of the fractional derivative is $1.5 \times 90°$ in the bandwidth of interest.

To summarize this section, the following conclusion can be made. Depending on the implementation constraint on the system, a frequency range has first be chosen. In this range, the approximation (B.41) provides $-\alpha \times 20$ dB/dec for the slope of the magnitude and $-\alpha \times 90°$ for the phase. For the fractional derivative approximation (B.42), it is $\beta \times 20$ dB/dec for the slope of the magnitude and $\beta \times 90°$ for the phase. The order N has to be chosen accordingly.

B.4 Uncertain Plants

One of the goals of this thesis is the design of robust controllers for uncertain systems. For this purpose, it is necessary to define the type of uncertainty of interest. It consists of uncertain continuous linear time invariant SISO systems with static gain variations. Depending whether the uncertain plant is given in state-space form or as a transfer function, we discuss which conditions have to be fulfilled such that the uncertainty falls into the class of static variations.

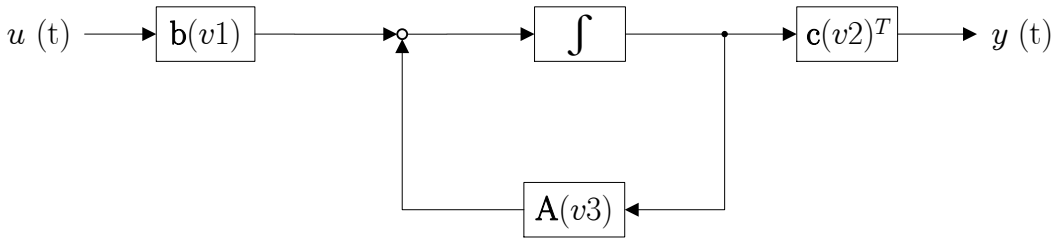

Figure B.5: Linear time invariant system with varying input vector $\mathbf{b}(v1)$, output vector $\mathbf{c}(v2)^T$ and state matrix $\mathbf{A}(v3)$ depending on the parameters $v1$, $v2$ and $v3$

Figure B.5 shows a general representation of parameter uncertain LTI systems. The input vector $\mathbf{b}(v1)$, output vector $\mathbf{c}(v2)$ and the state matrix $\mathbf{A}(v3)$ depends on the parameters $v1$, $v2$ and $v3$, respectively. Assume that these parameters are available in real-time, the structure shown in Figure B.5 describes a parameter-dependent system, see (Apkarian et al. 1995). In this case, designing a gain scheduled controller seems to be more convenient. The controller is also parameter dependent and is scheduled in real-time. The design of gain-scheduled controllers for LPV systems is discussed in detail in (Apkarian et al. 1995) and (Apkarian and Gahinet 1995).

In case that the parameters are not available in real-time and the estimation of the parameters is not possible, the structure shown in Figure B.5 describes a LTI system with uncertain parameters. In this case, the design of a robust controller covering the range of parameter variations should be considered. In case that this range is known, the performance of the controller can be validated through simulation. In this work, we are considering the design of such controllers for this class of LTI systems.

Now consider the following state-space system

$$G : \begin{cases} \mathbf{x}(t) &= \mathbf{A}\,\mathbf{x}(t) + \mathbf{b}(v1)\,u(t) \\ y(t) &= \mathbf{c}(v2)^T\,\mathbf{x}(t). \end{cases} \tag{B.43}$$

with $\mathbf{x}(t) \in \mathbb{R}^{n\times 1}$, $u(t) \in \mathbb{R}$ and $y(t) \in \mathbb{R}$ denote the state vector, the input and output, respectively. Assume that the matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ is parameter independent and the vectors $\mathbf{b}(v1) \in \mathbb{R}^{n\times 1}$ and $\mathbf{c}^T(v2) \in \mathbb{R}^{1\times n}$ depend on the variable parameters $v1$ and $v2$, respectively. Moreover, this dependency is as follows

$$\mathbf{b}(v1) = v1 \cdot \mathbf{b}_0 \;\text{ and }\; \mathbf{c}(v2)^T = v2 \cdot \mathbf{c}_0^T. \tag{B.44}$$

The vectors $\mathbf{b}_0 \in \mathbb{R}^{n\times 1}$ and $\mathbf{c}_0^T \in \mathbb{R}^{1\times n}$ constitute the nominal case with $v1 = 1$ and $v2 = 1$. Moreover, we assume that the parameters $v1$ and $v2$ lie in the intervals $[v1_{min},\ v1_{max}]$ and $[v2_{min},\ v2_{max}]$, respectively.

The representation of the parameter dependent vectors $\mathbf{b}(v1)$ and $\mathbf{c}(v2)$ in the form given by Equation (B.44) restricts the general paramater structure to be static gain uncertainty. To make this clear, consider the parameter dependent transfer function of the system (B.43) as follows

$$G(s) = \mathbf{c}(v2)^T(s \cdot \mathbf{I_n} - \mathbf{A})^{-1}\mathbf{b}(v1). \tag{B.45}$$

Now substituting $(v1 \cdot \mathbf{b_0})$ for $\mathbf{b}(v1)$ and $(v2 \cdot \mathbf{c_0})$ for $\mathbf{c}(v2)$ results in

$$G(s) = \mathbf{c}_0^T \cdot v2(s \cdot \mathbf{I_n} - \mathbf{A})^{-1}\mathbf{b_0} \cdot v1 = \underbrace{v1 \cdot v2}_{V}\ \underbrace{\mathbf{c}_0^T(s \cdot \mathbf{I_n} - \mathbf{A})^{-1}\mathbf{b_0}}_{G_0(s)}. \tag{B.46}$$

The new introduced parameter V lies in the intervall $[V_{min},\ V_{max}]$. $G_0(s)$ is the nominal transfer function of the state-space model (B.43) for the case $V = 1$. The new parameter bounds are deduced from the upper and lower limits of the parameters $v1$ and $v2$ as follows

$$V \in \left[\underbrace{v1_{min} \cdot v2_{min}}_{V_{min}}\ \ \underbrace{v1_{max} \cdot v2_{max}}_{V_{max}}\right]. \tag{B.47}$$

The parameter V consists the static gain of the plant. The uncertainty of the plant consists in variations of this parameter.

It is now straightforward to check if a given uncertain plant belongs to this class of uncertainties. In case that the plant model is provided in state-space form, the input and output vectors should be representable in the form given by Equation (B.44). In case that the plant model is given as a tranfer function, the representation should be in the form given by the Equation (B.46) consisting of a nominal function $G_0(s)$ and a static gain V. To get an insight into the dynamic of such uncertain plants, an example is considered.

Example: Electronic Throttle

Generally, the design of robust controllers to ensure a specified performance in case of parameter variations is always desired and should be considered. This is due to the fact that uncertainties are mostly present. For example, uncertainty in the parameters can be the result of an imperfect identification of the system. Another well known fact is the influence of external factors as temperature or ageing effects in mechanical components.

In this context, the electronic throttle consists an interesting plant to be studied. It is a main component in modern engines. It regulates the amount of the air going inside the engine. For safety reason, it is necessary to provide the desired performance in case of parameter variations. These are the result of ageing effects or temperature changes. The design of robust controllers that guaranty a desired performance in the presence of static gain variations is considered in Section 2, 3 and 4.

In (Alt et al. 2010), the following transfer function

$$G(s) = \frac{F}{s(\tau s + 1)} = \frac{\alpha(s)}{U_a(s)} \tag{B.48}$$

is used to catch the dynamic of a simplified model of the electronic throttle. The input and output of the system are the voltage U_a and the angular position α, respectively. Assume that the time constant τ is fixed and well known. Thereby, the static gain F is uncertain. The parameter dependent transfer function, which describes the dynamic of the electronic throttle is given by

$$G(s) = V \underbrace{\frac{F_0}{s(\tau s + 1)}}_{G_0(s)} \text{ with } V \in [V_{min}\, V_{max}] \tag{B.49}$$

with $V_{min} = 0.5$, $V_{max} = 1.5$ and the nominal value $V_{nom} = 1$. F_0 consists the nominal static gain. A robust controller should cope with the variations of the parameter V and provides the desired performance in the whole parameter range.

To understand the impact of parameter variations on the dynamic of the electronic throttle, the frequency response of the system (B.49) is considered. Figure B.6 shows the magnitude and phase of the Bode plot. The variations of V cause a shift in the magnitude response. The crossover frequency varies between ω_{Cmin} and ω_{Cmax}. The parameter variations have no influence on the bode phase.

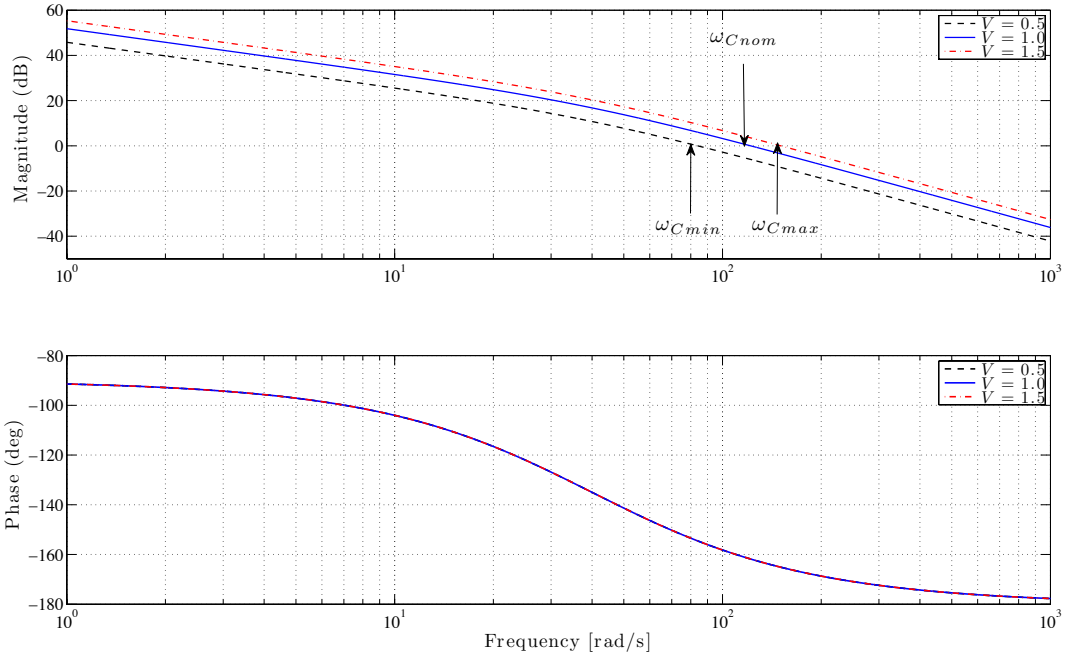

Figure B.6: Bode plot of the uncertain transfer function of the electronic throttle for the minimal, nominal and maximal values $V = 0.5$ (dashed black), $V = 1$ (blue) and $V = 1.5$ (dashdot red), respectively

B.5 Robust Control

In the previous section, the class of plants being considered in this work was introduced. In this context, it is now appropriate to discuss the robustness specifications. Specifically, which types of robustness are taken into consideration. Therefore, a brief introduction is given here. Generally, robustness has been extensively discussed in several books, see for example (Maciejowski 1989) or (Boyd et al. 1991).

In (Skogestad and Postlethwaite 2007), the classification of robustness into stability robustness and performance robustness is considered. The first type of robustness is reserved to problems in which stability has to be guaranteed in case of uncertainties. In this case, the main goal of the control problem is to find a controller that stabilizes the uncertain system. Additionally to robust stability, it is often required that the controller achieves a predefined performance for parameter variations. In this case, the controller provides robust performance. The closed-loop structure shown in Figure B.7 is used to define this performance.
Thereby, $K(s)$ is a given LTI controller. $G(s)$ is an uncertain plant with the uncertain parameter V and the nominal transfer function $G_0(s)$. The input and output of the system are denoted by U and Y, respectively. The transfer function from the reference R to the

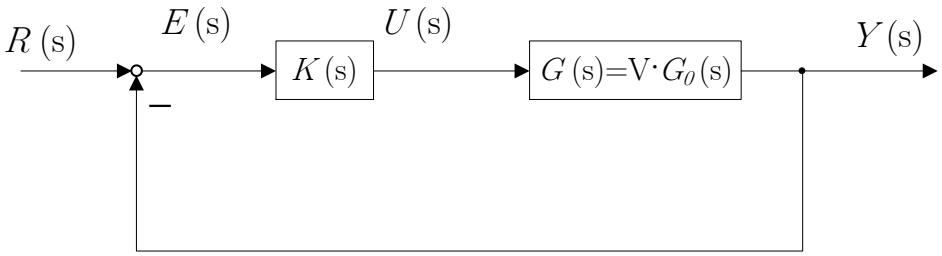

Figure B.7: Closed-loop structure for an uncertain plant with static gain V variations and a controller $K(s)$

output Y is given by

$$T(s) = \frac{V \cdot G_0(s)K(s)}{1 + V \cdot G_0(s)K(s)}. \tag{B.50}$$

The function $T(s)$ defines the frequency response depending on the gain V. It can be used to specify the performance in terms of the static gain V. For this purpose, one can impose constraints on its frequency response as well as on its time response. This fact is shown in Figure B.8. The time domain specifications are given in terms of the closed-loop step

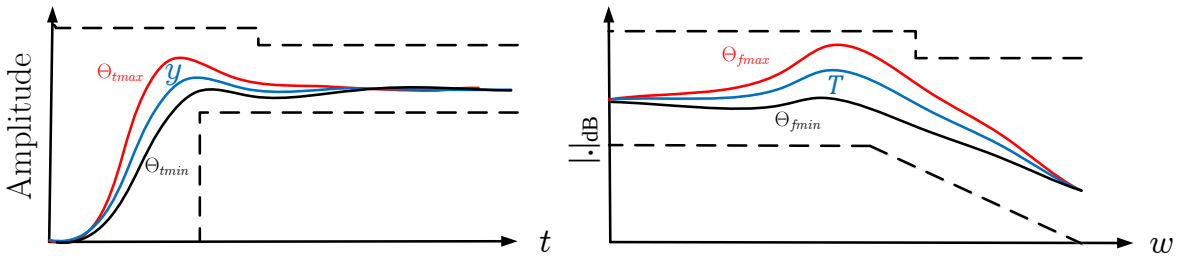

Figure B.8: Performance specification in case of parameter variation, time domain specifications (left) with an upper and lower time performance bounds, frequency domain specifications (right) with an upper and lower frequency performance bounds for minimal (dark blue line), nominal (blue line) and maximal (red line) parameter variation

response $y(t)$ for the minimal V_{min}, nominal V_{nom} and maximal value V_{max}. The frequency domain specifications are given in terms of the Bode magnitude of the closed-loop transfer

function $T(j\omega)$. Formally, this is given by the following

$$|\Theta_{fmin}(j\omega)| < |T(j\omega)| < |\Theta_{fmax}(j\omega)| \qquad \forall\omega, \forall V. \tag{B.51}$$

Thereby, $\Theta_{fmin}(j\omega)$ and $\Theta_{fmax}(j\omega)$ define the lower and upper frequency bounds on the function $T(j\omega)$. Generally, these are expressed in terms of linear high- or low-pass filters. Depending on the approach used to design robust controllers, time domain specifications can be used instead of frequency constraints. In this case, the step response $y(t)$ is constrained as follows

$$\Theta_{tmin}(t) < y(t) < \Theta_{tmax}(t) \qquad \forall t, \forall V \tag{B.52}$$

with $\Theta_{tmin}(t)$ and $\Theta_{tmax}(t)$ are the lower and upper time constraints.

Additionally to the bounds on the complementary sensitivity function T, one can specify the desired response for the open-loop function $L = GK$. Moreover, using the loop-shaping technique and an appropriate choice of the related upper and lower bounds one can even enforce the nominal open-loop function $L_0 = G_0K$ to have a well defined frequency response. The whole method is presented and discussed in details in Chapter 2.2.

C Implementation of Fractional Orders

The number of formulas dedicated to the implementation of fractional orders are considerably high. In (Valério and da Costa 2005), several methods to implement fractional orders are presented and discussed. Moreover, they can be categorized into discrete-time and continuous time formulas.

Formulas for Discrete-Time Implementation

One of the well known discrete-time formulas for fractional orders as well as for integer orders is given by backward finite difference. For example, consider the fractional derivative operator D^v, the approximation using backward differentiation is given by

$$D^v \approx \left(\frac{1-z^{-1}}{T}\right)^v \tag{C.1}$$

with z denotes the shift operator and T the sampling time. Using the MacLaurin series expansion, the following approximation (Valério and da Costa 2005, Equation 24)

$$D^v \approx \frac{1}{T}\sum_{i=0}^{n}(-1)^i \frac{\Gamma(v+1)}{\Gamma(i+1)\Gamma(v-i+1)} z^{-i} \tag{C.2}$$

is obtained. For the seek of comparison, higher order approximations are considered. The Tustin approximation is given by

$$D^v \approx \left(\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}\right)^v \approx \left(\frac{2}{T}\right)^v \Gamma(v+1)\Gamma(-v+1)\sum_{k=0}^{n} z^{-k} \tag{C.3}$$

$$\times \sum_{j=0}^{k} \frac{(-1)^j}{\Gamma(v-j+1)\Gamma(j+1)\Gamma(k-j+1)\Gamma(-v+j-k+1)} \tag{C.4}$$

and Simpson approximation is given by

$$\begin{aligned} D^v \approx & \left(\frac{3}{T}\frac{(1+z^{-1})(1-z^{-1})}{1+4z^{-1}+z^{-2}}\right)^v \approx \left(\frac{3}{T}\right)^v \Gamma(v+1)\Gamma(-v+1)^2 \\ & \times \sum_{q=0}^{n} z^{-q} \sum_{n=0}^{q/2} \sum_{p=2n}^{q} \frac{(-1)^n}{\Gamma(n+1)\Gamma(v-n+1)} \\ & \times \frac{(2-\sqrt{3})^{-v-p+2n}(2+\sqrt{3})^{-v-q+p}}{\Gamma(p-2n+1)\Gamma(-v-p+2n+1)\Gamma(p-q+1)\Gamma(-v-q+p+1)} \end{aligned} \tag{C.5}$$

see (Valério and da Costa 2005, Equation 27-28). Additionally to this type of approximations, the impulse and step response to the fractional operator can be used to provide other formulas.

The filter used to approximate the time response is given by $F(z) = \sum_{k=0}^{+\infty} a_i z^{-i}$. For the

case of the impulse response, the goal is to approximate this response, which is given by

$$\frac{t^{-v-1}}{\Gamma(-v)} \text{ for } t > 0 \tag{C.6}$$

using the impulse response of the filter $F(z)$ given by $y(kT) = a_k$ with $k \in \mathbb{N}$. The discrete-time approximation is given by (Valério and da Costa 2005, Equation 33)

$$D^v \approx \frac{T^{-v}}{\Gamma(-v+1)} - \frac{T^{-v-1}}{\Gamma(-v)} + \sum_{n}^{i=1} \frac{(iT)^{-v-1}}{\Gamma(-v)} z^{-i} \tag{C.7}$$

Now using the step response, another formula can be obtained. For this reason, the analytical formula is first recalled as follows

$$\frac{t^{-v}}{\Gamma(-v+1)} \text{ for } t > 0 \tag{C.8}$$

and (Valério and da Costa 2005, Equation 35)

$$D^v = \sum_{i=0}^{n} a_i z^{-i} \text{ with } a_0 = \frac{T^{-v}}{\Gamma(-v+1)} - \frac{T^{-v-1}}{\Gamma(-v)} \text{ and} \tag{C.9}$$

$$a_k = -\sum_{k-1}^{i=0} a_i + \frac{(kT)^{-v}}{\Gamma(-v+1)}, \quad k = 1, 2, \cdots n \tag{C.10}$$

The idea of using series expansion to approximate fractional derivatives in the discrete-time domain can be also applied in the continuous time domain. One of the well known continuous time formula is given by (Melchior et al. 2002). This is also the method adopted in this work, which has been introduced in Section using the expression (B.38), (B.39) and (B.40).

Table C.1: Laplace and Inverse Laplace Transforms

$\frac{1}{\sqrt{s}}$	$\frac{1}{\sqrt{\pi t}}$
$\frac{1}{s\sqrt{s}}$	$2\frac{t}{\sqrt{\pi}}$
$\frac{1}{s^n\sqrt{s}}$	$\frac{2^n t^{n-(1/2)}}{1\cdot 3\cdot 5\cdots(2n-1)\sqrt{(\pi)}}$
$\frac{s}{(s-a)^{\frac{3}{2}}}$	$\frac{1}{\sqrt{\pi t}}e^{at}(1+2at)$
$\sqrt{s-a}\sqrt{s-b}$	$\frac{1}{2\sqrt{\pi t^3}e^{bt}-e^{at}}$
$\frac{1}{\sqrt{(s)}+a}$	$\frac{1}{\sqrt{\pi t}} - ae^{a^2t}\text{erfc}(a\sqrt{t})$
$\frac{\sqrt{s}}{s-a^2}$	$\frac{1}{\sqrt{\pi t}} + ae^{a^2t}\text{erfc}(a\sqrt{t})$
$\frac{1}{\sqrt{s}(s-a^2)}$	$\frac{1}{a}e^{a^2t}\text{erfc}(a\sqrt{t})$
$\frac{1}{\sqrt{s}(s+a^2)}$	$\frac{2}{a\sqrt{\pi}}\int_0^{a\sqrt{t}} e^{\tau^2}d\tau$
$\frac{\sqrt{s}}{s+a^2}$	$\frac{1}{\sqrt{\pi t}} - \frac{2a}{\sqrt{\pi}}e^{-a^2t}\int_0^{2\sqrt{t}} e^{\tau^2}d\tau$
$\frac{b^2-a^2}{(s-a^2)(\sqrt{(s)}+b)}$	$e^{a^2t}\left[b-a\left(\text{erfc}(a\sqrt{t})\right)\right] - be^{b^2t}\text{erfc}\left(b\sqrt{t}\right)$
$\frac{1}{\sqrt{s}(\sqrt{s}+a)}$	$e^{a^2t}\text{erfc}(a\sqrt{t})$
$\frac{1}{\sqrt{s+b}(s+a)}$	$\frac{1}{\sqrt{b-a}}e^{-at}\text{erfc}(\sqrt{b-a}\sqrt{t})$
$\frac{b^2-a^2}{\sqrt{s}(s-a^2)(\sqrt{s}+b)}$	$e^{a^2t}\left[\frac{b}{a}erf(a\sqrt{t}-1) + e^{b^2t}erfc(b\sqrt{t})\right]$
$\frac{(1-s)^n}{s^{n+0.5}}$	$\frac{n!}{(2n)!\sqrt{\pi t}}H_{2n}(\sqrt{t}),\ H_n(x) = e^{x^2}\frac{d^n}{dx^n}\left(e^{-x^2}\right)$
$\frac{(1-s)^n}{s^{n+1.5}}$	$-\frac{n!}{(2n+1)!\sqrt{\pi}}H_{2n+1}(\sqrt{t})$
$\frac{\sqrt{s+2a}-\sqrt{s}}{\sqrt{s}}$	$ae^{-at}\left[I_1(at)+I_0(at)\right]$, where $I_n(x) = j^{-n}J_n(jt)$, J_n is the Bessel function
$\frac{1}{\sqrt{s+a}\sqrt{s+b}}$	$e^{-0.5(a+b)t}I_0\left(\frac{a-b}{2}\right)$
$\frac{\Gamma(k)}{(s+a)^k(s+b)^k},\ k\geq 0$	$\sqrt{\pi}\left(\frac{t}{a-b}\right)^{k-0.5}e^{-0.5(a+b)t}I_{k-0.5}\left(\frac{a-b}{2}t\right)$
$\frac{\Gamma(k)}{(s+a)^{0.5}(s+b)^{1.5}}$	$te^{-0.5(a+b)t}\left[I_0\left(\frac{a-b}{2}\right)+I_1\left(\frac{a-b}{2}\right)\right]$
$\frac{\sqrt{s+2a}-\sqrt{s}}{\sqrt{s+2a}+\sqrt{s}}$	$\frac{1}{t}e^{-at}I_1(at)$
$\frac{(a-b)^k}{(\sqrt{s+a}+\sqrt{s+b})^{2k}},\ k>0$	$\frac{k}{t}e^{-0.5(a+b)t}I_k\left(\frac{a-b}{2}t\right)$
$\frac{1}{\sqrt{s}\sqrt{s+a}(\sqrt{s+a}+\sqrt{s})^{2v}},\ k>0$	$\frac{1}{a^v}e^{-0.5at}I_v\left(\frac{a}{2}t\right)$
$\frac{1}{\sqrt{s^2+a^2}}$	$J_0(at)$
$\frac{1}{\sqrt{s^2-a^2}}$	$I_0(at)$
$\frac{(\sqrt{s^2+a^2}-s)^v}{\sqrt{s^2+a^2}},\ v>-1$	$a^vJ_v(at)$
$\frac{1}{\sqrt{s^2+a^2}^k},\ k>0$	$\frac{\sqrt{\pi}}{\Gamma(k)}\left(\frac{t}{2a}\right)^{k-0.5}J_{k-0.5}(at)$
$(\sqrt{s^2+a^2}-s)^k,\ k>0$	$\frac{ka^k}{t}J_k(at)$
$\frac{(\sqrt{s^2-a^2}+s)^v}{\sqrt{s^2+a^2}},\ v>-1$	$a^vI_v(at)$
$\frac{1}{(s^2-a^2)^k},\ k>0$	$\frac{\sqrt{\pi}}{\Gamma(k)}\left(\frac{t}{2a}^{k-0.5}\right)I_{k-0.5}(at)$
$\frac{1}{s\sqrt{s+1}}$	$\text{erfc}(\sqrt{t})$

$\frac{1}{s+\sqrt{s^2+a^2}}$	$\frac{J_1(at)}{at}$
$\frac{1}{(s+\sqrt{s^2+a^2})^N}$ with $N \in \mathbb{N}$	$\frac{NJ_N(at)}{d^N t}$
$\frac{1}{\sqrt{s^2+a^2}(s+\sqrt{s^2+a^2})}$	$\frac{J_1(at)}{a}$
$\frac{1}{\sqrt{s^2+a^2}(s+\sqrt{s^2+a^2})^N}$	$\frac{J_N(at)}{a^N}$
$\frac{k}{s^2+k^2}\coth\left(\frac{\pi s}{2k}\right)$	$\lvert\sin(kt)\rvert$
$\frac{1}{s}e^{-\frac{k}{s}}$	$J_0(2\sqrt{kt})$
$\frac{1}{\sqrt{s}}e^{-\frac{k}{s}}$	$\frac{1}{\sqrt{\pi t}}\cos(2\sqrt{kt})$
$\frac{1}{\sqrt{s}}e^{\frac{k}{s}}$	$\frac{1}{\sqrt{\pi t}}\cosh(2\sqrt{kt})$
$\frac{1}{s\sqrt{s}}e^{-\frac{k}{s}}$	$\frac{1}{\sqrt{\pi t}}\sin(2\sqrt{kt})$
$\frac{1}{s\sqrt{s}}e^{\frac{k}{s}}$	$\frac{1}{\sqrt{\pi t}}\sinh(2\sqrt{kt})$
$\frac{1}{s^v}e^{-\frac{k}{s}}$, $v>0$	$(\frac{t}{k})^{\frac{v-1}{2}}J_{v-1}(2\sqrt{kt})$
$\frac{1}{s^v}e^{\frac{k}{s}}$, $v>0$	$(\frac{t}{k})^{\frac{v-1}{2}}I_{v-1}(2\sqrt{kt})$
$e^{-k\sqrt{s}}$, $k>0$	$\frac{k}{2\sqrt{\pi t^3}}e^{-\frac{k^2}{4t}}$
$\frac{1}{s}e^{-k\sqrt{s}}$, $k\geq 0$	$\operatorname{erfc}\left(\frac{k}{2\sqrt{t}}\right)$
$\frac{1}{\sqrt{s}}e^{-k\sqrt{s}}$, $k\geq 0$	$\frac{1}{\sqrt{\pi t}}e^{-\frac{k^2}{4t}}$
$\frac{1}{s\sqrt{s}}e^{-k\sqrt{s}}$, $k\geq 0$	$2\frac{1}{\sqrt{\pi t}}e^{-\frac{k^2}{4t}} - k\operatorname{erfc}\left(\frac{k}{2\sqrt{t}}\right)$
$\frac{ae^{-k\sqrt{s}}}{s(a+\sqrt{s})}$ $k\geq 0$	$-e^{ak}e^{a^2t}\operatorname{erfc}(a\sqrt{(t)}+\frac{k}{2\sqrt{t}}) + \operatorname{erfc}\left(\frac{k}{2\sqrt{t}}\right)$
$\frac{e^{-k\sqrt{s}}}{\sqrt{s}(a+\sqrt{s})}$	$e^{ak}e^{a^2t}\operatorname{erfc}\left(a\sqrt{(t)}+\frac{k}{2\sqrt{t}}\right)$
$\ln\left(\frac{s-a}{s-b}\right)$	$\frac{1}{t}(e^{bt}-e^{at})$
$\ln\left(\frac{s^2+a^2}{s^2}\right)$	$\frac{2}{t}(1-\cos(at))$
$\ln\left(\frac{s^2-a^2}{s^2}\right)$	$\frac{2}{t}(1-\cosh(at))$
$\arctan\left(\frac{k}{s}\right)$	$\frac{1}{t}\sin(kt)$
$\frac{1}{s^\alpha}$	$\frac{t^{\alpha-1}}{\Gamma(\alpha)}$

D Training Artificial Neural Networks

The aim of this section is to provide a basic overview about the training of neural networks. Basically, it consists of adapting the weights of the specified network such that a defined cost function is minimized. This task is performed iteratively using well known optimization algorithms. First of all, one has to specify the network structure. It is shown in Figure D.1. It defines a recurrent neural network. It has r inputs φ_1 to φ_r, two layers

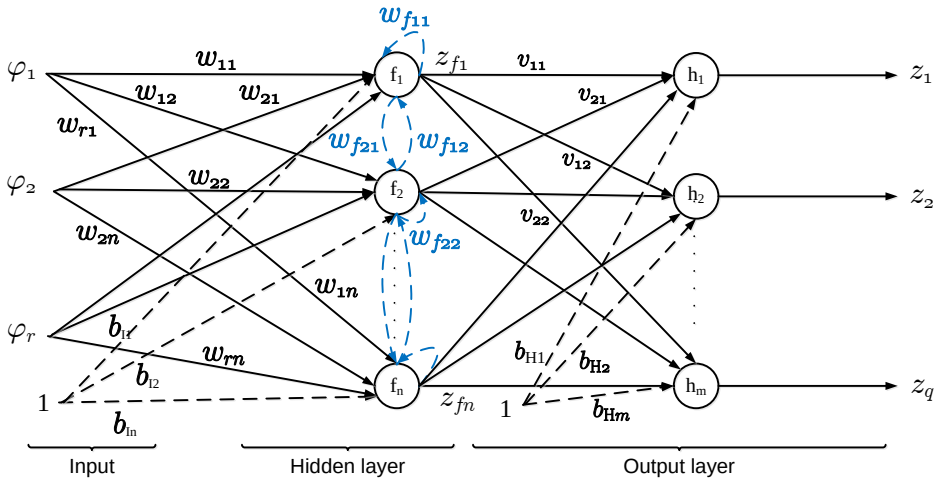

Figure D.1: Multilayer neural network (three layers: input, output and hidden layer)

and q outputs z_1 to z_q. The output of the above recurrent network is given by

$$z_i(k) = h_i\left(\sum_{l=1}^{n} z_{fl}(k)v_{li} + b_{Hi}\right) \tag{D.1}$$

with h_i denotes the output activation functions. b_{Hi} and $v_{l,i}$ consist the output bias and weights, respectively. Thereby, the internal states of the hidden layer z_{fi} can be computed as follows

$$z_{fi}(k+1) = f_i\left(\sum_{l=1}^{r} \varphi_l(k)w_{li} + \sum_{j=1}^{n} z_{fi}(k)w_{fij} + b_{Ii}\right). \tag{D.2}$$

Whereas f_i denote the activation of the hidden layer. Assume that the first network output z_1 has to be fit to a desired response given by y_d. Therefore, a cost function has to be specified. It is given in terms of the error signal $e(k, \mathbf{p}) = y_d(k) - z_1(k, \mathbf{p})$. The vector $\mathbf{p}$ denotes all the network parameters given by the network bias and weights. In the field of NN, it common to use one of three objective functions, $mse(\cdot)$, $sse(\cdot)$ or $mae(\cdot)$. The $mse(\cdot)$ function stands for mean square error and is defined as follows

$$mse(e) = \sum_{k=1}^{N} \frac{(y_d(k) - z_1(k,\mathbf{p}))(y_d(k) - z_1(k,\mathbf{p}))}{N} = \sum_{k=1}^{N} \frac{e(k,\mathbf{p})e(k,\mathbf{p})}{N} \quad \text{(D.3)}$$

or in vector form

$$mse(\mathbf{e}) = \frac{\mathbf{e}^T(\mathbf{p})\mathbf{e}(\mathbf{p})}{N} \quad \text{(D.4)}$$

for a given error vector $\mathbf{e} = \begin{bmatrix} e(1,\mathbf{p}) & e(2,\mathbf{p}) & \cdots & e(N,\mathbf{p}) \end{bmatrix} \in \mathbb{R}^N$. The error is a function in the parameter $\mathbf{p}$. In case that the error depends on the parameter $\mathbf{p}$ linearly, minimizing the cost function $mse(\mathbf{e})$ describes a quadratic convex problem. In this case, any gradient based method can be used. Due to the convexity, any local minimum is the global minimum.

In some cases, it is preferable to use the sum of square error $sse(e)$ given as follows

$$sse(\mathbf{e}) = \sum_{k=1}^{N} e(k,\mathbf{p})e(k,\mathbf{p}) \quad \text{(D.5)}$$

instead of $mse(\mathbf{e})$. In case that the size of the vector $\mathbf{e}$ is not high, one can use the function (D.5). Otherwise, the function $mse(\cdot)$, which is normalized with N, is numerically well conditioned. For high dimensions, its value does not become very high.

The third function is the mean absolute error mae). It is defined as follows

$$mae(\mathbf{e}) = \sum_{k=1}^{N} |\mathbf{e}(k,\mathbf{p})|. \quad \text{(D.6)}$$

In our case the mse function (D.3) is used to train the recurrent neural network. This is due to its quadratic normalized form. After defining the objective function, the used training algorithm is briefly discussed.

Levenberg Marquadt

There exists a huge number of algorithms to solve optimization problems, see (Boyd and Vandenberghe 2004), (Werner 2004) and (Mark et al. 2011). Generally, it differs in the literature between two main approaches, gradient free and gradient based methods. The first method includes all algorithms that do not require the gradient of the objective function as Evolutionary Method or Particle Swarm. They are sometimes called stochastic

search methods. The second method consists of gradient based methods, which need the gradient of the objective function to perform a search for the optimal value. The computation of the gradient can be performed either analytically or numerically.

In the field of the NN training it is common to use gradient based methods. The widely used methods are Steepest Descent, Newton or quasi-Newton methods. In this section a brief summary of these methods is given.
Consider now the objective function

$$f(\mathbf{p}) = \frac{\mathbf{e}(\mathbf{p})^T \mathbf{e}(\mathbf{p})}{N} \tag{D.7}$$

with the parameter vector $\mathbf{p} = \begin{bmatrix} p_1 & p_2 & \cdots p_n \end{bmatrix}$. The optimal vector that minimizes (D.7) is denoted by $\mathbf{p}^\star$ and its optimal value is $f^\star = f(\mathbf{p}^\star)$. An iterative procedure to compute $\mathbf{p}^\star$ is given by

$$\mathbf{p}_{k+1} = \mathbf{p}_k + \alpha d \tag{D.8}$$

where $\mathbf{p}_k$ denotes the current approximated solution to $\mathbf{p}^\star$ and $\mathbf{p}_{k+1}$ the improved solution. Here d and α denote the search direction and the learning rate, respectively, see (Boyd and Vandenberghe 2004) or (Wilamowski and Irwin 2011)[chap. 12]. Depending on the choice of d and α equation (D.8) defines either the Steepest Descent or Newton search algorithm. (Gavin 2011)

Before proceeding to the definition of both methods, the gradient $\mathbf{g} = \nabla f$ and the Hessian $\mathbf{H} = \nabla^2 f$ of the objective function is recalled as follows

$$\nabla f(\mathbf{p}) = \begin{bmatrix} \frac{\partial f}{\partial p_1} \\ \frac{\partial f}{\partial p_2} \\ \vdots \\ \frac{\partial f}{\partial p_n} \end{bmatrix} \qquad \text{and} \qquad \nabla^2 f(\mathbf{p}) = \begin{bmatrix} \frac{\partial^2 f}{\partial p_1 \partial p_1} & \frac{\partial^2 f}{\partial p_1 \partial p_2} & \cdots & \frac{\partial^2 f}{\partial p_1 \partial p_n} \\ \frac{\partial^2 f}{\partial p_2 \partial p_1} & \frac{\partial^2 f}{\partial p_2 \partial p_2} & \cdots & \frac{\partial^2 f}{\partial p_2 \partial p_n} \\ \frac{\partial^2 f}{\partial p_3 \partial p_1} & \frac{\partial^2 f}{\partial p_3 \partial p_2} & \cdots & \frac{\partial^2 f}{\partial p_3 \partial p_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial p_n \partial p_1} & \frac{\partial^2 f}{\partial p_n \partial p_2} & \cdots & \frac{\partial^2 f}{\partial p_n \partial p_n} \end{bmatrix}. \tag{D.9}$$

Starting with an initial value $\mathbf{p}_0$, an update using the Steepest Descent method is defined as follows

$$\mathbf{p}_{k+1} = \mathbf{p}_k - \alpha \mathbf{g}(\mathbf{p}). \tag{D.10}$$

Depending on the choice of the rate α it exists different classes of the Steepest Descent.

Steepest Descent with fixed step α_0 consists in choosing a fixed value of the learning rate for all iterations. Another variety is to update the rate α_k during each iteration. In both cases only first order information (gradient) is used. In case that information about the Hessian exists, second order information methods can be used. The Newton method is given by

$$\mathbf{p}_{k+1} = \mathbf{p}_k - \mathbf{H}^{-1}(\mathbf{p})\mathbf{g}(\mathbf{p}) \tag{D.11}$$

with the Hessian matrix $\mathbf{H} = \nabla^2 f(\mathbf{p})$. In some cases it is not possible to compute the Hessian or it is computationally expensive. In these cases an approximation of the Hessian is used instead of the exact matrix. It is well known that far from the optimal value $\mathbf{p}^*$ the Steepest Descent methods perform well. On the other side, Newton methods perform close to $\mathbf{p}^*$ better. The following update law

$$\mathbf{p}_{k+1} = \mathbf{p}_k - (\mathbf{H}(\mathbf{p}) + \mu I)^{-1}\mathbf{g}(\mathbf{p}) \tag{D.12}$$

combine both methods. To make this clear, consider a very small value of $\mu \approx 0$, then equation (D.12) becomes (D.11). In the case of a large value of μ it becomes the Steepest Descent method. The use of the parameter μ allows a combination of both methods. Using μ it is possible to switch between the two algorithms. The resulting algorithm is called Levenberg Marquadt algorithm and is used in this work to train the closed-loop recurrent neural network.