La collana della Scuola di Alta Formazione Dottorale accoglie le migliori tesi di dottorato dell'Università degli Studi di Bergamo, insignite della dignità di stampa e sottoposte a procedura di *blind peer review*.

Model uncertainty is not always taken into account in the engineering fields, notwithstanding it plays a key role in practical applications. In the beginning, the uncertainty information was selected by the engineer. Instead, in the last decades, the system identification community has searched for different ways to model this information. This study proposes an automatic procedure to solve this problem by employing the innovative kernel-based system identification. The resulting data-driven uncertainty model is then applied to the robust control design and to robust model-based fault diagnosis problem.

Nicholas Valceschini received a PhD in Engineering and Applied Sciences at the Control Systems and Automation Laboratory (35th Cycle) at the University of Bergamo. Since June 2024 he is an Assistant Professor at the University of Bergamo. His research activity is mainly focused on data-driven robust control based on non-parametric models and predictive maintenance of system components manufacturing.

ISBN: 978-88-97413-97-4
DOI: 10.13122/978-88-97413-97-4

N. Valceschini DATA-DRIVEN ROBUST CONTROL AND DIAGNOSIS

72

Collana della Scuola di Alta Formazione Dottorale
- 72 -

Nicholas Valceschini

DATA-DRIVEN ROBUST CONTROL AND DIAGNOSIS
Theory and Application

UNIVERSITÀ DEGLI STUDI DI BERGAMO

2024

Collana della Scuola di Alta Formazione Dottorale

Diretta da Paolo Cesaretti

Ogni volume è sottoposto a *blind peer review*.

ISSN: 2611-9927

Sito web: https://aisberg.unibg.it/handle/10446/130100

Nicholas Valceschini

DATA-DRIVEN ROBUST CONTROL AND DIAGNOSIS
Theory and Application

Università degli Studi di Bergamo

2024

Data-driven robust control and diagnosis : Theory and Application
/ Nicholas Valceschini. – Bergamo :
Università degli Studi di Bergamo, 2024.
(Collana della Scuola di Alta Formazione Dottorale; 72)

ISBN: 978-88-97413-97-4
DOI: 10.13122/978-88-97413-97-4

Progetto grafico: Servizi Editoriali – Università degli Studi di Bergamo
© 2018 Università degli Studi di Bergamo
via Salvecchio, 19
24129 Bergamo
Cod. Fiscale 80004350163
P. IVA 01612800167

https://aisberg.unibg.it/handle/10446/283409

Acknowledgments

I would like to thank all the people of the Control and Automation research group, at the University of Bergamo. In particular:

professor Mirko Mazzoleni for his patience and availability, who leads me every day to give my best, even in difficult moments;

professor Fabio Previdi, who has made me a better researcher and worker with his precious knowledge;

professor Antonio Ferramosca, who shared to me his experiences and professionalism;

professor Simone Formentin, who gave me the right insight at the right time;

Last but not least, I'd like to thank my girlfriend Sabrina and my parents for always supporting and helping me to become a better person.

Table of Contents

Introduction

Since the birth of humankind, the desire to understand the natural phenomena around us has played a key role in our existence. In the beginning, the representation of nature was entrusted to artists through their paintings or sculptures. Then, with the advent of the scientific method, the artist was replaced by the scientist. For centuries, the latter drew natural phenomena with mathematical relations, called *models*, validated by certain experimental tests. This method represents reality in an understandable way: however, it is applicable only on simple systems and it requires a great effort. This approach is called *white-box modeling* (or *first principle approach*). Nowadays, thanks to computer, it is still widely used. Neverthless, scientific research has investigated an alternative way, due to the limited real-world applications. Leveraging the huge amount of collectable data and exploiting the statistical literature, the researchers have defined the innovative *black-box modeling*. This method develops a model through some algorithms that build a representation of the real system, starting from the examination of a dataset acquired on the real plant. Doing so, no prior knowledge is required, but the resulting model has lost any physical interpretation. A third method, called *grey-box modeling* relies on a model, made with the first principle approach, coupled with a black-box modeling that identifies the parameters. This method funnel the main pros of the other two branches, but it is slower than the black-box approach.

A *system* is defined as a process that transforms inputs into outputs. The systems can be *static* or *dynamic*. The first represent those systems where the input and output have no temporal relationships. Systems where the output is explained by the input and the state of the system, belong to the second type. Dynamic systems are the most spread in the modeling of natural phenomena. The literature that studies the modeling of dynamic systems takes the name of *system identification* [1, 2].

Black-box modeling is useful and widely used, but it conceals a problem that is usually overlooked. This consists in the uncertainty quantification of the resulting model. Often, the acquired data are the results of some experiments performed on the true system, where the input signal is a priori chosen and the output is sensed by a sensor. This setting produces a noisy dataset, due to the nature of the sensors and actuators. The uncertainty due to this source is called *variance*. This is not the only source of the model uncertainty. Often, the model has a structural mismatch with respect to the real system or even is endowed with unpredictable events. The latter cannot be assessed during the model identification procedure, but the former, called *bias*, depends directly on the identification algorithms.

The most widespread system identification framework is that of *Prediction Error Methods* [2, 3]. It assumes to identify a parameters vector (and its variance) of a selected model class and model complexity. These choices are usually done by the user and therefore the wrong decisions causes bias. Instead, the *Set Membership identification* [4, 5], in addition to the model structure and complexity choices, assumes a unknown but bounded output noise. It is useful to quantify the variance, but the bias term (if present) cannot be avoided. However, not all identification algorithms deals with the bias criticality. For instance, the state-of-the-art method, called *kernel-based identification* [6, 7, 8], identifies a *non-parametric model* that has, by definition, low-bias. Due to the Bayesian interpretation of this methodology, the variance is assessed since the results are given as a model distribution.

The branch of identification literature that evaluates both bias and variance during system identification is called *Robust identification*. Three main approaches are: Stochastic Embedding (SE)[9], Model Error Modeling (MEM)[10] and a variation of Set Membership [11]. Nevertheless, this field is also dependent with respect to the user's choices. Furthermore, there are some methods that aim to reduce the bias and/or variance, for instance Akaike Information Criterion [12], Bayesian Information Criterion[13], Cross Validation [14][15] and regularization methods, but these do not avoid the uncertainty problem.

The uncertainty quantification is crucial for real-world applications, since the difference between the model and the real plant can often cause problems. For instance, the model may be used for control purposes: in this situation, a mismatch between model and plant can cause unexpected actions or even system instability. Therefore, by leveraging the (bounded) uncertainty information, we have the assurance that the resulting controller performs ever as we expect. This problem is called *robust control synthesis* [16, 17, 18, 19]. Another use of the uncertainty information consists in producing a *robust residual generator* [20, 21, 22, 23], which estimates the state of a machinery under analysis. This application is widely spread with the advent of industry 4.0. The model mismatch causes false alarms, therefore the uncertainty assessment guarantees a reduction of these. This is helpful to reduce the machine stops and so to minimize the loss of money.

Main contributions

All system identification techniques have their own depiction of uncertainty. These representations differ from the uncertainty depiction employed in robust control design or robust fault detection. Therefore, the first contribution of this study aims to bridge this representation gap in an automatic way, i.e. design a controller or residual generator without engaging the user in difficult structural choices that translate into poor results when the wrong choices are selected. In particular, the entire

proposed procedure is *data-driven*. It is fundamental since, often in robust control and robust fault diagnosis literature, the uncertainty is considered a priori known, derived from the user's knowledge. To quantify the uncertainty, manual selection is often carried out using a trial and error procedure. This can cause: an underestimation that produces some unwanted events (such as: system instability or false alarm detection) or even an overestimation which is less catastrophic than the other case. The latter leads to a performance reduction, compared to the real capabilities of the plant. Instead, with a data-driven method, the uncertainty quantification is estimated directly from the available dataset, therefore it avoids underestimation and overestimation phenomena. Furthermore, it is easier to apply since no iterative procedure is needed. The proposed method exploits the *kernel based identification*, which allows estimating a low-bias model without choosing the model class or complexity. Doing so, the uncertainty is (mostly) only due to variance. Thanks to this characteristic, our method solves the problem of Robust identification in a simpler and automatic way. This is valid because, in Robust identification literature, the uncertainty is represented by a parametric model. So, this procedure needs the selection of the model family of the uncertainty model. We have developed an algorithm that translates the uncertainty information into an understandable representation from the robust control and fault detection point of view. The proposed method, for control design aims, employs the *S/T mixed-sensitivity loop-shaping* problem to design a controller that guarantees some robustness and performance requirements [16]. The proposed technique is developed for Single input Single output for nominal performance, robust stability and robust performance aims [24, 25, 26]. Furthermore, we have extended our method to multi-model systems, by considering it as a single uncertain system. The proposed algorithm deals with respect to multiple sources of uncertainty, i.e. the parameters uncertainty and the model identification uncertainty. Therefore, the resulting closed-loop system is stable and meets the performance requirements for all systems that belongs to the general uncertain system.

The second main theoretical contribution consists of using the same uncertainty information, adopted in the proposed data-driven robust control design, to produce a model-based fault detection that is robust with respect to the model uncertainty. In particular, we solve the *Approximate Fault Detection Problem* to design the *residual generator* [27, 28]. When the uncertainty cannot be completely avoided through the robust residual generator, the fault is diagnosed through a threshold. This operation is not always straightforward, since an incorrect threshold selection can cause wrong fault detection, therefore we propose also a simple threshold selection technique that minimize the false alarms.

Finally, three practical applications are presented. The first is an application of the proposed data-driven robust control for multi-model system. The plant is a reconfigurable industrial oven. To

characterize the various configurations of the highly complex industrial oven, the system is considered as a multi-model system. We aim to design a robust controller capable of handling the uncertainty resulting from the model identification and the uncertainties of the model parameters.
The other two practical contributions are fault detection applications for: a rotating machine and a sliding gate system. The former proposes a signal-based fault detection [29], where signal processing carries out the core work of the detection algorithm. Specifically, the system under analysis is a part of an entire PolyEthylene Terephthalate bottles production line, sensed with accelerometers. The latter describes a complete model-based fault detection work, from the Failures Mode, Effects and Criticality Analysis to the final fault detection algorithm. Also, the fault injection procedure is presented. The considered plant is an Electro-Mechanical Actuator, which is employed to actuate a sliding gate.

Book outline

Chapter 2 gives an overview of the uncertainty representation, usage and design in the system identification procedure. In particular, the uncertainty representation in system modeling, the uncertainty in robust control design, the uncertainty in robust fault detection and the uncertainty in system identification are illustrated. The last focuses on uncertainty modeling with Prediction Error Method, kernel-based system identification and Stochastic Embedding approaches. This chapter is endowed with Appendices A and B, in which some fundamentals of the functional analysis and some knowledge on the model complexity selection are described.

Chapter 3 shows the proposed methodology for designing a data-driven robust controller. The concept of S/T mixed-sensitivity loop-shaping is introduced and the transformation from the a priori knowledge approach into the automatic data-driven methodology are illustrated. Furthermore, the proposed method is tested on a benchmark problem.

Chapter 4 describes an extension of the previous chapter's methodology for multi-model dynamic systems.

Chapter 5 shows the second main theoretical contribution, i.e. the data-driven robust fault detection design. The effectiveness of the proposed procedure is then shown on a benchmark problem.

Chapter 6 shows the application of the proposed data-driven S/T mixed-sensitivity loop-shaping for multi-model systems. The plant is a reconfigurable industrial oven for heat shrinking.

Chapter 7 describes a signal-based fault detection algorithm applied to a rotating machine, which belongs to a PolyEthylene Terephthalate (PET) bottle production line.

Chapter 8 describes a model-based fault diagnosis algorithm applied to Electro-Mechanical Actuator, highlighting the entire design procedure: Failures Mode, Effects and Criticality Analysis, fault injection, fault detection design and fault isolation.

Chapter 1. State of the art

This chapter reviews the state of the art: uncertainty modeling, uncertainty in robust control, uncertainty in fault diagnosis and uncertainty in system identification. As can be easily guessed, the "fil rouge" between all sections is uncertainty information. The first section describes how the uncertainty in dynamical systems is modeled. Instead, the second and third sections review how the uncertainty information, coupled with the model, can be used to design a robust controller or a robust fault diagnosis algorithm. The results of these robust methods consist of: a controller that operates under different conditions, limited by the uncertainty specification, and a residual generator that diagnoses a fault, that occurs to the system, by rejecting the uncertainties, modeled as fictitious noise. The fourth argument focuses on the generation of uncertainty information during the system identification step. In this section, we review also two black-box identification methods: the state-of-the-art method, called *kernel-based system identification*, and the most popular and traditional *Prediction Error Method (PEM)*. This recap is helpful to understand better the uncertainty sources and computation in the identification literature. To complete this literature analysis, the Robust identification methods are shown. This section is completed by the appendices A and B, which review the model complexity selection and functional analysis. Notice that almost all state of the art is described for Single input Single output (SISO) systems, but it is also scalable for Multiple input Multiple output (MIMO) systems.

1.1 Uncertainty in system modeling

The sources of uncertainty can be grouped into three categories:

- *Unpredictable events*: they are typically due to some perturbations generated by the external environment;
- *Unmodeled dynamics*: the model that represents the system, usually, must be tractable (for instance, with the aim of: designing a controller, designing a fault detection algorithm and so on). This specification leads to oversimplified models that neglect some complex dynamics;
- *Poor available data*: the dataset usually is not informative enough for system identification. In a real-world applications, often, is not possible to design an identification experiment that identifies a model with the lowest possible estimation uncertainty. As we will describe in Section 1.4.3, the experiment design is an important step to produce a model endowed with low uncertainty.

Not all of these categories are assessed during the system identification procedure. Nonetheless, it is important to know all possible sources of uncertainty, since this leads to a better understanding of the limits of the uncertainty representation in system identification. Mainly, the model uncertainty representations are grouped in two categories [30]:

- *Parametric uncertainties* or *Structured uncertainties*: deal with a parametric class of models (see Section 1.4.1). In this case the uncertainty information is given as bounds on model's parameters;
- *Non-parametric uncertainties* or *Unstructured uncertainties*: deal with systems in which some complex dynamics are neglected or with systems affected by measurement noise. The information is provided as a single constraint.

1.1.1 Structured Uncertainty

Assume that the uncertainty is represented by the uncertain parameters vector $\boldsymbol{q}(t)$, defined as [31, Chapter 2]:

$$\boldsymbol{q}(t) \equiv [q_1(t), q_2(t), ..., q_l(t)]^\top \in \mathbb{R}^{l\times 1} \quad \forall t \geq 0\,, \tag{1.1}$$

where t is the time index. This leads to two major parametric uncertainty dynamic system representations:

- The state space representation:

$$\begin{cases} \boldsymbol{x}(t) & = \boldsymbol{A}(\boldsymbol{q}(t))\boldsymbol{x}(t) + \boldsymbol{B}(\boldsymbol{q}(t))w(t) \\ z(t) & = \boldsymbol{C}(\boldsymbol{q}(t))\boldsymbol{x}(t) + D(\boldsymbol{q}(t))w(t) \end{cases}; \tag{1.2}$$

  with $\boldsymbol{A} \in \mathbb{R}^{n_x\times n_x}$, $\boldsymbol{B} \in \mathbb{R}^{n_x\times 1}$, $\boldsymbol{C} \in \mathbb{R}^{1\times n_x}$, $D \in \mathbb{R}$, $\boldsymbol{x} \in \mathbb{R}^{n_x\times 1}$ are the state space matrices, $w(t) \in \mathbb{R}$ is a generic input, $z(t) \in \mathbb{R}$ is a generic output and $n_x \in \mathbb{N}^+$ the number of states. In case of this system is an open loop system the notation becomes $w(t) = u(t)$ and $z(t) = y(t)$, where $y(t)$ is the sensed output and $u(t)$ is the system input. From now, with a little abuse of notation, we will refer to the uncertain parameters vector without clarifying the time index t;

- The space of Laplace transforms:

$$Z(s) = G(s, \boldsymbol{q})W(s)\,, \tag{1.3}$$

  where $G(s, \boldsymbol{q})$ is an uncertain rational transfer function. The latter can be written as:

$$G(s, \boldsymbol{q}) \equiv \boldsymbol{C}(\boldsymbol{q})\Big(s\boldsymbol{I}_{n_x} - \boldsymbol{A}(\boldsymbol{q})\Big)^{-1}\boldsymbol{B}(\boldsymbol{q}) + D(\boldsymbol{q}) = \frac{N_g(s, \boldsymbol{q})}{D_g(s, \boldsymbol{q})} = \frac{\sum_{i=0}^{n_n-1} n_{g_i}(\boldsymbol{q})s^i}{\sum_{j=0}^{n_d-1} d_{g_j}(\boldsymbol{q})s^j}\,; \tag{1.4}$$

where:

- $N_g(s, \boldsymbol{q})$ is the numerator of order n_n;
- $D_g(s, \boldsymbol{q})$ is the denominator of order n_d;
- n_{g_i} and d_{g_i} are the coefficients of numerator and denominator.

Note that the uncertainty may afflict the numerator and denominator with different uncertain parameters. In this more general case, the parametric uncertainty becomes: $G(s, \boldsymbol{q}, \boldsymbol{r}) = \frac{N_g(s,\boldsymbol{q})}{D_g(s,\boldsymbol{r})}$, with $\boldsymbol{r} \in \mathbb{R}^{l_r \times 1}$ is the second vector of uncertain parameters.

The parametric uncertainty with linear systems can be rearranged in a more clear representation using the so-called $\Delta - M$ *model*. The structure of this model is represented in Figure 1, where: $\boldsymbol{M}(s) \in \mathcal{RH}_\infty^{(n_{z_\Delta}+n_z)\times(n_{w_\Delta}+n_w)}$ (for SISO systems $n_z = n_w = 1$, which respectively are the number of z signals and w signals) is a transfer function matrix that represents how the uncertainty affects the system, instead $\boldsymbol{\Delta}$ defines the uncertainty and it belongs to a block-diagonal structure set $\bar{C}$:

$$\bar{C} \equiv \left\{ \boldsymbol{\Delta} \in \mathcal{RH}_\infty^{n_{w_\Delta} \times n_{z_\Delta}} : \boldsymbol{\Delta} = \text{bdiag}\left(\boldsymbol{I}_{m_1} q_1, ..., \boldsymbol{I}_{m_l} q_l\right) \right\} ; \tag{1.5}$$

with: $n_{w_\Delta} \in \mathbb{N}^+$ and $n_{z_\Delta} \in \mathbb{N}^+$ are the number of signals that connect $\boldsymbol{M}(s)$ with $\boldsymbol{\Delta}$. Note that the uncertain parameters q_u can be repeated with multiplicity m_u, $u = 1, ..., l$. Usually, they are $n_{w_\Delta} = n_{z_\Delta}$. In this way, the structured uncertainty roughly corresponds to multiple constraints on the uncertain system.

The space $\mathcal{RH}_\infty^{n_{w_\Delta} \times n_{z_\Delta}}$ is defined in [32, Chapter 3] as:

Definition 1.1: $\mathcal{RH}_\infty^{r,c}$

$\mathcal{RH}_\infty^{r,c}$, with generic $r, c \in \mathbb{Z}^+$, is the space of stable proper real rational $r \times c$ matrix transfer functions $\boldsymbol{F}(s)$, where exists the $\mathcal{H}_\infty$ norm of $\boldsymbol{F}(s)$:

$$\|\boldsymbol{F}(s)\|_\infty = \sup_{\omega \in \mathbb{R}} |\boldsymbol{F}(j\omega)| \equiv \operatorname{ess\,sup}_{\omega \in \mathbb{R}} \bar{\sigma}(\boldsymbol{F}(j\omega)) , \tag{1.6}$$

where $\bar{\sigma}(\cdot)$ denotes the largest singular value of the frequency response of the $\boldsymbol{F}(s)$ and ess represents the supremum (or least upper bound).

The $\Delta - M$ model establishes the relation $\boldsymbol{w}_\Delta(t) = \boldsymbol{\Delta} \boldsymbol{z}_\Delta(t)$, which describes how the uncertainty acts on the vector of scalar signals $\boldsymbol{z}_\Delta \in \mathbb{R}^{n_{z_\Delta} \times 1}$ to produce $\boldsymbol{w}_\Delta \in \mathbb{R}^{n_{w_\Delta} \times 1}$. By partitioning the transfer

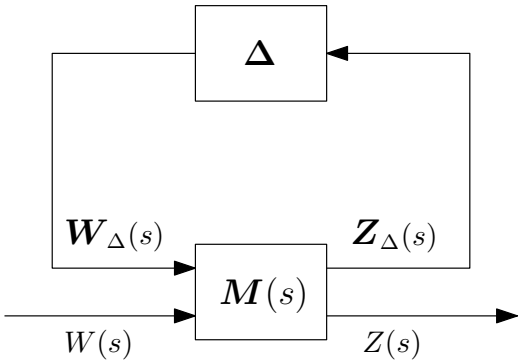


Figure 1: $\Delta - M$ model resulting from the linear fractional transformation.

function matrix $\boldsymbol{M}(s)$, Figure 1 can be written as:

$$\begin{bmatrix} \boldsymbol{Z}_\Delta(s) \\ Z(s) \end{bmatrix} = \begin{bmatrix} \boldsymbol{M}_{11}(s)\boldsymbol{M}_{12}(s) \\ \boldsymbol{M}_{21}(s)\boldsymbol{M}_{22}(s) \end{bmatrix} \begin{bmatrix} \boldsymbol{W}_\Delta(s) \\ W(s) \end{bmatrix} \tag{1.7}$$

with $\boldsymbol{M}_{11}(s) \in \mathcal{RH}^{n_{z_\Delta} \times n_{w_\Delta}}$,$\boldsymbol{M}_{12}(s) \in \mathcal{RH}^{n_{z_\Delta} \times 1}$, $\boldsymbol{M}_{21}(s) \in \mathcal{RH}^{1 \times n_{w_\Delta}}$, $\boldsymbol{M}_{22}(s) \in \mathcal{RH}$ and $\boldsymbol{Z}_\Delta(s), Z(s), \boldsymbol{W}_\Delta(s), W(s)$ are the Laplace transformation of the respectively signals. With the representation (1.7), it is possible to employ a mathematical framework called upper *Linear Fractional Transformation (LFT)*, which allows defining the transfer function that ties $W(s)$ with $Z(s)$:

$$\mathcal{F}_u(\boldsymbol{M}(s), \boldsymbol{\Delta}) \equiv M_{22}(s) + \boldsymbol{M}_{21}(s)\boldsymbol{\Delta}(\boldsymbol{I}_{n_{w_\Delta}} - \boldsymbol{M}_{11}(s)\boldsymbol{\Delta})^{-1}\boldsymbol{M}_{12}(s)\ . \tag{1.8}$$

This representation is *well-posed* if $(\boldsymbol{I}_{n_{w_\Delta}} - \boldsymbol{M}_{11}(j\omega)\boldsymbol{\Delta})$ is non-singular $\forall \boldsymbol{\Delta}$ and with $\omega = \infty$. Furthermore, if $(\boldsymbol{I} - \boldsymbol{M}_{11}(s)\boldsymbol{\Delta}(s))^{-1} \in \mathcal{RH}_\infty^{n_{w_\Delta} \times n_{w_\Delta}}$, then $\boldsymbol{M}(s) \in \mathcal{RH}_\infty^{(n_{z_\Delta}+1)\times(n_{w_\Delta}+1)}$ and $\boldsymbol{\Delta} \in \mathcal{RH}_\infty^{n_{w_\Delta} \times n_{z_\Delta}}$, as defined in (1.5). Hence, the feedback interconnection of the upper LFT is internally stable [32, Chapter 3]. The definition of the internally stability adjective is:

Definition 1.2: Internally stable

A feedback interconnection is **internally stable** if all signals in the system are bounded provided that the injected signals at any location are bounded.

The upper LFT (1.8) is in the space of Laplace transforms, but it is also applicable in the state space as:

$$\begin{cases} \boldsymbol{x}(t) & = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}_1\boldsymbol{w}_\Delta(t) + \boldsymbol{B}_2 w(t) \\ \boldsymbol{z}_\Delta(t) & = \boldsymbol{C}_1\boldsymbol{x}(t) + \boldsymbol{D}_{11}\boldsymbol{w}_\Delta(t) + \boldsymbol{D}_{12} w(t) \\ z(t) & = \boldsymbol{C}_2\boldsymbol{x}(t) + \boldsymbol{D}_{21}\boldsymbol{w}_\Delta(t) \\ \boldsymbol{w}_\Delta(t) & = \boldsymbol{\Delta}\boldsymbol{z}_\Delta(t) \end{cases}; \tag{1.9}$$

with $\boldsymbol{B}_1 \in \mathbb{R}^{n_x \times n_{w_\Delta}}, \boldsymbol{B}_2 \in \mathbb{R}^{n_x \times 1}, \boldsymbol{C}_1 \in \mathbb{R}^{n_{z_\Delta} \times n_x}, \boldsymbol{D}_{11} \in \mathbb{R}^{n_{z_\Delta} \times n_{w_\Delta}}, \boldsymbol{D}_{12} \in \mathbb{R}^{1 \times n_{z_\Delta}}, \boldsymbol{C}_2 \in \mathbb{R}^{1 \times n_x}, \boldsymbol{D}_{21} \in \mathbb{R}^{1 \times n_{w_\Delta}}$ [32].

The example 1.1 shows the efficiency of the LFT representation.

Example 1.1: Linear Fractional Transformation (LFT) for pulling out the parametric uncertainty

Consider an open loop system described by the following transfer function

$$G(s;\boldsymbol{q}) = \frac{s+3+q_1}{s^2+(2+q_1)s+5+q_2}, \tag{1.10}$$

where the vector $\boldsymbol{q} = [q_1, q_2]^\top \in \mathbb{R}^{2\times 1}$ represents the parametric uncertainty vector. The graphical representation of (1.10) is depicted in Figure 2. The green area distinguishes the uncertainty parameters from well-known parts. By employing the upper LFT, it is possible to redraw the transfer function as depicted in Figure 3. Thus, we obtain:

$$\boldsymbol{M}(s) = \left[\begin{array}{ccc|c} 0 & \frac{-1}{s^2+2s+5} & \frac{-1}{s^2+2s+5} & \frac{1}{s^2+2s+5} \\ 0 & \frac{-s}{s^2+2s+5} & \frac{-s}{s^2+2s+5} & \frac{s}{s^2+2s+5} \\ 0 & \frac{-1}{s^2+2s+5} & \frac{-1}{s^2+2s+5} & \frac{1}{s^2+2s+5} \\ \hline 1 & \frac{-(s+3)}{s^2+2s+5} & \frac{-(s+3)}{s^2+2s+5} & \frac{s+3}{s^2+2s+5} \end{array}\right] \quad \boldsymbol{\Delta} = \begin{bmatrix} q_1 & 0 & 0 \\ 0 & q_1 & 0 \\ 0 & 0 & q_2 \end{bmatrix}. \tag{1.11}$$

Doing so, we put all the uncertainties in $\boldsymbol{\Delta}$ matrix. The lines in $\boldsymbol{M}(s)$ highlight the sub matrices $\boldsymbol{M}_{11}(s)$, $\boldsymbol{M}_{12}(s)$, $\boldsymbol{M}_{21}(s)$, $M_{22}(s)$.

The structure of the uncertainty can be generalized as:

$$\boldsymbol{\Delta}(s) = \begin{bmatrix} q_1\boldsymbol{I}_{m_1} & \dots & \mathbf{0} & & & \\ \vdots & \ddots & \vdots & & \mathbf{0} & \\ \mathbf{0} & \dots & q_l\boldsymbol{I}_{m_l} & & & \\ & & & \Delta_1(s) & \dots & \mathbf{0} \\ & \mathbf{0} & & \vdots & \ddots & \vdots \\ & & & \mathbf{0} & \dots & \Delta_c(s) \end{bmatrix} \tag{1.12}$$

with $\Delta_1(s), ..., \Delta_c(s) \in \mathcal{RH}_\infty$ represent the bounded ($\|\Delta_i(s)\|_\infty < 1$), stable and proper transfer functions. This allows considering both linear uncertainty that afflicts the parameters and nonlinear uncertainties, represented by $\Delta_1(s), ..., \Delta_c(s)$ with $c \in \mathbb{N}^+$.

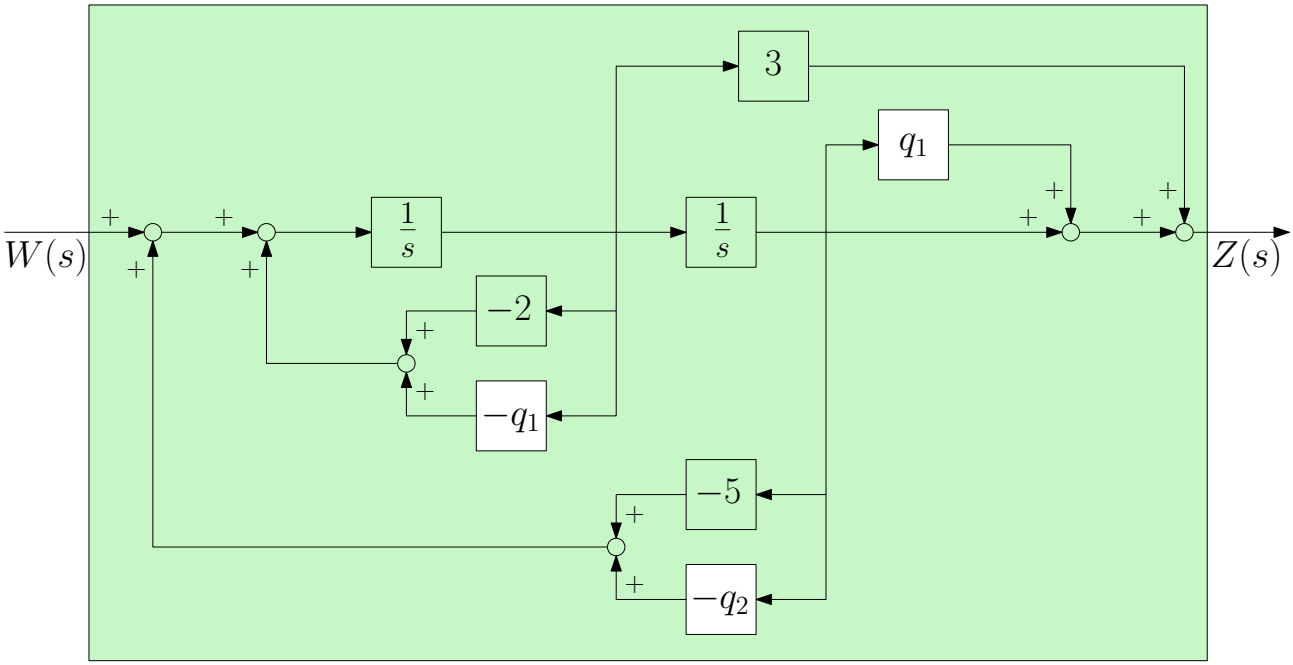


Figure 2: Block scheme representation of the considered system.

Another more general and complex representation redesigns the state space domain by replacing the fourth equation of (1.9),with:

$$\boldsymbol{w}_\Delta(t) = \boldsymbol{\Delta}\Big(\boldsymbol{x}(\cdot), w(\cdot), t\Big)\boldsymbol{z}_\Delta(t) \tag{1.13}$$

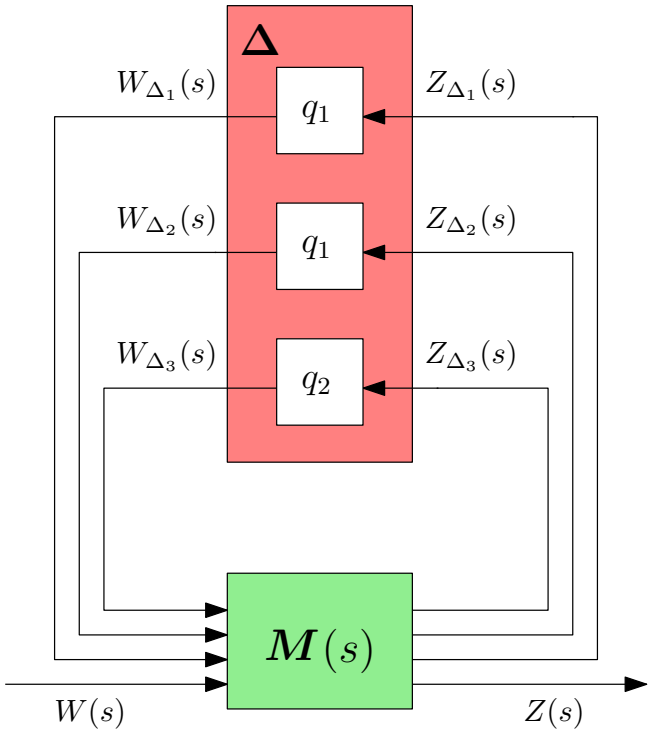


Figure 3: Upper LFT scheme of the example system.

by assuming $\boldsymbol{D}_{11} = \mathbf{0}$. This uncertainty representation is admissible for (1.9) if, given a locally square integrable input $w(\cdot)$ and any corresponding solutions to (1.9), with (1.13) defined on an existence interval $(0, \bar{t})$, there exist a sequence $t_{i=1}^{\infty}$ and constants $d_1 \geq \ldots \geq d_{n_{w_\Delta}} \geq 0$, such that $t_i \to \bar{t}, t_i \geq 1$ and

$$\int_0^{t_i} \left|w_{\Delta_k}(t)\right|^2 \, dt \leq \int_0^{t_i} \left|z_{\Delta_k}(t)\right|^2 \, dt + d_k \,, \tag{1.14}$$

where $w_{\Delta_k}(t)$ and $z_{\Delta_k}(t)$ are the k-th element of respectively $\boldsymbol{w}_\Delta(t)$ and $\boldsymbol{z}_\Delta(t)$, with $k = 1, \ldots, n_{w_\Delta}$ (considering $n_{w_\Delta} = n_{z_\Delta}$). This representation is called *IQC uncertainty description* [33, 34]. It is useful to exploit structural information (such as uncertainty of some parameters) for characterizing the property of $\boldsymbol{w}_\Delta$ and $\boldsymbol{z}_\Delta$ and for studying the non-linear uncertainties. In particular, there are two ways to implement it: in time domain or frequency domain. The choice depends on the application under analysis.

Remark 1.1

The structured uncertainty matrix $\boldsymbol{\Delta}$ can be an unknown (real or complex) matrix subject to a matrix norm bound $\|\boldsymbol{\Delta}\| \leq 1$, as defined in (1.5), or even an unknown transfer function matrix subject to a norm bound $\|\boldsymbol{\Delta}(s)\|_\infty \leq 1$.

1.1.2 Unstructured Uncertainty

The LFT provides also a representation for the unstructured uncertainties. This typology of uncertainty is defined by a norm that bounds Δ[1] with a single constraint. A type of uncertainty model that belongs to this group is called *Norm bounded uncertainty*. This works with a non-dynamic time-varying nonlinear uncertainty, specified by an induced norm bound, i.e. $\|\Delta(\boldsymbol{x}, t)\|_2 \leq 1 \, \forall t$, in state space representation [35]. Instead, a more general unstructured uncertainty model is defined as a dynamic nonlinear uncertainty, denoted by the transfer function $\Delta(s)$, constrained as $\|\Delta(s)\|_\infty \leq 1$. This type is called *Bounded real uncertainty*.

Assume that $\boldsymbol{M}(s)$ represents an open-loop system (i.e. $w(t) = u(y)$ and $z(t) = y(t)$ of Figure 1), the most common types of Bounded real unstructured uncertainty are [26, Chapter 9] [24, Chapter 4]:

- *Additive uncertainty*: which is derived from the upper LFT equation (1.8) by letting $M_{11}(s) = 0$, $M_{22}(s) = G_0(s)$, $M_{12}(s) = 1$[2] and by choosing a suitable weight transfer function to define $M_{21}(s)$, denoted by $W_a(s)$. Hence, the output of the uncertain system is defined as

$$Y(s) = \Big[G_0(s) + W_a(s)\Delta(s) \Big] U(s) \tag{1.15}$$

  where $G_0(s)$ represents the nominal model. Figure 4 depicts the graphical representation of

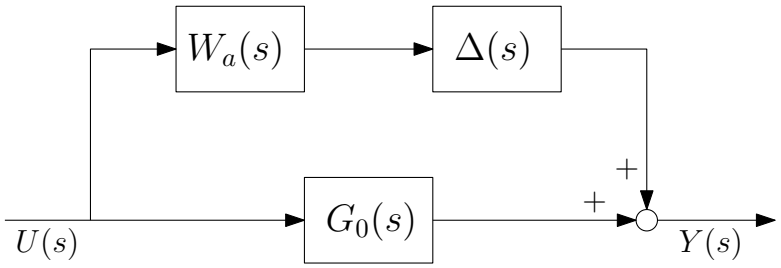


Figure 4: Graphical representation of additive uncertainty.

  the additive uncertainty.

- *Multiplicative input uncertainty*: which is derived from (1.8) by setting $M_{11}(s) = 0$, $M_{21}(s) = G_0(s)$, $M_{22}(s) = G_0(s)$, and by choosing a suitable weight transfer function matrix to define $M_{12}(s)$, denoted by $W_i(s)$. Thus

$$Y(s) = G_0(s) \Big[1 + \Delta(s) W_i(s) \Big] U(s) \, . \tag{1.16}$$

  Figure 5 depicts the graphical representation of the multiplicative input uncertainty.

[1] Note that Δ is written without the bold symbol, because for the unstructured uncertainty in SISO case it is not a matrix.

[2] With SISO systems $\boldsymbol{M}(s) \in \mathcal{RH}_\infty^{2\times 2}$, since $n_{w_\Delta} = n_{z_\Delta} = n_z = n_w = 1$, the sub-matrices of $\boldsymbol{M}(s)$ are all scalars transfer functions.

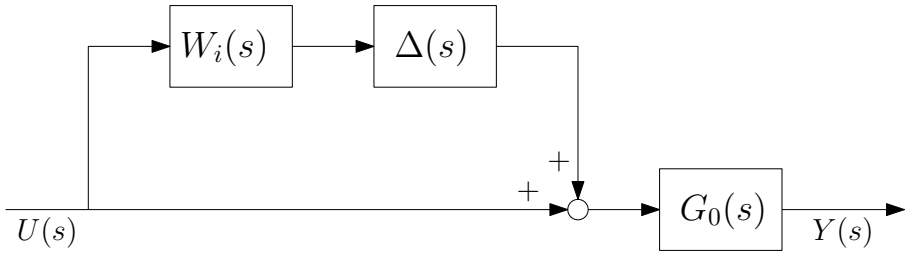


Figure 5: Graphical representation of multiplicative input uncertainty.

- *Multiplicative output uncertainty*: which is derived from (1.8) by imposing $M_{11}(s) = 0$, $M_{12}(s) = G_0(s)$, $M_{22}(s) = G_0(s)$, and by choosing suitable weight transfer function matrix to define $M_{21}(s)$, denoted by $W_o(s)$, so

$$Y(s) = \Big[1 + \Delta(s)W_o(s)\Big]G_0(s)U(s)\ . \tag{1.17}$$

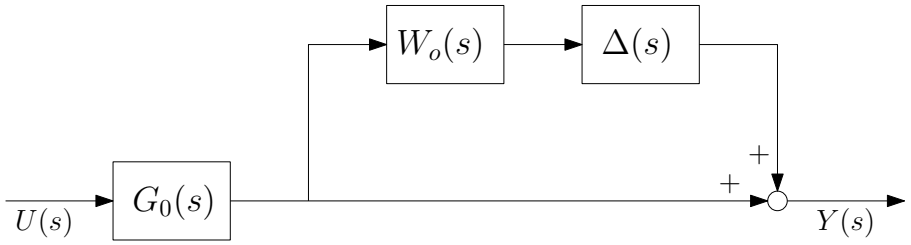


Figure 6: Graphical representation of multiplicative output uncertainty.

In the SISO case the multiplicative output and input uncertainty coincide, instead with MIMO systems they differ. Figure 6 shows the block scheme of the multiplicative output uncertainty.

- *Inverse multiplicative output uncertainty*: which is computed from (1.8) by setting $M_{12}(s) = G_0(s)$, $M_{22}(s) = G_0(s)$, and by choosing a suitable weight transfer function matrix to define $M_{21}(s) = M_{11}(s) = W_{Io}(s)$. Thus

$$\begin{aligned} Y(s) &= \Big[G_0(s) + W_{Io}(s)\Delta(s)\Big(1 - W_{Io}(s)\Delta(s)\Big)^{-1}G_0\Big]U(s) \\ &= \Big[1 + W_{Io}(s)\Delta(s)\Big(1 - W_{Io}(s)\Delta(s)\Big)^{-1}\Big]G_0(s)U(s) \\ &= \Big[1 + \Big(1 - W_{Io}(s)\Delta(s)\Big)^{-1} - 1\Big]G_0(s)U(s) \\ &= \Big(1 - W_{Io}(s)\Delta(s)\Big)^{-1}G_0(s)U(s) \end{aligned}\ . \tag{1.18}$$

Figure 7 shows the graphic interpretation of the inverse multiplicative output uncertainty.

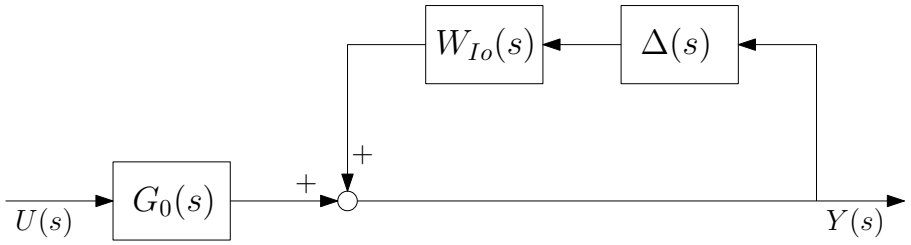


Figure 7: Graphical representation of inverse multiplicative output uncertainty.

- *Left coprime factor uncertainty*: by imposing $G_0(s) = D_{lcf}(s)^{-1}N_{lcf}(s)$ (see Definition 1.8) and by having two different $\Delta_n(s), \Delta_d(s)$ uncertainty transfer functions, one for $N_{lcf}(s)$ and one for $D_{lcf}(s)$, that comply with

$$\left\| \begin{bmatrix} \Delta_n(s), \Delta_d(s) \end{bmatrix} \right\|_\infty \leq 1 \, . \tag{1.19}$$

The input-output relations of $\boldsymbol{M}(s)$ (see Figure 1), considered as an open loop system, becomes

$$Y(s) = \Big[D_{lcf}(s) + \Delta_d(s) \Big]^{-1} \Big[N_{lcf}(s) + \Delta_n(s) \Big] U(s) \tag{1.20}$$

by setting

$$\boldsymbol{M}(s) = \left[\begin{array}{c|c} -D_{lcf}^{-1} & -G_0(s) \\ 0 & 1 \\ \hline D_{lcf}(s)^{-1} & G_0(s) \end{array}\right] . \tag{1.21}$$

The uncertainty in left coprime factorization can be also weighted, such as the previous forms, by setting $W_d(s), W_n(s)$. Therefore, the formulation (1.20) becomes

$$Y(s) = \Big[D_{lcf}(s) + W_d(s)\Delta_d(s) \Big]^{-1} \Big[N_{lcf}(s) + W_n(s)\Delta_n(s) \Big] U(s) \, . \tag{1.22}$$

These two representations of the left coprime factorization are useful for different aims. The unweighted form is employed in $\mathcal{H}_\infty$ loop-shaping design and robust stability analysis, instead the weighted form is used for the mixed-sensitivity loop-shaping (see Section 1.2).

Figure 8 illustrates the graphical representation of the weighted left coprime factorization uncertainty. There are also other representations of uncertainty, such as inverse multiplicative input uncertainty, inverse additive uncertainty and right coprime factorization, but they are less used. The most employed

Type of uncertainty modeling of Section 1.1.2	*Uncertainty physical source*
Additive (1.15)	• Additive plant errors • Neglected high frequency dynamics • Uncertain right half plane zeros
Input multiplicative (1.16)	• Actuators errors • Neglected high frequency dynamics • Uncertain right half plane zeros
Output multiplicative (1.17)	• Sensors errors • Neglected high frequency dynamics • Uncertain right half plane zeros
Inverse output multiplicative (1.18)	• Low frequency parameter errors • Uncertain right half plane poles
Left coprime factorization (1.20)	• Low frequency parameter errors • Neglected high frequency dynamics • Uncertain right half plane poles and zeros

Table 1: Physical sources of uncertainty accounted by the uncertainty models.

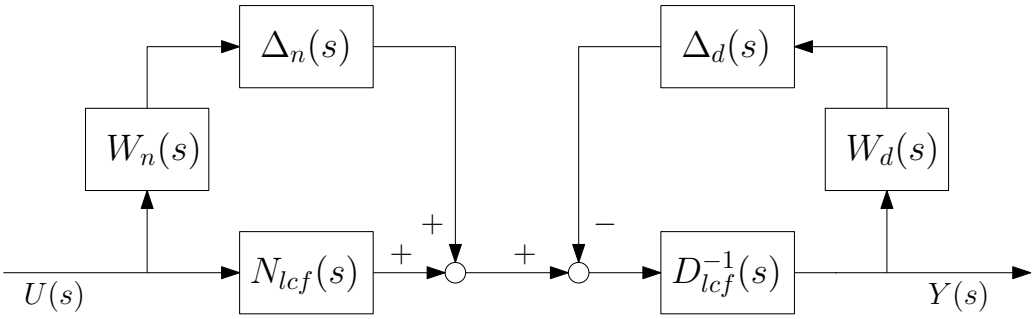


Figure 8: Graphical representation of weighted left coprime factorization uncertainty.

form is the multiplicative output uncertainty. Table 1 reports a recap that shows which types of physical sources are characterized by the corresponding unstructured uncertainty scheme.

Another two types of unstructured uncertainties, in addition to the norm bounded uncertainty and the bounded real uncertainty, are: *Positive real uncertainty* and *Negative imaginary uncertainty*. Positive real uncertainty considers a stable transfer function matrix $\mathbf{\Delta}(s)$ that complies with the condition:

$$\mathbf{\Delta}(j\omega) + \mathbf{\Delta}(j\omega)^* \geq 0 \quad \forall\omega\,, \tag{1.23}$$

where $\boldsymbol{A} \geq 0$ denotes that the matrix $\boldsymbol{A}$ is Hermitian and positive semidefinite. Negative imaginary uncertainty is similar to Positive real uncertainty, but the uncertainty constraint is:

$$j(\mathbf{\Delta}(j\omega) + \mathbf{\Delta}(j\omega)^*) \geq 0 \quad \forall\omega > 0\,. \tag{1.24}$$

The equations (1.23) and (1.24) are used mainly with MIMO systems, specifically with mechanical systems. For instance, Negative imaginary uncertainty is used when the input-output data are force and position; instead, Positive real uncertainty is used when the available data are force and speed [36].

1.2 Uncertainty for robust control

Models endowed with the uncertainty information, represented by the LFT, as described in Section 1.1, can be used to design a control that guarantees the robust stability of an uncertain system. The link between the uncertain model and control design is represented by the *Small gain theorem*. Before introducing this, it is necessary to state the definition of robust stability:

Definition 1.3: Robust stability [34]

Consider the system in Figure 1 with $\boldsymbol{M}(s) \in \mathcal{RH}_\infty^{(n_{z_\Delta}+1)\times(n_{w_\Delta}+1)}$ and the unstructured uncertainty $\Delta(s) \in \mathcal{RH}_\infty$. This interconnection is said **robust stable** if it is stable for all uncertainties

$\Delta(s)$ in a given norm bounded set

$$\mathcal{B}_\Delta(\nu) \equiv \{\Delta(s) \in \mathcal{RH}_\infty : \|\Delta(s)\|_\infty \leq \nu\} \ , \tag{1.25}$$

where ν is a fixed radius.

Note that ν is a measure of the robustness of the uncertain system. Usually, the largest value of ν that guarantees the robust stability of the system is called *robustness margin*.

Theorem 1.1: Small gain theorem [37]

Consider the system in Figure 1 with $\boldsymbol{M}(s) \in \mathcal{RH}_\infty^{(n_{z_\Delta}+1)\times(n_{w_\Delta}+1)}$ and an unknown unstructured uncertainty $\Delta(s)$ with $\nu > 0$. The interconnection is well-posed and internally stable for all $\Delta(s) \in \mathcal{RH}_\infty$ with:

- $\|\Delta(s)\|_\infty \leq \nu$ if and only if $\|\boldsymbol{M}(s)\|_\infty < \frac{1}{\nu}$;
- $\|\Delta(s)\|_\infty < \nu$ if and only if $\|\boldsymbol{M}(s)\|_\infty \leq \frac{1}{\nu}$.

It follows that the stability radius for uncertain Linear Time-Invariant (LTI) systems $\nu_{LTI}\left(\boldsymbol{M}(s)\right)$ is given by:

$$\nu_{LTI}\left(\boldsymbol{M}(s)\right) \equiv \frac{1}{\|\boldsymbol{M}(s)\|_\infty} \ , \tag{1.26}$$

where the stability radius is defined as follows:

Definition 1.4: Stability radius [34]

The **stability radius** is the smallest value of ν such that exists $\Delta(s) \in \mathcal{B}_\Delta(\nu)$.

The Small gain theorem, described in Theorem 1.1, establishes an equivalence between the dynamic uncertainty $\Delta(s) \in \mathcal{RH}_\infty$ and static complex uncertainty $\Delta(j\omega) \in \mathbb{C}$:

Definition 1.5: Static and dynamic uncertainty [34]

Consider the system in Figure 1 with $\boldsymbol{M}(s) \in \mathcal{RH}_\infty^{(n_{z_\Delta}+1)\times(n_{w_\Delta}+1)}$ with $\nu > 0$. The interconnection is well-posed and internally stable $\forall\Delta(s) \in \mathcal{RH}_\infty$ with $\|\Delta(s)\|_\infty \leq \nu$, if and only if the interconnection is well-posed and internally stable $\forall\Delta(j\omega) \in \mathbb{C}$ with $\|\Delta(j\omega)\| \leq \nu$.

So, the robust stability feature of system, depicted in Figure 1, with static complex uncertainty $\Delta(j\omega)$ is necessary and sufficient for having also the robust stability of the same system under general dynamic uncertainty $\Delta(s) \in \mathcal{RH}_\infty$.

As described previously in Section 1.1, the LFT with unstructured uncertainty can be declined in various types of uncertainty models. Hence, also the Small gain theorem can be developed into different robust stability tests:

- *Additive uncertainty* $G(s) = G_0(s) + \Delta(s)W_a(s)$: by having a stabilizing controller $K(s)$ for $G_0(s)$, the closed loop is well-posed and internally stable $\forall\Delta(s) \in \mathcal{RH}_\infty$ with
  - $\|\Delta(s)\|_\infty < 1$ if and only if $\|W_a(s)K(s)S_0(s)\|_\infty \leq 1$,
  - $\|\Delta(s)\|_\infty \leq 1$ if and only if $\|W_a(s)K(s)S_0(s)\|_\infty < 1$,

  where $S_0(s) = \frac{1}{1+G_0(s)K(s)}$ is the nominal sensitivity function.
- *Multiplicative uncertainty* $G(s) = \Big(1 + \Delta(s)W_o(s)\Big)G_0(s)$: by having a stabilizing controller $K(s)$ for $G_0(s)$, the closed loop is well-posed and internally stable $\forall\Delta(s) \in \mathcal{RH}_\infty$ with
  - $\|\Delta(s)\|_\infty < 1$ if and only if $\|W_oT_0(s)\|_\infty \leq 1$,
  - $\|\Delta(s)\|_\infty \leq 1$ if $\|W_oT_0(s)\|_\infty < 1$,
  - $\|\Delta(s)\|_\infty \leq 1$ only if $\|W_oT_0(s)\|_\infty \leq 1$,

  where $T_0(s) = 1 - S_0(s)$ is the nominal complementary sensitivity function. Output and input multiplicative uncertainty modeling share the same robust stability test.

Remark 1.2: [26, Chapter 9]

The robust stability of the closed loop system for all $\forall\Delta(s) \in \mathcal{RH}_\infty$ with $\|\Delta(s)\|_\infty \leq 1$ does not necessary imply $\|W_oT_0(s)\|_\infty < 1$.

- *Left coprime uncertainty* $G(s) = \Big[D_{lcf}(s)+\Delta_d(s)\Big]^{-1}\Big[N_{lcf}(s)+\Delta_n(s)\Big]$: by having a stabilizing controller $K(s)$ for $G_0(s)$, the closed-loop system is well-posed and internally stable $\forall\mathbf{\Delta}_{lfc}(s) = [\Delta_d(s), \Delta_n(s)] \in \mathcal{RH}_\infty^{2\times1}$ with $\left\|\mathbf{\Delta}_{lfc}(s)\right\|_\infty < 1$ if and only if

$$\left\|\begin{bmatrix} K(s) \\ \\ 1 \end{bmatrix} S_0(s)D_{lcf}^{-1}(s)\right\|_\infty \leq 1\,. \tag{1.27}$$

  Again, in this uncertain system model, the weight functions, depicted in Figure 8, are considered external from the uncertain model for control design purposes. Therefore, the robust stability test does not account those weights.

All proofs are reported in [26, Chapter 9]. Table 2 resumes a series of robust stability tests of almost all types of perturbed system, by letting $W_a(s), W_o(s), W_i(s)\ , W_{Io}(s) \in \mathcal{RH}_\infty$ and $\Delta(s) \in \mathcal{RH}_\infty$ with $\|\Delta(s)\|_\infty < 1$.

Type of uncertainty modeling	*Robust stability test*
Additive	$\|W_a(s)K(s)S_0(s)\|_\infty \leq 1$
Input multiplicative	$\|W_i(s)T_0(s)\|_\infty \leq 1$
Output multiplicative	$\|W_o(s)T_0(s)\|_\infty \leq 1$
Inverse output multiplicative	$\|W_{Io}(s)S_0(s)\|_\infty \leq 1$
Right coprime factorization	$\left\|D_{rcf}^{-1}(s)S_0(s)\begin{bmatrix}K(s), 1\end{bmatrix}\right\|_\infty \leq 1$
Left coprime factorization	$\left\|\begin{bmatrix}K(s)\\ 1\end{bmatrix}S_0(s)D_{lcf}^{-1}(s)\right\|_\infty \leq 1$

Table 2: Robust stability test at varying the unstructured uncertainty model.

The resulting components of the right coprime factorization of $\boldsymbol{M}(s)$ are denoted as $D_{rcf}(s)\ N_{rcf}(s)$. Notice that all rows in Table 2 can be resumed by the *unstructured analysis theorem*:

Theorem 1.2: Unstructured analysis theorem

Given a uncertain system $G(s)$ and a controller $K(s)$ that stabilizes the nominal plant $G_0(s)$, closed-loop robust stability is achieved if and only if the robust stability test of the employed uncertainty modeling is valid.

Considering the multiplicative uncertainty[3] with $\|\Delta(s)\|_\infty \leq 1$, it is possible to generalize the Small gain theorem as the Nyquist criterion for the stability of a feedback system:

$$\begin{aligned} \|W_o(s)T_0(s)\|_\infty < 1 \Leftrightarrow \quad & \left|\frac{W_o(j\omega)L_0(j\omega)}{1+L_0(j\omega)}\right| < 1 \quad \forall\omega \\ \Leftrightarrow \quad & |W_o(j\omega)L_0(j\omega)| < |1 + L_0(j\omega)| \quad \forall\omega \end{aligned} \quad , \tag{1.28}$$

with $L_0(s) = G_0(s)K(s)$. Notice that the distance between -1 and $L_0(s)$ is represented by $|1 + L_0(j\omega)|$ $\forall\omega$, therefore, thanks to the Nyquist stability theorem, the closed loop system is stable if the Nyquist plot does not encircle the critical point -1. So, the system is robust stable if the distance between -1 and $L_0(j\omega)$ is higher than the absolute value of $W_o(j\omega)L_0(j\omega)$. These propositions are true if $L_0(s)$ has not right half plane poles. The graphical representation is depicted in Figure 9. For simplicity, $L_0(j\omega)$ and $W_o(j\omega)$ dependencies with respect to the frequency are not reported.

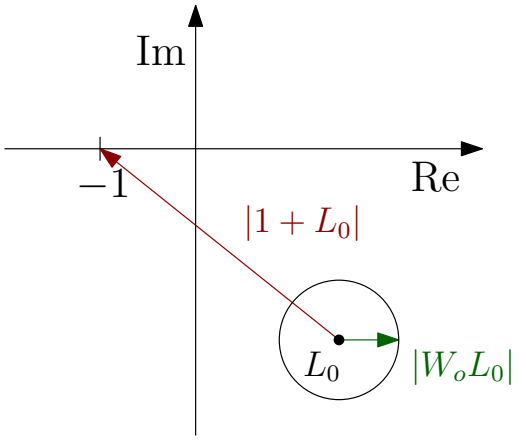


Figure 9: Nyquist representation of robust stability with multiplicative unstructured uncertainty.

The literature shows also the possibility of adding some performance requirements to robust stability, such as:

- *Nominal performance*: some performance objectives are satisfied for the nominal plant $G_0(s)$;
- *Robust performance*[4]: some performance objectives are satisfied for every plant in the uncertainty model $G(s)$, with $\Delta(s) \in \mathcal{B}_\Delta(\nu)$ and $\nu = 1$.

Usually, the performance requirements are designed with suitable weight functions that shape open-loop or closed-loop frequency responses [25, Chapter 8]. The robust performance test changes by

[3]Since for the SISO systems the input multiplicative uncertainty and output multiplicative uncertainty are equivalent, often the input/output adjectives are omitted.

[4]The prerequisites of the robust performance are nominal performance and robust stability.

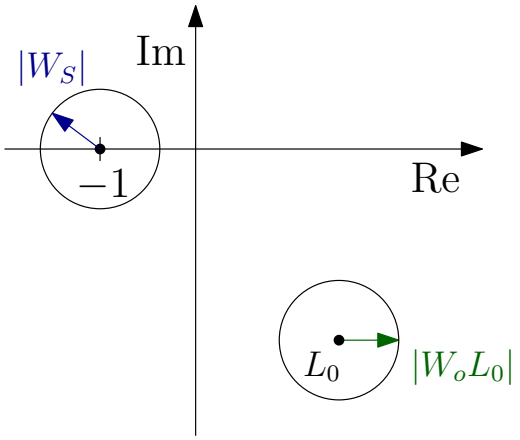


Figure 10: Nyquist representation of robust performance with multiplicative unstructured uncertainty.

varying the unstructured uncertainty scheme employed and by modifying the nominal performance test. Figure 10 depicts the robust performance representation in the Nyquist plot. To obtain a controller which have this property, the two circles must not have an intersection.

In literature, there is a variation of Small gain theorem called *Small μ theorem*, used for systems with structured uncertainty. To describe this theorem, it is necessary to explain the different structured uncertainty sets. With the more general uncertainty matrix described in (1.12), the corresponding structured set is:

$$\tilde{\mathcal{A}} \equiv \left\{ \boldsymbol{\Delta}(s) = \text{bdiag}\left(\boldsymbol{I}_{m_1} q_1, ..., \boldsymbol{I}_{m_l} q_l, \Delta_1(s), ..., \Delta_c(s) \right) \right\} , \tag{1.29}$$

where q_u are real (or complex) uncertain parameters, with multiplicity m_u with $u = 1, ..., l$, and $\Delta_i(s)$ with $i = 1, ..., c$ are real (or complex) transfer functions. If we replace the last c transfer functions with the corresponding static real (or complex) $\Delta_i(j\omega)$ for a fixed frequency ω, the structured set becomes:

$$\mathcal{A} \equiv \left\{ \boldsymbol{\Delta}(j\omega) = \text{bdiag}\left(\boldsymbol{I}_{m_1} q_1, ..., \boldsymbol{I}_{m_l} q_l, \Delta_1(j\omega), ..., \Delta_c(j\omega) \right) \right\} . \tag{1.30}$$

Therefore, the corresponding norm bounded set is equal to:

$$\mathcal{B}_{\mathcal{A}} \equiv \left\{ \boldsymbol{\Delta}(j\omega) \in \mathcal{A} : \|\boldsymbol{q}\|_p \leq 1, \bar{\sigma}(\Delta_i(j\omega)) \leq 1, i = 1, ..., c \right\} ; \tag{1.31}$$

with $\bar{\sigma}(\cdot)$ the largest singular value and $\|\cdot\|_p$ the p-norm.

Furthermore, the Small μ theorem relies on the definition of the *Structured singular value*:

Definition 1.6: Structured singular value

Letting the complex matrix $\boldsymbol{M}(j\omega) \in \mathbb{C}^{(n_{z_\Delta}+1)\times(n_{w_\Delta}+1)}$, considered as the evaluation of $\boldsymbol{M}(s)$ for $s = j\omega$ with $\omega > 0$, the **structured singular value** of $\boldsymbol{M}(j\omega)$ with respect to $\mathcal{A}$ is:

$$\mu_{\mathcal{A}}(\boldsymbol{M}(j\omega)) \equiv \frac{1}{\min\left\{ \bar{\sigma}(\boldsymbol{\Delta}(j\omega)) : \det(\boldsymbol{I} - \boldsymbol{M}(j\omega)\boldsymbol{\Delta}(j\omega)) = 0, \boldsymbol{\Delta}(j\omega) \in \mathcal{A} \right\}} . \tag{1.32}$$

An alternative expression of (1.32) corresponds to:

$$\mu_{\mathcal{A}}(\boldsymbol{M}(j\omega)) \equiv \max_{\boldsymbol{\Delta}(j\omega)\in\mathcal{B}_{\mathcal{A}}} \rho\left(\boldsymbol{M}(j\omega)\boldsymbol{\Delta}(j\omega)\right) , \tag{1.33}$$

with $\rho(\cdot)$ the maximum modulus of the eigenvalues, called *Spectral radius*.

Now, it is possible to enunciate the Small μ theorem:

Theorem 1.3: Small μ theorem

Consider the system of Figure 1 with $\boldsymbol{M}(s) \in \mathcal{RH}_{\infty}^{\left(n_{z_\Delta}+1\right)\times\left(n_{w_\Delta}+1\right)}$ and an unknown structured uncertainty $\boldsymbol{\Delta}(s)$ with $\nu > 0$. The interconnection is said internally stable and well-posed $\forall\boldsymbol{\Delta}(s) \in \tilde{\mathcal{A}}$ with $\|\boldsymbol{\Delta}(s)\|_{\infty} \leq \nu$ if and only if

$$\sup_{\omega\in\mathbb{R}} \mu_{\mathcal{A}}\left(\boldsymbol{M}(j\omega)\right) < \frac{1}{\nu} . \tag{1.34}$$

The Small μ theorem, in Theorem 1.3, counterpart explains the same equivalence between the static and dynamic uncertainty with structured uncertainty, i.e. it establishes an equivalence between the dynamic uncertainty $\boldsymbol{\Delta}(s)$ and static complex uncertainty $\boldsymbol{\Delta}(j\omega)$ (see Definition 1.5).

Also with structured uncertainty, it is possible to design a controller that complies with the robust performance specifications. In [38], the authors modeled the performance requirements as a fictitious uncertainty $\Delta_f(s)$. In doing so, the robust performance is shifted as a robust stability problem by letting:

$$\boldsymbol{\Delta}_{rp}(s) = \begin{bmatrix} \boldsymbol{\Delta}(s) & \mathbf{0} \\ \mathbf{0} & \Delta_f(s) \end{bmatrix} ; \tag{1.35}$$

with $\Delta_f(s) \in \mathcal{RH}_{\infty}$[5]. This scheme is depicted in Figure 11.

1.2.1 Robust control synthesis

After explaining the connection between the uncertainty system and the robust stability, the question is: How does the robust controller design work? The general representation of the upper LFT can be redesigned for robust control synthesis purposes. Again, $\boldsymbol{M}(s)$ can be an open-loop or closed-loop system [6]. If we consider a closed-loop, we denote the open-loop system as $\boldsymbol{P}(s)$ and the controller

[5] $\boldsymbol{\Delta}_f(s) \in \mathcal{RH}_{\infty}^{n_w\times n_z}$ with $n_w = n_z \neq 1$ for MIMO systems.

[6] If $\boldsymbol{M}(s)$ is considered as a closed-loop system, $\boldsymbol{M}(s)$ should be also not in $\mathcal{RH}_{\infty}$.

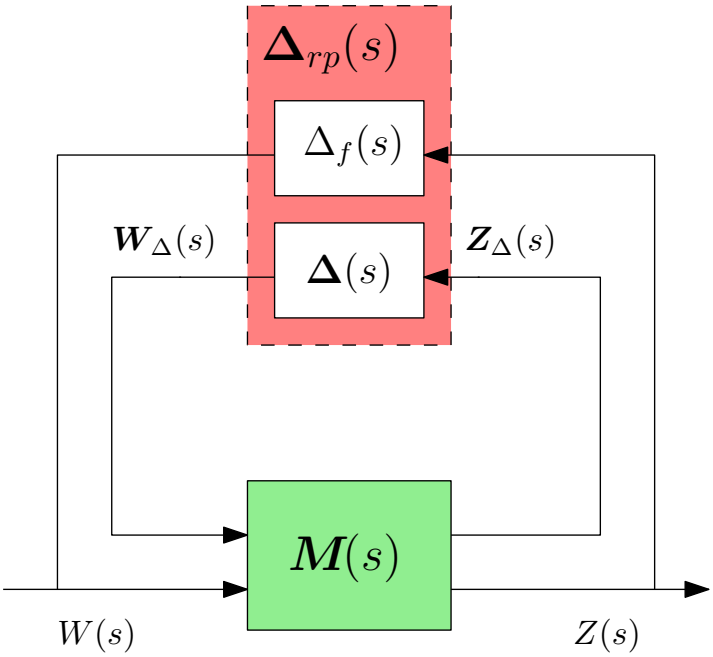


Figure 11: Robust performance represented by fictitious uncertainty for structured uncertainty systems.

$K(s)$[7], as illustrated in Figure 12. By grouping $\boldsymbol{P}(s)$ with the uncertainty block $\Delta(s)$, the LFT representation (Figure 12) can be considered as the classical control synthesis scheme, depicted in Figure 13. This grouped system is denoted as $\boldsymbol{N}(s) \in \mathcal{RH}_{\infty}^{2\times 2}$ (for SISO systems and unstructured uncertainty). The state space form (derived from (1.9)) of $\boldsymbol{P}(s)$ becomes:

$$\begin{cases} \boldsymbol{x}(t) &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}_1 w_{\Delta}(t) + \boldsymbol{B}_2 u(t) + \boldsymbol{B}_w w(t) \\ z_{\Delta}(t) &= \boldsymbol{C}_1 \boldsymbol{x}(t) + D_{11} w_{\Delta}(t) + D_{12} u(t) \\ y(t) &= \boldsymbol{C}_2 \boldsymbol{x}(t) + D_{21} w_{\Delta}(t) \\ z(t) &= \boldsymbol{C}_z \boldsymbol{x}(t) + D_z u(t) \end{cases}, \tag{1.36}$$

where:

- $w(t)$ is the exogenous input, such as disturbance, noise or reference signal;
- $z(t)$ is the exogenous output, which represents the system performance signal, for instance tracking error or controlled signal;
- $u(t)$ is the control input signal;
- $y(t)$ is the sensed output;
- unstructured uncertainty $\Delta(s)$;

[7]Therefore, $\boldsymbol{M}(s)$ represents transfer function of the group composed by $\boldsymbol{P}(s)$ and $K(s)$.

- $\boldsymbol{A} \in \mathbb{R}^{n_x \times n_x}, \boldsymbol{x} \in \mathbb{R}^{n_x \times 1}, \boldsymbol{B}_1 \in \mathbb{R}^{n_x \times 1}, \boldsymbol{B}_2 \in \mathbb{R}^{n_x \times 1}, \boldsymbol{B}_w \in \mathbb{R}^{n_x \times 1}, \boldsymbol{C}_1 \in \mathbb{R}^{1 \times n_x}, \boldsymbol{C}_2 \in \mathbb{R}^{1 \times n_x}, \boldsymbol{C}_z \in \mathbb{R}^{1 \times n_x} D_z \in \mathbb{R}, D_{11} \in \mathbb{R}, D_{12} \in \mathbb{R}, D_{21} \in \mathbb{R}$.

In case of structured uncertainty, the uncertainty is a matrix and the signals $w_\Delta(t)$ and $z_\Delta(t)$ become vectors, as already seen in Section 1.1.1.

Remark 1.3

In this setting, $\boldsymbol{P}(s)$ is coupled with a controller as represented in Figure 12, therefore $z(t) \neq y(t)$ and $w(t) \neq u(t)$.

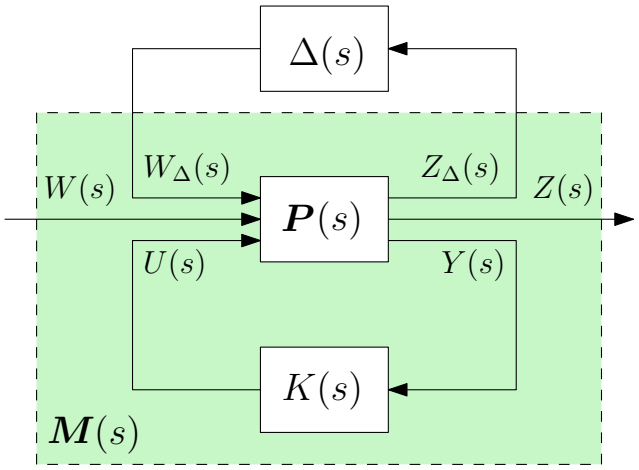


Figure 12: Linear Fractional Transformation for robust control synthesis.

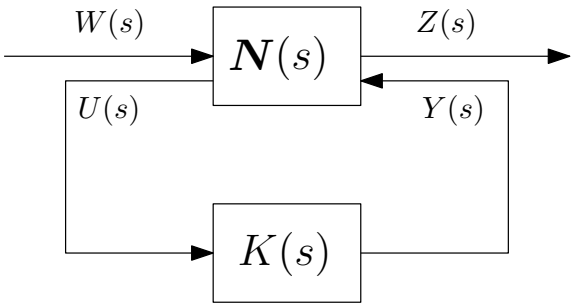


Figure 13: Traditional control synthesis scheme.

The robust control literature shows different methodologies to solve the control synthesis:

- $\mathcal{H}_2$ *optimal control*: this type of control design aims to regulate the transfer function between $W(s)$ and $Z(s)$, defined as the *lower Linear Fractional Transformation*:

$$\mathcal{F}_l(\boldsymbol{N}(s), K(s)) \equiv N_{11}(s) + N_{12}(s)K(s)(1 - N_{22}K(s))^{-1}N_{21}(s)\ . \tag{1.37}$$

  The robust controller is obtained by finding a stabilizing controller that minimizes the 2-norm of the lower LFT: $\|\mathcal{F}_l(\boldsymbol{N}(s), K(s))\|_2$. This optimization problem corresponds to minimize the total energy of the impulse response of $\mathcal{F}_l(\boldsymbol{N}(s), K(s))$. Given $w(t)$ as stationary noise, the $\mathcal{H}_2$ optimal control is often called *Linear Quadratic Gaussian (LQG) control* [39, Chapter 6]. Furthermore, $\mathcal{H}_2$ can be solved by two Riccati equations without iterative procedure and, doing so, the resulting controller is unique [25, Chapter 13].

- $\mathcal{H}_\infty$ *optimal control*: using the representation described with the $\mathcal{H}_2$ design, the robust controller is designed by finding a stabilizing controller that minimizes the infinity norm of the lower LFT:

$\|\mathcal{F}_l(\boldsymbol{N}(s), K(s))\|_\infty$[8]. As described in [40], finding an optimal controller with $\mathcal{H}_\infty$-norm is usually complicated. Therefore, this control design is translated into searching the suboptimal controllers that are closed, in norm sense, to the optimal one [25, 18, Chapter 14]. The *suboptimal $\mathcal{H}_\infty$ control* is defined as: given a $\gamma > 0$, find all stabilizing controllers $K(s)$, if there are any, such that $\|\mathcal{F}_l(\boldsymbol{N}(s), K(s))\|_\infty < \gamma$.

- $\mathcal{H}_\infty$ *loop-shaping*: this method adds a loop shaping design to the $\mathcal{H}_\infty$ optimization problem. Usually, to study this methodology the uncertainty is represented as the unweighted left coprime factorization. Thus, the design procedure corresponds to:

  1. Choose, by the user's knowledge, two weight functions $W_1(s)$ and $W_2(s)$, called pre-compensator and post-compensator. The aim of this consist of giving a desired open-loop shape to the system. Doing so, the shaped plant becomes $G_s(s) = W_2(s)G(s)W_1(s)$, where $G(s) = Y(s)/U(s)$ and it is defined as in (1.20);
  2. Solve the robust stabilization with the $\mathcal{H}_\infty$ optimization to obtain ν_{max}, using:

$$\nu_{max} = \left(\inf_{K \text{stabilizing}} \left\| \begin{bmatrix} K(s) \\ \\ 1 \end{bmatrix} \left(1 + G_s(s)K(s)\right)^{-1} D_{lcf}^{-1}(s) \right\|_\infty \right). \tag{1.38}$$

     Note that ν_{max} represents the stability margin. Therefore, if ν_{max} is not big enough, the final controller would be not compatible with the robust stability requirements. If this happens, we should redesign the pre and post-compensator.
  3. Select $\nu \leq \nu_{max}$ and synthesize a controller $K_\infty(s)$ by exploiting the robust stability test;
  4. The final controller is made as $K(s) = W_1(s)K_\infty(s)W_2(s)$.

  With this methodology, ν is used as a design indicator that measures both the closed-loop stability and the loop shaping specs, drawn by the pre and post-compensator.

  A more stringent method of $\mathcal{H}_\infty$ loop-shaping is *S/KS mixed sensitivity loop-shaping* [19]. This employs the weighted left coprime uncertainty model (1.22). Specifically, the weight functions $W_d(s)$ and $W_n(s)$ shape the robustness of $D_{lcf}(s)$ and $N_{lcf}(s)$. Often, $W_d(s)$ is also denoted as $W_S(s)$, while $W_n(s)$ as $W_Q(s)$. By grouping the left coprime terms and the weight functions together, as depicted in Figure 14, we obtain $\boldsymbol{H}(s) \in \mathcal{RH}_\infty^{2\times 1}$, i.e. the transfer function from

[8]The $\mathcal{H}_\infty$ norm corresponds to the worst case of the gain of $\mathcal{F}_l(\boldsymbol{N}(s), K(s))$.

$\boldsymbol{z}_\Delta = [z_{\Delta 1}, z_{\Delta 2}]^\top \in \mathbb{R}^{2\times 1}$ to w_Δ, where $\boldsymbol{\Delta}_{lfc}(s) = [\Delta_n(s), -\Delta_d(s)]$. It corresponds to:

$$\boldsymbol{H}(s) = \frac{W_\Delta(s)}{\boldsymbol{Z}_\Delta(s)} = \begin{bmatrix} W_d(s)S_0(s) \\ \\ W_n(s)Q_0(s) \end{bmatrix}; \tag{1.39}$$

where $Q_0(s) = K(s)S_0(s)$ is the control sensitivity function. Hence, the controller $K(s)$ is

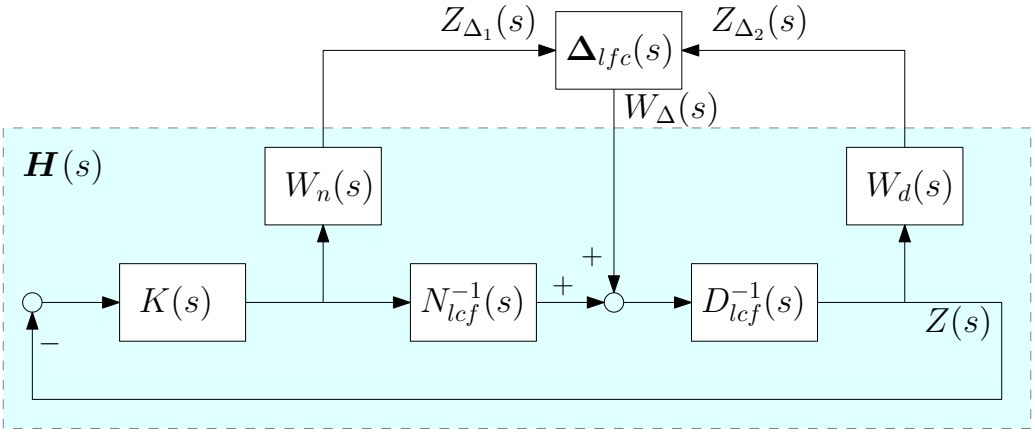


Figure 14: Graphic representation of left coprime factorization uncertainty for mixed-sensitivity loop-shaping [19].

designed with the $\mathcal{H}_\infty$ optimization of $\boldsymbol{H}(s)$[9]. In literature, there are also some variations of this mixed-sensitivity loop-shaping, such as: *S/T mixed-sensitivity loop-shaping* or *S/T/KS mixed-sensitivity loop-shaping* [16, Chapter 3]. In these alternatives, the weight functions represent some desired closed-loop shapes: sensitivity function, control sensitivity function or complementary sensitivity function. Furthermore, in these case the weight functions are denoted respectively as: $W_S(s)$, $W_Q(s)$ and $W_T(s)$ (see Section 2).

- *μ synthesis*: is a control synthesis that works with structured uncertainty (modeled as (1.12) or (1.5)). It minimizes the structured singular value of $\mathcal{F}_l(\boldsymbol{N}(s), K(s))$ [41] by solving the $\mathcal{H}_\infty$ optimization problem:

$$\hat{K}(s) = \min_K \inf_{D(s), D^{-1}(s)\in\mathcal{H}_\infty} \left\| D(s)\mathcal{F}_l(\boldsymbol{N}(s), K(s))D(s)^{-1} \right\|_\infty , \tag{1.40}$$

where inf is a standard convex optimization problem that can be solved pointwise in frequency domain. The lower LFT is scaled with a stable and minimum phase transfer function $D(s)$,

[9] If the designed controller $K(s)$ does not guarantees $\|\boldsymbol{H}(s)\|_\infty \leq 1$, then no controller exists that stabilizes the system for all perturbations $\boldsymbol{\Delta}$.

called *scaling transfer function*[10]. The entire optimization problem is solved iteratively by the *D-K iteration* technique [25, Chapter 10]. Specifically, the procedure is composed of: (i) minimize over $K(s)$ with a fixed $D(s)$, (ii) minimize pointwise over $D(s)$ with a fixed $K(s)$. Figure 15 represents the block scheme employed in the μ-synthesis controller design solved by the $D-K$ iteration procedure.

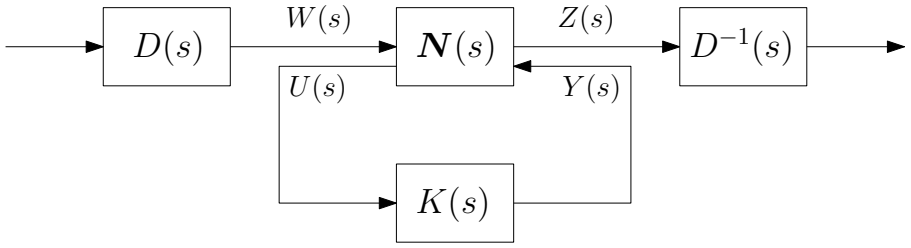


Figure 15: Graphical representation of the μ-synthesis employing the scaling transfer function.

- *Mixed $\mathcal{H}_2/\mathcal{H}_\infty$ or Mixed $\mathcal{H}_\infty/\mathcal{H}_\infty$ optimal control*: these methods are used with MIMO systems. They split the system $\boldsymbol{N}(s)$ into two parts, such as $\boldsymbol{N}_1(s)$ and $\boldsymbol{N}_2(s)$. In this way, the relationship between the exogenous input $\boldsymbol{w}_1(t)$ and the exogenous output $\boldsymbol{z}_1(t)$ (which belong to the subsystem $\boldsymbol{N}_1(s)$) are treated differently from the signals $\boldsymbol{w}_2(t)$ and $\boldsymbol{z}_2(t)$ (which belong to the subsystem $\boldsymbol{N}_2(s)$). The general optimization problem that solves these two types of controller design is:

$$\begin{aligned} \min \quad & \left\| \mathcal{F}_{l_{\boldsymbol{w}_1 \to \boldsymbol{z}_1}}(\boldsymbol{N}_1(s), \boldsymbol{K}(s)) \right\|_a \\ \text{s.t.:} \quad & \left\| \mathcal{F}_{l_{\boldsymbol{w}_2 \to \boldsymbol{z}_2}}(\boldsymbol{N}_2(s), \boldsymbol{K}(s)) \right\|_\infty < \gamma \quad , \\ & \boldsymbol{K}(s) \text{ stabilize internally } \boldsymbol{N}(s) \end{aligned} \tag{1.41}$$

  where $a = 2, \infty$ [42].

For completeness, in literature, there are other control approaches for uncertain systems, such as: *game-theoretic* or *minimimax* [43] , *guaranteed-cost control* [44], *norm uncertainty* [45], *quantitative-feedback theory* [46], the new *polynomial* and *probabilistic techniques* [34].

Often, in robust control design, the resulting controller has high order, typical comparable with the highly complex uncertain system under analysis. Therefore, for practical purposes, order reduction is widely used since a low-order controller is easily understandable and implementable in a real process. Another possible way to solve this problem consists of: identifying a low-order model and then

[10]With MIMO systems, the choice of the scaled transfer function must agree with $\boldsymbol{D}(s)\boldsymbol{\Delta}(s) = \boldsymbol{\Delta}(s)\boldsymbol{D}(s)$.

synthesizing a low-order controller using the identified model. However, this methodology does not guarantee that the controller designed with the low-order model stabilizes also the full-order plant. Hence, the authors of [25, Chapter 19] report some techniques to reduce the controller order without losing the robust stability property of the high order controller. Another novel way, proposed in [47], corresponds to solving the $\mathcal{H}_\infty$ loop-shaping by adding a constraint that selects the structure of the controller. This is more conservative than the free control structure synthesis, but for real application development, it is more practical.

1.3 Uncertainty for robust fault diagnosis

This section briefly reviews some fundamentals of the fault diagnosis literature and its taxonomy, with focus on model-based fault diagnosis. Specifically we describe both robust and not robust residual generation. A *fault* is a not permitted deviation of at least one characteristic of a system, from the acceptable, usual, standard condition. A fault causes a malfunction or a failure. The *malfunction* is an intermittent irregularity in the fulfillment of a system's function. Instead, the *failure* is a permanent interruption of a component or of the entire system. Note that the difference between the failure and the malfunction is represented by the time of the interruption. Figure 16 recaps the distinction between the fault and failure/malfunction definitions.

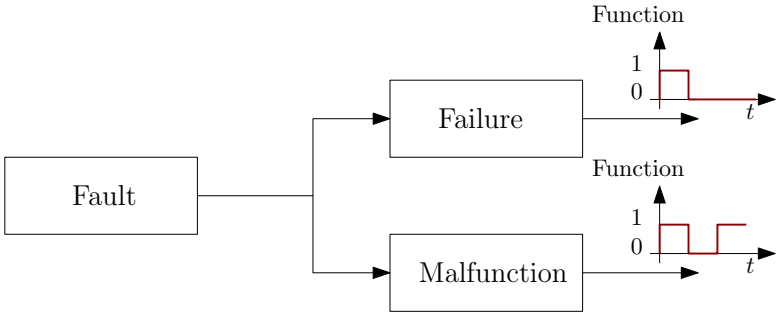


Figure 16: Fault evolution scheme [48, 49, chapter 2].

Figure 17 depicts three different approaches to fix faults. The upper and middle methods were applied before the advent of industry 4.0. In particular, the *Reactive maintenance* perspective fixes malfunctions/failures only when they occur. This method causes a large waste of time. Instead, with *Preventive maintenance*, the factories plan a time-based maintenance schedule. It prevents malfunctions/failures occurrence, but it is not optimal, because this program is usually more conservative than the machinery necessities. Instead, *Predictive maintenance* exploits some algorithms that generate alarms before the fault occurrence. In this way, the maintenance intervention is done only when the system needs it. The definition of Fault diagnosis is defined as:

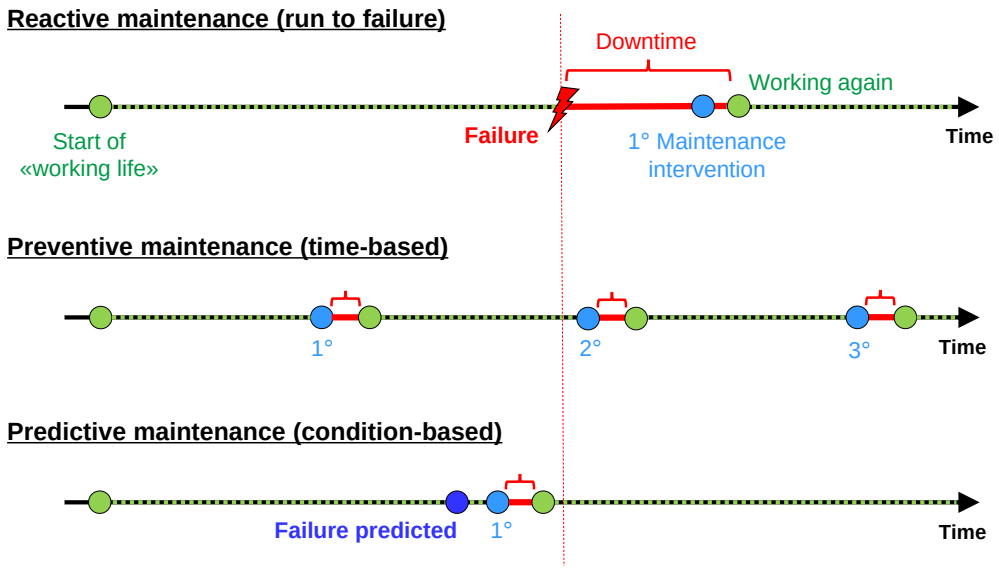


Figure 17: Representation of some types of maintenance strategies.

Definition 1.7: Fault diagnosis [29, Chapter 3]

Fault diagnosis refers to the usage of techniques to evaluate the status of a system with respect to possible faults.

Fault diagnosis methods perform the following tasks:

- *Fault detection*: indicates if a fault arise or not. It also determines the time of the fault occurrence;
- *Fault isolation*: determines the location of the fault;
- *Fault analysis or identification*: estimates the size and nature of the detected fault;
- *Fault estimation*: reconstructs the behavior of the fault.

The list is ordered by complexity, from the lowest to the highest. The first task is mandatory to implement a fault diagnosis algorithm, while the others are optional. Often, the last two are used as synonyms. Notice that the nomenclature is not consistent across the kinds of literature, such as for industry [50, 51], for control system community [52] [27, Chapter 3] [53, Chapter 1].
Figure 18 [29, Chapter 3] reports the scheme of the fault diagnosis taxonomy, specifically:

- *Hardware redundancy*: which provides the physical replication of the critical components of the system, for example: actuators, sensors or even software. These components are in parallel with the standard system, they are fed by the same input, but they work only when a fault occurs.

This is useful for those systems that work in a critical environment, as in electric aircraft [54] or nuclear power plants [55].

- *Analytical redundancy*: which solves the fault diagnosis problem by feeding the input and output signals of the critical system to an algorithm that predicts the status of the considered system. It is less expensive than the hardware redundancy. Usually, the algorithm compares the healthy state with respect to the state of the working system. The healthy state has different representations which depend on the employed algorithm, e.g.: white box model, black box model, a priori known signal pattern, signal behaviors, a priori known information, and so on. As depicted in Figure 18, the analytical redundancy scheme has a broad range of methodologies. They can be grouped in [28, Chapter 1]:

  1. *Plausibility tests*: the fault diagnosis is done by checking some physical laws that govern the system. If the machinery does not pass these tests, it is said that the system lose its plausibility and therefore it is faulty;
  2. *Model-based*: the healthy system is modeled by a mathematical model. The model-based procedure is mainly composed of two steps: *residual generation* and *residual evaluation* [56]. A *residual* is a signal generated by processing the input and output data with the employment of the mathematical model (during the residual generation step). The evaluation procedure decides if there is a fault or not by studying the behaviour of the residual signal. Ideally, if the residual signal differs from zero, a fault occurs;
  3. *Signal-based*: assumes that the signals carry information about the state of the system. The fault diagnosis compares some features, called *symptoms*, extracted from the acquired signals, with respect to a priori known values which represent the healthy state;
  4. *Knowledge-based*: assumes that a large amount of historical data is available. The user does not know any prior knowledge or behaviour of the system. The only thing that he or she can do is to learn some knowledge from the historical data and consequentially evaluate the data acquired online by exploiting the extrapolated information. These two phases are called: *training* and *online evaluation*;
  5. *Hybrid*: is a combination of two or more methods: signal-based, model-based, plausibility test and knowledge-based. Doing so, the resulting approach takes the benefits of the considered algorithms;
  6. *Active*: assumes that the input can be chosen and injected, thereby the fault can be detected more easily.

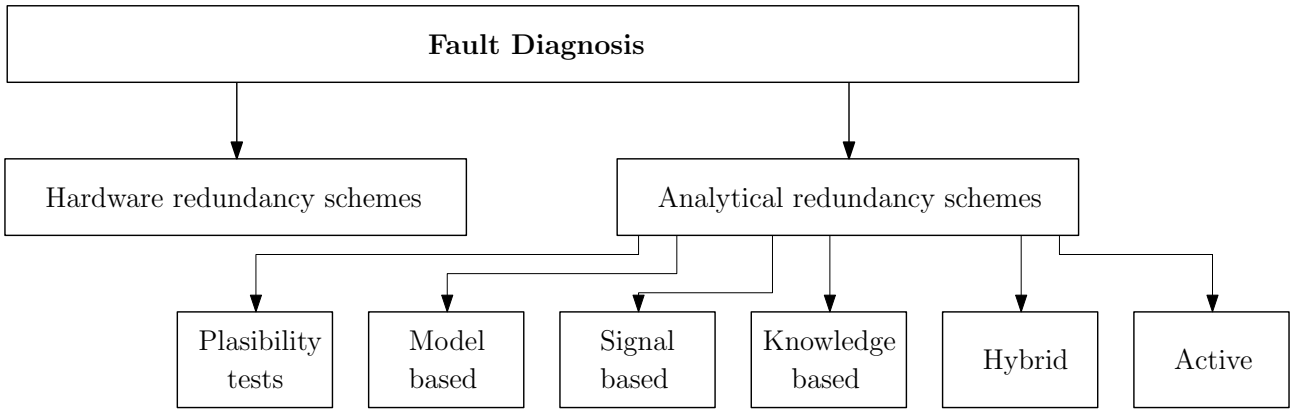


Figure 18: Representation of the taxonomy of the fault diagnosis algorithm.

The most widespread approaches are: model-based and signal-based. Specifically, the former is used with dynamic processes and the latter is employed with steady-state or complex processes. In light of this, those are reviewed deeply, with more focus on model-based.

Figure 19 depicts the signal-based fault diagnosis scheme. From the output signal, the algorithm generates some features called *fault symptoms*. The fault diagnosis is done by performing an analysis of the symptoms. In particular, the measured fault symptoms are compared with a priori known fault symptoms values which represent the healthy state. Signal-based algorithms are grouped according to the domain to which their symptoms belong [57], such as:

- *Time domain approach*: computes the statistics of the signal in the time domain, such as: mean, variance, peak to peak, Root Mean Square, kurtosis, crest factor and so on. For instance, these types of symptoms are used with gear fault diagnosis [58], power converters of switched reluctance motors [59] and permanent magnet synchronous generators in wind turbine applications [60];

- *Frequency domain approach*: obtains a more clear representation of the symptoms in frequency domain and thus designs an efficient symptoms analysis. Nevertheless, it is also possible to compute some statistical features employed in the time-domain approach, such as: Root Mean Square, mean and standard deviation. A famous tool that belongs to this methodology is Motor-Current Signature Analysis (MCSA). It is used to diagnose the broken rotor bars of an electrical motor by performing the spectral analysis of the stator current [61]. Another well-known approach, described in [62], is vibration analysis for gearbox faults and bearing faults. Note that, acoustic signals are also employed with this approach since they are also a vibrational signals. The authors of [63] propose an example of frequency domain signal-based fault diagnosis based on acoustic signals;

- *Time-Frequency domain approach*: this approach is useful with transient dynamic conditions. These variable situations cause a time-varying frequency spectrum. The time-frequency analysis identifies the signal frequency components and reveals their time-varying characteristics. Some methodologies that employs this approaches are: STFT [64], WT [65], HHT [66] and WVD [67].

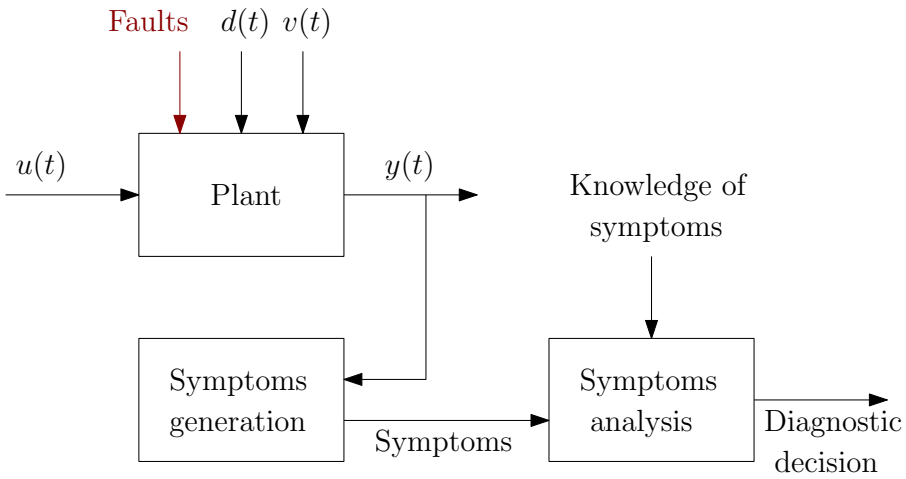


Figure 19: Representation of the general signal-based scheme [28, Chapter 1].

The definition of model-based fault diagnosis is the determination of faults through the comparison of the available system measurements with respect to a priori information provided by a mathematical model [20, Chapter 1]. The model acts as a *digital twin* of the health system. Figure 20 represents the overall scheme of the model-based fault diagnosis. As already said, the two major steps of these techniques are: residual generation and residual evaluation. The residual generator produces the *residual signal* $r(t)$ by processing the measured input $u(t)$ and output $y(t)$ signals. The signal $r(t)$ contains the information useful to diagnose the system state. The residual evaluation strategy helps to enhance the fault information by performing some signal processing strategies. After that, during the decision logic step, the processed residual $\theta(t)$ is compared with respect to a selected threshold. The popular approaches of this methodology are: deterministic fault diagnosis and stochastic fault diagnosis. In literature, there are also discrete events and hybrid, networked and distributed methods, but they are less widespread.

The deterministic branch replaces the process model with a deterministic model. Specifically, it can be described by: *stable coprime factorization*, *observers* scheme and *parity relations*. The stochastic approach models the process with a stochastic method, for instance: *Kalman filter* and *Parameters estimation*. Ideally, the Kalman filter method is the stochastic counterpart of the observers. With this setting, the changes in the distribution of the residuals represent the fault symptoms. Often,

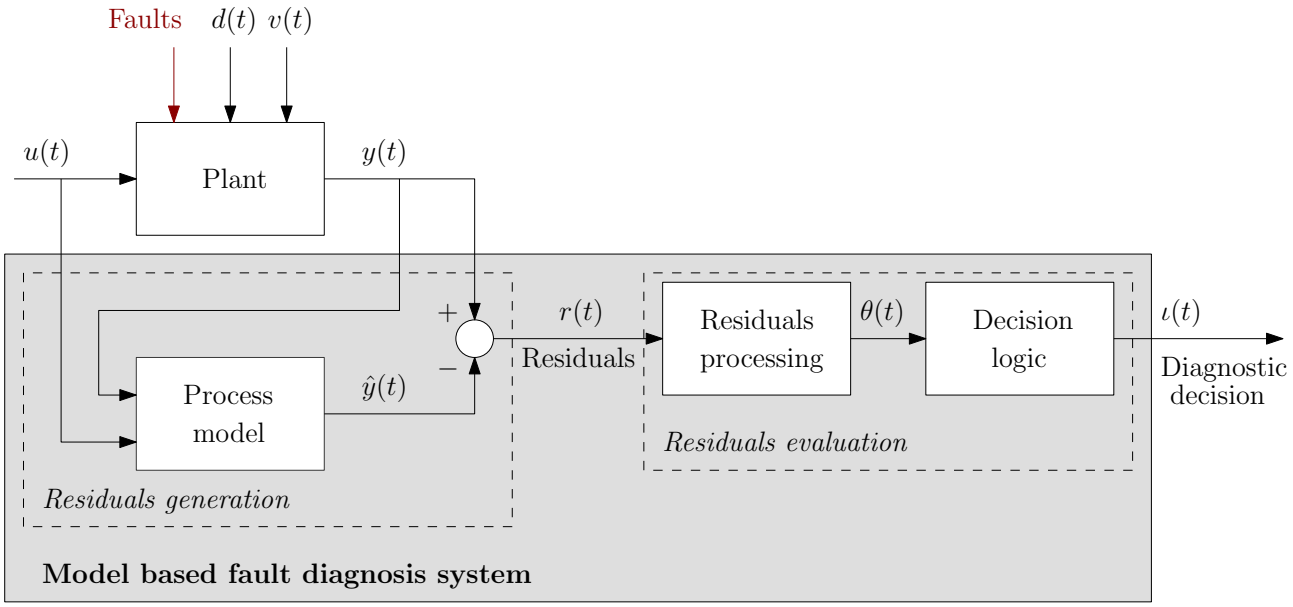


Figure 20: Representation of the general model-based scheme [28, Chapter 1].

the residual evaluation is made by some statistical tests, such as: χ^2 test, cumulative sum algorithm or multiple hypothesis test. Furthermore, there exist also some extensions of the Kalman filter that account the system non-linearities [68]. In [69], the authors have proposed a residual evaluation based on the Generalized Likelihood Ratio coupled with a Kalman residual generator.

The parameter estimation, introduced by [70], performs the system identification, which estimates online the model parameters. The rationale is that the variation of system parameters reflects the faults. If the model parameters have an explicit mapping in the physical coefficients, this procedure can be very efficient and simple, but usually the parameter mapping is difficult to obtain, as described in [71]. Furthermore, the fault evaluation is done by verifying the variations in the parameters vector against the estimated parameters under healthy conditions [72].

The next subsections describe briefly the deterministic approaches with SISO system and single-fault, but as before the concepts are easily scalable for MIMO systems and multiple faults detections.

1.3.1 Stable coprime factorization

This technique relies on the *coprime factorization* [28, Chapter 3] of a transfer function, defined as:

Definition 1.8: Coprime factorization

Two discrete[a] transfer functions $\hat{M}(z) \in \mathcal{RH}_\infty \quad \hat{N}(z) \in \mathcal{RH}_\infty$ are called left coprime over $\mathcal{RH}_\infty$ if there exist another two transfer functions $\hat{X}(z) \in \mathcal{RH}_\infty$ and $\hat{Y}(z) \in \mathcal{RH}_\infty$ such that:

$$\begin{bmatrix} \hat{M}(z) & \hat{N}(z) \end{bmatrix} \begin{bmatrix} \hat{X}(z) \\ \\ \hat{Y}(z) \end{bmatrix} = 1 \ . \tag{1.42}$$

By letting $A(z)$ be a proper real-rational transfer function, the **Left Coprime Factorization (LFC)** of $A(z)$ results into two stable and coprime transfer functions. Similarly, there exists a counterpart of the LFC called **Right Coprime Factorization (RFC)**. It is defined as:

$$\begin{bmatrix} X(z) & Y(z) \end{bmatrix} \begin{bmatrix} M(z) \\ \\ N(z) \end{bmatrix} = 1 \ . \tag{1.43}$$

Where the transfer functions $M(z), N(z), Y(z), X(z)$ have the same properties as with the LFC. The relation that ties the model of the system $A(z)$ with the left coprime pair $\left(\hat{M}(z), \hat{N}(z)\right)$ is:

$$A(z) = \hat{M}^{-1}(z)\hat{N}(z) \ . \tag{1.44}$$

Instead, the relation with the right coprime pair $\Big(M(z), N(z)\Big)$ becomes:

$$A(z) = N^{-1}(z)M(z) \ . \tag{1.45}$$

The coprime factorization has also the state space representation. In the robust control theory, the RFC has a feedback control interpretation, while the LFC has an observer interpretation [28, Chapter 3].

[a] This definition is valid also for continuous transfer functions.

By considering the estimation error $r(t) = y(t) - \hat{y}(t)$ as the residual signal, the residual generator is $\begin{bmatrix} \hat{M}(z) & -\hat{N}(z) \end{bmatrix}$, computed from the system model $G(z)$. Therefore, $r(t)$ is computed by:

$$r(t) = \begin{bmatrix} \hat{M}(z) & -\hat{N}(z) \end{bmatrix} \begin{bmatrix} y(t) \\ \\ u(t) \end{bmatrix} . \tag{1.46}$$

In literature, this residual generator is also called *kernel representation* of the system. Usually, if $r(t)$ is equal to zero then the system is healthy, instead, if $r(t)$ differs from zero then the system has a fault. This consideration is only ideal because, in real applications, disturbances and uncertainties arise, therefore this fault diagnosis scheme can reveal false alarms. This problem is addressed by the robust residual generator, which generates the residual signal by decoupling disturbances and uncertainties. Before proceeding to the analysis of the robust residual generator counterpart, the modeling of faults, disturbance and uncertainties, in the fault diagnosis literature, must be faced.
The disturbances $d(t)$ and noise $v(t)$, which afflict the process under analysis, are usually designed as unknown input vectors. Thus, the input-output model in the healthy state is:

$$y(t) = G_0(z)u(t) + G_d(z)d(t) + G_v(z)v(t) \; ; \tag{1.47}$$

where: $G_0(z)$ is the model of the system, $G_d(z)$ is a known transfer function from disturbance to output, $G_v(z)$ is a known transfer function from noise to output, $d(t) \in \mathbb{R}$ is the deterministic unknown input signal, $v(t) \sim \mathcal{N}(0, \sigma_v)$ is a noise signal or represents a *fictitious noise* which models the uncertainties [27, Chapter 2]. Hence, the input-output model in faulty conditions is:

$$y(t) = G_0(z)u(t) + G_d(z)d(t) + G_v(z)v(t) + G_f(z)f(t) \; ; \tag{1.48}$$

where: $f(t) \in \mathbb{R}$ is an unknown signal that represents the fault, while $G_f(z)$ is a known transfer function from fault to output.
The taxonomy of the fault is:

- *Actuator fault* : which is a fault that acts on the actuator component;
- *Sensor fault* : which is a fault that acts on the sensor component;
- *Process fault*: which is a fault that acts directly on the process.

The sensor fault is considered as an additive term, so the transfer function from output to fault is set as $G_f(z) = 1$. Instead, with a process or actuator fault, the fault term is multiplicative and thus

$G_f(z) = G_0(z)$. Note that the additive term does not affect the system stability, while the multiplicative term can cause system instability. As introduced in [28, Chapter 3], the multiplicative fault can be readapted as an additive fault. For this reason, a fault is often represented by an additive term.
The residual generation, described with (1.46), does not decouple the model uncertainties, therefore the latter input-output model needs of a further filter, called *post-filter* $Q(z) \in \mathcal{RH}_\infty$, which makes robust the residual generator with respect to the uncertainties. Figure 21 represents the robust residual generation obtained by the stable coprime factorization scheme, where $G_T(z)$ is the true system. It corresponds to write:

$$r(t) = Q(z)\left(\hat{M}(z)y(t) - \hat{N}(z)u(t)\right) . \tag{1.49}$$

This equation is called *implementation form* and it has an equivalent and alternative form called *internal form*. The former is used to generate the residual signal online, instead the latter is used to design the post-filter. The internal form is equal to:

$$r(t) = Q(z)\hat{M}(z)G_v(z)v(t) + Q(z)\hat{M}(z)G_f(z)f(t) + Q(z)\hat{M}(z)G_d(z)d(t) . \tag{1.50}$$

The proof of the equivalence between the internal and the implementation forms are reported in the following.

Proof 1.1: Internal form and implementation form are equivalent

By substituting the input-output model (1.48) into the implementation form (1.49) [28, Chapter 5], we get:

$$r(t) = Q(z)\Big(\hat{M}(z)\Big(G_0(z)u(t) + G_d(z)d(t) + G_v(z)v(t) + G_f(z)f(t)\Big) - \hat{N}(z)u(t)\Big) . \tag{1.51}$$

Then, by applying the coprime factorization $G_0(z) = \hat{M}(z)^{-1}\hat{N}(z) \rightarrow \hat{N}(z) = G_0(z)\hat{M}(z)$, the equation (1.51) becomes:

$$r(t) = Q(z)\Big(\cancel{\hat{M}(z)G_0(z)u(t)} + \hat{M}(z)\Big(G_d(z)d(t) + G_v(z)v(t) + G_f(z)f(t)\Big) - \cancel{G_0(z)\hat{M}(z)u(t)}\Big) . \tag{1.52}$$

Thus, the internal form (1.49) is obtained.

Post filter design goals are to decouple the residual signal from $u(t)$ and $d(t)$. These requirements are called *decoupling conditions*. Note that, with the presented formulation, the decoupling of $u(t)$ is automatically obtained, indeed in (1.49) the residual signal does not depend on $u(t)$. So the latter condition is:

$$G_d(z)\hat{M}(z)Q(z) = 0 . \tag{1.53}$$

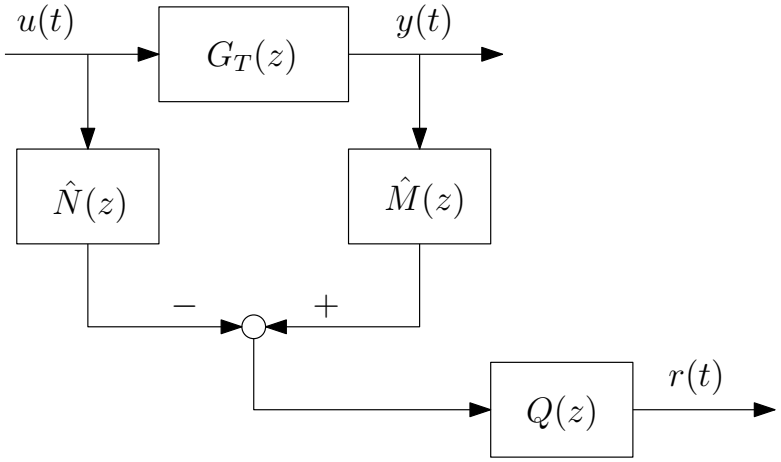


Figure 21: Representation of the robust residual generator scheme with stable coprime factorization.

The author in [27] have described two different methods to design $\tilde{\boldsymbol{Q}}(z) = \left[\hat{Q}(z)\hat{M}(z), -\hat{Q}(z)\hat{N}(z)\right]$:

- *Exact Fault Detection Problem (EFDP)*: which develops a suitable and stable $Q(z)$ in absence of $v(t)$, such that:
  - The residual is sensitive to fault;
  - The decoupling conditions are hold;
  - The detection condition is $G_f(z)\hat{M}(z)Q(z) \neq 0$;
  - $G_f(z)\hat{M}(z)Q(z)$ is stable.
- *Approximate Fault Detection Problem (AFDP)*: which develops a suitable and stable $Q(z)$ such that:
  - The residual is sensitive to a fault and noise input because $v(t)$ can not be fully decoupled but negligible;
  - The decoupling conditions are hold;
  - The detection condition are $G_f(z)\hat{M}(z)Q(z) \neq 0$ and $G_v(z)\hat{M}(z)Q(z) \approx 0$;
  - $G_f(z)\hat{M}(z)Q(z)$ and $G_v(z)\hat{M}(z)Q(z)$ are stable.

  Since $Q(z)$ ensures that the contribution of the disturbances and the control input are null, the implementation form (1.49) in the case of AFDP becomes:

$$r(t) = Q(z)\hat{M}(z)G_v(z)v(t) + Q(z)\hat{M}(z)G_f(z)f(t)\ . \tag{1.54}$$

The cost function for designing $Q(z)$ is defined as [27, 20, Chapter 3]:

$$\hat{Q}(z) = \max_{Q(z)} \frac{\left\| Q(z)\hat{M}(z)G_f(z) \right\|_a}{\left\| Q(z)\hat{M}(z)G_v(z) \right\|_a} ; \tag{1.55}$$

where $a = 2, \infty$. Doing so, the post filter enhances the sensibility of the residual signal $r(t)$ with respect to the fault signal and reduces the influences of the fictitious noise in the residual signal. A similar robust fault residual generator that solves AFDP is derived from the $\mathcal{H}_\infty$ theory, defined as:

$$\hat{Q}(z) = \min_{Q(z)} \frac{\|r(t) - f(t)\|_2}{\|v(t)\|_2} . \tag{1.56}$$

These two optimization procedures take also the name $\mathcal{H}_\infty$ *robust fault diagnosis* [73].

There exists a variation of (1.56) where the $f(t)$ signal is replaced by $r_o(t) = R(z)f(t)$, where $r_o(t)$ represents the desired behavior of the residual signal, modeled by $R(z)$[11]. This problem is solved as a robust controller synthesis. Doing so, the resulting controller is considered as the residual generator, as depicted in Figure 12. The input is composed of $w(t) = \left[f(t), d(t), v(t) \right]^\top$, respectively: fault, noise and input signals. Instead, the output of the general system is $z(t) = r_o(t) - r(t)$ and the output of the filter $Q(z)$ (that replaces $K(z)$) is the residual signal $r(t)$ [74]. With this representation, the cost function becomes $J = \sup_{w(t)} \frac{\|z(t)\|_2}{\|w(t)\|_2} = \|G_{zw}(z)\|_\infty$, thus the filter design can be done by μ or $\mathcal{H}_\infty$ synthesis. This method does not work with all applications: therefore, two functions that weight the signals $w(t)$ and $z(t)$ are designed to avoid this problem. In [75, 76], the authors have proposed an extension of this approach by coupling the filter $Q(s)$ with a controller $K(z)$, where the general output is composed of $r_o(t) - r(t)$ and a performance signal, denoted by $y_e(t)$. This scheme is depicted in Figure 22. The choice of the $Q(z)$ structure and weight functions are critical and done by the user[12].

1.3.2 Observer schemes

The second scheme exploits the so-called *output observer*. Notice that the output observer differs from the state observer employed in the control literature, nevertheless, in the history usually are considered as the same object, but as reported in [28, Chapter 2] this is a mistake. Furthermore, the state observer is not always applicable since it assumes that all states $\boldsymbol{x}(t)$ are measurable. Thus, the output observer is useful to perform fault diagnosis by comparing the measured output with respect to the observed output. In the last three decades, the observer schemes have achieved certain popularity for being able to decouple residual signals from certain disturbances and modelling errors (uncertainties). This robust method is called *Unknown input observer* [77].

[11] The structure of $R(z)$ must contain the filter $Q(z)$.

[12] $R(z)$ can be derived from $Q(z)_{nom}$, which represents the filter computed with the nominal model [73, Chapter 6].

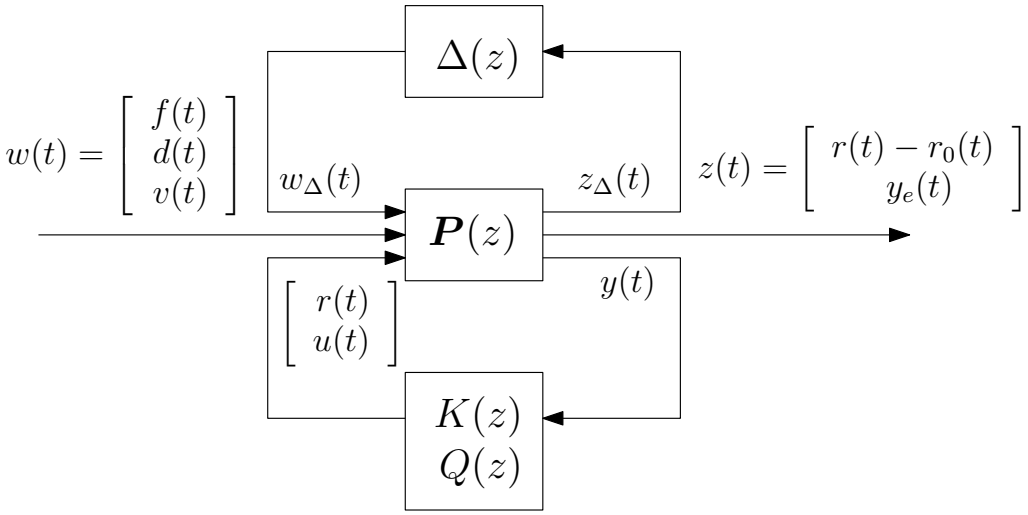


Figure 22: Representation of the robust control scheme for robust residual generator synthesis with stable coprime factorization.

The precursor of the observer-based residual generation is the *Fault Detection Filter*, proposed by [78, 79]. The well-known full-order state observer is defined as:

$$\begin{aligned} \hat{\boldsymbol{x}}(t) &= \boldsymbol{A}\hat{\boldsymbol{x}}(t) + \boldsymbol{B}u(t) + \boldsymbol{L}\Big(y(t) - \boldsymbol{C}\boldsymbol{x}(t) - Du(t)\Big) \\ \hat{y}(t) &= \boldsymbol{C}\hat{\boldsymbol{x}}(t) + Du(t) \end{aligned}, \tag{1.57}$$

with $\boldsymbol{A} \in \mathbb{R}^{n_x \times n_x}, \boldsymbol{B} \in \mathbb{R}^{n_x \times 1}, \boldsymbol{C} \in \mathbb{R}^{1 \times n_x}, D \in \mathbb{R}$ and $\boldsymbol{L} \in \mathbb{R}^{n_x \times 1}$ is a suitable chosen matrix that ensures a limited difference between the real output $y(t) \in \mathbb{R}$ and the estimated $\hat{y}(t) \in \mathbb{R}$. The symbol $\hat{\boldsymbol{x}}(t) \in \mathbb{R}^{n_x \times 1}$ denotes the estimated states. The observer acts as a feedback loop that ensures:

- $r(t) = y(t) - \hat{y}(t) = y(t) - \boldsymbol{C}\hat{\boldsymbol{x}}(t) - Du(t) = 0 \quad \forall u(t)$;
- Unbiased estimation $\lim_{t \to \infty} \Big(x(t) - \hat{x}\Big) = 0$, with $\boldsymbol{A} - \boldsymbol{L}\boldsymbol{C}$ stable.

The matrix $\boldsymbol{L}$ is the only degree of freedom of this procedure, indeed usually the output estimation error is weighted with a designed weight V, as:

$$\boldsymbol{r}(t) = V\Big(y(t) - \hat{y}(t)\Big). \tag{1.58}$$

The choice of $\boldsymbol{L}$ and V must be done to ensure the stability of the full-order observer.
The advantage of this method consists of the simplicity of implementation, but the drawback is due to the difficulty to develop the full-order observer in an online application. This is the reason why a lot of work has been done to propose an evolution of the Fault detection filter, called *Diagnostic Observer*.

The construction of the diagnostic observer-based residual generator is well-known and explicated in [28, Chapter 5]. Specifically, the procedure is quite similar, but by employing a Luenberger output observer a reduced order observer is obtainable [80]. The Diagnostic Observer offers a greater degree of design freedom than the older Fault Detection Filter.

Remark 1.4

The system composed by (1.57) and (1.58) can be seen as a state space representation of (1.46).

As for the previous scheme, we also present the robust counterpart of the Fault detection filter approach. It models disturbances, time-varying term and parameter variation, noise, non-linear terms and model reduction error with the additive terms $\boldsymbol{E}_v v(t)$[13], as:

$$\begin{aligned} \boldsymbol{x}(t) &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}u(t) + \boldsymbol{E}_v v(t) + \boldsymbol{E}_f f(t) \\ & \qquad ; \\ y(t) &= \boldsymbol{C}\boldsymbol{x}(t) + Du(t) + F_f f(t) \end{aligned} \tag{1.59}$$

where the terms $F_f f(t)$ and $\boldsymbol{E}_f f(t)$ ($\boldsymbol{E}_f \in \mathbb{R}^{n_x \times 1}$) represent the impact of the fault on the system. By employing the Fault Detection filter ((1.57) and (1.58) equations), the Unknown input decoupling is obtained by a suitable choice of V and $\boldsymbol{L}$, such as:

- The Fault Detection Filter is stable;
- $V\boldsymbol{C}\left(s\boldsymbol{I}_{n_x} - \boldsymbol{A} + \boldsymbol{L}\boldsymbol{C}\right)^{-1}\boldsymbol{E}_v = 0$ is guaranteed;
- $V\left(\boldsymbol{C}(s\boldsymbol{I}_{n_x} - \boldsymbol{A} + \boldsymbol{L}\boldsymbol{C})^{-1}(\boldsymbol{E}_f - \boldsymbol{L}F_f) + F_f\right) \neq 0$ is hold.

Two algorithms are studied to solve this design problem: *eigenstructure assignment* and *geometric approach*. The former is explained in [22]. It develops a linear state space feedback system by exploring its: eigenvalues, left and right eigenvectors. Instead, the geometric approach seeks $\boldsymbol{L}$ which makes $(\boldsymbol{A} - \boldsymbol{L}\boldsymbol{C}, \boldsymbol{E}_v, \boldsymbol{C})$ maximally uncontrollable[14] by $v(t)$ [81]. There exist also the *Unknown Input Diagnostic Observer* [77] that shares the same goal with the Unknown Input Fault Detection Filter with a reducer order observer, such for the not robust counterpart.

1.3.3 Parity equation schemes

The parity space framework has the peculiarity of presenting the residual signals in the form of algebraic equations, therefore the solutions can be achieved with linear algebra tools. By considering

[13] This representation does not consider the influence of the noise applied to the input.

[14] Maximally uncontrollable means the uncontrollable subspace with the maximal dimension.

the state space model (1.59), with: $v(t) = f(t) = 0$ and $b \geq 0$, the output can be seen as:

$$\begin{aligned} y(t-b) &= \boldsymbol{C}\boldsymbol{x}(t-b) + Du(t-b) \\ y(t-b+1) &= \boldsymbol{C}\boldsymbol{A}\boldsymbol{x}(t-b) + \boldsymbol{C}\boldsymbol{B}u(t-b) + Du(t-b+1) \\ y(t-b+2) &= \boldsymbol{C}\boldsymbol{A}^2\boldsymbol{x}(t-b) + \boldsymbol{C}\boldsymbol{A}\boldsymbol{B}u(t-b) + \boldsymbol{C}\boldsymbol{B}u(t-b+1) + Du(t-b+2) \end{aligned} \quad . \tag{1.60}$$

By generalizing, it becomes:

$$y(t) = \boldsymbol{C}\boldsymbol{A}^b\boldsymbol{x}(t-b) + \boldsymbol{C}\boldsymbol{A}^{b-1}\boldsymbol{B}u(t-b) + ... + \boldsymbol{C}\boldsymbol{B}u(t+1) + Du(t) \; . \tag{1.61}$$

It corresponds to write:

$$\boldsymbol{y}_b(t) = \boldsymbol{H}_{o,b}\boldsymbol{x}(t-b) + \boldsymbol{H}_{u,b}\boldsymbol{u}_b(t) \; ; \tag{1.62}$$

where:

$$\boldsymbol{y}_b(t) = \begin{bmatrix} y(t-b) \\ y(t-b+1) \\ \vdots \\ y(t) \end{bmatrix}, \qquad \boldsymbol{u}_b(t) = \begin{bmatrix} u(t-b) \\ u(t-b+1) \\ \vdots \\ u(t) \end{bmatrix} \tag{1.63}$$

$$\boldsymbol{H}_{o,b} = \begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{C}\boldsymbol{A} \\ \vdots \\ \boldsymbol{C}\boldsymbol{A}^{\boldsymbol{b}} \end{bmatrix}, \qquad \boldsymbol{H}_{u,b} = \begin{bmatrix} D & 0 & \dots & 0 \\ \boldsymbol{C}\boldsymbol{B} & D & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ \boldsymbol{C}\boldsymbol{A}^{b-1}\boldsymbol{B} & \dots & \boldsymbol{C}\boldsymbol{B} & D \end{bmatrix}$$

with: $\boldsymbol{y}_b(t) \in \mathbb{R}^{(b+1)\times 1}, \boldsymbol{u}_b(t) \in \mathbb{R}^{(b+1)\times 1}, \boldsymbol{H}_{o,b} \in \mathbb{R}^{(b+1)\times n_x}, \boldsymbol{H}_{u,b} \in \mathbb{R}^{(b+1)\times(b+1)}$. Notice that the equation (1.62) is called *parity relation*, which describes the input-output relationship by exploiting

the past state vector $\boldsymbol{x}(t-b)$. Furthermore, $\boldsymbol{H}_{o,b}$, $\boldsymbol{H}_{u,b}$, $\boldsymbol{y}_b(t)$ and $\boldsymbol{u}_b(t)$ are known, therefore the only unknown vector is $\boldsymbol{x}(t-b)$. For $n_x \leq b$, where n_x is the number of states, the following ranking condition holds:

$$\text{rank}(\boldsymbol{H}_{o,b}) \leq n_x \leq \text{ number of rows of } \boldsymbol{H}_{o,b} \,. \tag{1.64}$$

Therefore, there exists a vector $\boldsymbol{V}_b^\top \in \mathbb{R}^{1\times(b+1)}$, $\boldsymbol{V}_b \neq \boldsymbol{0}$, called *parity vector* such that:

$$\boldsymbol{V}_b^\top \boldsymbol{H}_{o,b} = \boldsymbol{0} \,. \tag{1.65}$$

The parity vector can be found by solving (1.65). However, $\boldsymbol{V}_b$ is not guaranteed unique. The residual generator is designed as:

$$\boldsymbol{r}(t) = \boldsymbol{V}_b^\top \Big(\boldsymbol{y}_b(t) - \boldsymbol{H}_{u,b}\boldsymbol{u}_b(t) \Big) \,. \tag{1.66}$$

The set $P_b = \big\{ \boldsymbol{V}_b | \boldsymbol{V}_b \boldsymbol{H}_{o,b} = \boldsymbol{0} \big\}$ is called *parity space*.

To complete the parity space algorithm analysis, we need to consider the influences of $f(t)$ and $v(t)$ by modeling the output as: $y(t) = \boldsymbol{C}\boldsymbol{x}(t) + Du(t) + F_v v(t) + F_f f(t)$. Therefore, (1.62) becomes:

$$\boldsymbol{y}_b(t) = \boldsymbol{H}_{o,b}\boldsymbol{x}(t-b) + \boldsymbol{H}_{u,b}\boldsymbol{u}_b(t) + \boldsymbol{H}_{f,b}\boldsymbol{f}_b(t) + \boldsymbol{H}_{v,b}\boldsymbol{v}_b(t) \,; \tag{1.67}$$

with:

$$\boldsymbol{f}_b(t) = \begin{bmatrix} f(t-b) \\ f(t-b+1) \\ \vdots \\ f(t) \end{bmatrix}, \qquad \boldsymbol{H}_{f,b} = \begin{bmatrix} F_f & 0 & \dots & 0 \\ \boldsymbol{C}\boldsymbol{E}_f & F_f & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ \boldsymbol{C}\boldsymbol{A}^{b-1}\boldsymbol{E}_f & \dots & \boldsymbol{C}\boldsymbol{E}_f & F_f \end{bmatrix} \tag{1.68}$$

$$\boldsymbol{v}_b(t) = \begin{bmatrix} v(t-b) \\ v(t-b+1) \\ \vdots \\ v(t) \end{bmatrix}, \qquad \boldsymbol{H}_{v,b} = \begin{bmatrix} F_v & 0 & \dots & 0 \\ \boldsymbol{C}\boldsymbol{E}_v & F_v & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ \boldsymbol{C}\boldsymbol{A}^{b-1}\boldsymbol{E}_v & \dots & \boldsymbol{C}\boldsymbol{E}_v & F_v \end{bmatrix}$$

with: $\boldsymbol{f}_b(t) \in \mathbb{R}^{(b+1)\times 1}, \boldsymbol{H}_{f,b} \in \mathbb{R}^{(b+1)\times(b+1)}, \boldsymbol{v}_b(t) \in \mathbb{R}^{(b+1)\times 1}, \boldsymbol{H}_{v,b} \in \mathbb{R}^{(b+1)\times(b+1)}$. The residual generator (1.66) becomes:

$$\boldsymbol{r}_b(t) = \boldsymbol{V}_b\Big(\boldsymbol{H}_{f,b}\boldsymbol{f}_b(t) - \boldsymbol{H}_{v,b}\boldsymbol{v}_b(t)\Big), \qquad \boldsymbol{V}_b \in P_b \; . \tag{1.69}$$

The diagnosis detects a fault when the residual signal differs from zero. This condition arises when there is a fault or due to the noise, therefore this is not robust. Furthermore, this form is not ideal for online implementation.

The robust counterpart is achieved if there exists a parity vector, such that:

$$\boldsymbol{V}_b^\top \boldsymbol{H}_{f,b} \neq \boldsymbol{0} \qquad \text{and} \qquad \boldsymbol{V}_b^\top \boldsymbol{H}_{v,b} = \boldsymbol{0} \; . \tag{1.70}$$

This treatise is exposed in state space, but it is also applicable to transfer function models. The main difference with respect to the observer methods lies in no output error correction. Furthermore, the parity space leads to a discrete-time observer, such as *dead-beat observer* [23]. The parity space design can be solved by AFDP, which exploits the parity equations to minimize a cost function similar to the optimization problem (1.55) [28, Chapter 7].

1.3.4 Passive robustness in fault diagnosis

The above presented three robust methods belong to *active robustness in fault diagnosis*, in which the robustness is guaranteed by the residual generation procedure. Nevertheless, the robustness can be handled also by the residual evaluation step, called *passive robustness in fault diagnosis*. Again, the residual evaluation is a composition of residual processing and decision logic. The residual processing procedure consists of producing $\theta(t)$ from $r(t)$, which improves the fault detectability. Instead, the decision logic is a comparison between the processed residual with respect to a selected threshold τ. For fault detection aim, this is structured as:

$$\begin{cases} f(t) = 0 & \text{for} \quad \theta(t) < \tau \\ f(t) \neq 0 & \text{for} \quad \theta(t) > \tau \end{cases} \; . \tag{1.71}$$

Instead, for fault isolation goal, as:

$$\begin{cases} f_i(t) = 0 & \text{for} \quad \theta_i(t) < \tau_i \\ f_i(t) \neq 0 & \text{for} \quad \theta_i(t) > \tau_i \end{cases} \quad \text{and} \quad i = 1, ..., a \; ; \tag{1.72}$$

where: $f_i(t)$ represent a faults, $\theta_i(t)$ are preprocessed residual signals produced by the residuals generator and τ_i is the i-th suitable threshold. The presented decision logic belongs to the norm-based approach, but in literature there is also the statistical-based decision logic [28, Chapter 2].

Passive robustness develops a robust threshold coupled with a non-robust residual generator. This methodology has been introduced in [82], where the authors develops the RMS norm and derived a sort of *adaptive threshold* via a mechanism called *threshold selector*. Another examples of passive robustness are proposed in [83] and [84] which employ a fuzzy logic procedure to make a robust decision logic.

Remark 1.5

These methodologies are also applicable when the unknown input cannot be exactly decoupled, but only approximate decoupled. Therefore, to reduce further the false alarms probability, we can couple the approximate decouple residual generator with a passive robust fault evaluation. This is true especially with unstructured uncertainties [21].

1.4 Uncertainty in system identification

This section briefly reviews the system identification literature. Sections 1.4.1 reports the black-box traditional approach Prediction Error Method (PEM), which identifies a parametric model with a chosen model structure and a fixed model order. As described in Appendix A, the model structure selection is done by the user's knowledge, while the model order selection can be obtained by some techniques that define a "hard threshold" or with regularization approaches. Sections 1.4.2 describes the state-of-the-art technique, called kernel-based identification. In particular, we will show the time-domain non-parametric approach, based on the Reproducing Kernel Hilbert Space (RKHS) framework. Kernel-based identification identifies a model without any prior knowledge of the model structure. This technique derives from the machine learning literature. Specifically, the problem casts from a system identification problem to a function estimation perspective. As for the parametric method, in Section 1.4.2, the regularization technique for non-parametric system identification will be described. A briefly functional analysis framework review is described in Appendix B: for more details see [85].

All black-box models are endowed by the identification uncertainties, but these pieces of information are not always taken into account. The last part of this section describes the sources of the identification uncertainties and the different methods to represents these information. Primarily, the identification uncertainties are composed of two terms, called bias and variance. The Section 1.4.3 describes various methods that represent uncertainties without bias or with bias.

1.4.1 Prediction Error Method (PEM)

Prediction Error Method (PEM) is a parametric system identification approach that relies on a model class and selected orders. Both are represented by the model family $\mathcal{M}$ that corresponds to a set of models that describe the relationship between the input $u(t)$, fed to the plant, and the output $y(t)$, produced by the plant. The set of parameters of the model family are denoted as $\boldsymbol{\beta} \in \mathbb{R}^{m\times 1}$, where m is the number of parameters. The variable m represents the *model order*, called also *model complexity*. The general model is depicted in Figure 23. The represented model is an Linear Time-Invariant (LTI) and Single input Single output (SISO) dynamic system. The taxonomy of the dynamic system models are well-known in the system identification literature, for more information about that see [2, Chapter 6].

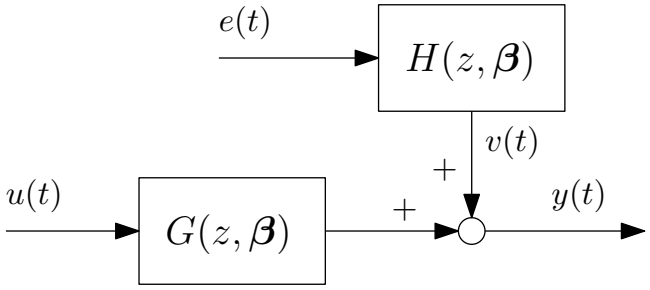


Figure 23: General model of an Linear Time-Invariant (LTI) Single input Single output (SISO) dynamic system.

The *model family* $\mathcal{M}$, parametrized by its parameters vector $\boldsymbol{\beta}$, is denoted by $\mathcal{M}(\boldsymbol{\beta}) \in \mathcal{M}$. The general model, represented in Figure 23, can be expressed in the following form:

$$\mathcal{M}(\boldsymbol{\beta}) : y(t) = G(z, \boldsymbol{\beta})u(t) + H(z, \boldsymbol{\beta})e(t) \; ; \tag{1.73}$$

where:

- $e(t)$ represents the zero mean withe noise;
- $G(z, \boldsymbol{\beta})$ is the transfer function from input to output;
- $H(z, \boldsymbol{\beta})$ is the transfer function from $e(t)$ to the output additive noise $v(t)$.

Usually, the transfer functions are chosen to be rational, as:

$$G(z, \boldsymbol{\beta}) = \frac{B(z, \boldsymbol{\beta})}{F(z, \boldsymbol{\beta})} \, , \qquad H(z, \boldsymbol{\beta}) = \frac{C(z, \boldsymbol{\beta})}{D(z, \boldsymbol{\beta})} \; ; \tag{1.74}$$

where $B(z), F(z), C(z), D(z)$ are polynomials functions. The measured output $y(t)$ corresponds to the real output model plus the uncertainties and disturbances which the model $G(z, \boldsymbol{\beta})$ cannot explain.

They are accounted by the signal $v(t)$. The impulse responses of both systems, by employing the inverse of the shift operator[15], are:

$$G(z, \boldsymbol{\beta}) = \sum_{k=1}^{\infty} g_k z^{-k} \,, \tag{1.75}$$

$$H(z, \boldsymbol{\beta}) = h_0 + \sum_{k=1}^{\infty} h_k z^{-k} \,, \quad \text{with } h_0 = 1 \,; \tag{1.76}$$

where h_k and g_k belongs to $\boldsymbol{\beta}$. By substituting the rational form of the transfer functions and by explicating the output signal, the general model becomes:

$$y(t) = \frac{B(z, \boldsymbol{\beta})}{F(z, \boldsymbol{\beta})} u(t-k) + \frac{C(z, \boldsymbol{\beta})}{D(z, \boldsymbol{\beta})} e(t) \,. \tag{1.77}$$

The first term is called *deterministic term*, instead the second *stochastic term*, since $H(z, \boldsymbol{\beta})$ is fed by white noise $e(t)$; therefore, $v(t)$ is seen as a stationary stochastic process. This interpretation is derived from the *spectral factorization theorem* [2, Chapter 6].

Definition 1.9: Stochastic process, Strictly stationary

A **stochastic process** $v(t, c)$ is a sequence of random variables produced by the same random experiment c. Usually, c is omitted. Furthermore, a stochastic process is **strictly stationary** if its joint probability distribution does not change when shifted in time.

The model classes are reported in table 3. All of these are well-known in the system identification literature. For instance: AR contains an AutoRegressive term described by the polynomial $A(z, \boldsymbol{\beta}) = 1 + a_1 z^{-1} + ... + a_{n_a} z^{-n_a}$, ARX corresponds to the composition of an AR model with an eXogenous part defined as $\frac{B(z,\boldsymbol{\beta})}{A(z,\boldsymbol{\beta})}$ or even ARMAX is the composition of MA with ARX.

The standard approach to identify the parameters vector $\boldsymbol{\beta}$ from a set of measured data $\mathcal{D} = \{u(1), ..., u(n), y(1), ..., y(n)\}$, is known as Prediction Error Method (PEM). It minimizes the difference between the measured output and the estimated output. The estimation is performed by an one-step-ahead predictor. This model representation aims of performing the prediction of the future output and it corresponds to write [86, Chapter 7]:

$$\hat{y}(t|t-1, \boldsymbol{\beta}) = \frac{H(z, \boldsymbol{\beta}) - 1}{H(z, \boldsymbol{\beta})} y(t) + \frac{G(z, \boldsymbol{\beta})}{H(z, \boldsymbol{\beta})} u(t) \,; \tag{1.78}$$

This is valid under the assumption of the inverse stability of $H(z, \boldsymbol{\beta})$. Furthermore, given a dataset $\mathcal{D}$, the parameters vector is obtained by optimizing:

$$\hat{\boldsymbol{\beta}} = \arg \min_{\boldsymbol{\beta}} J_n(\boldsymbol{\beta}) \,, \tag{1.79}$$

[15]The transfer functions are represented in the discrete time domain using z as the shift operator, i.e. $y(t+1) = z \cdot y(t)$.

Name	*Structure*
Moving Average (MA)	$y(t) = C(z, \boldsymbol{\beta})e(t)$
AutoRegressive (AR)	$y(t) = \frac{1}{A(z,\boldsymbol{\beta})}e(t)$
AutoRegressive with an eXogenous variable (ARX)	$y(t) = \frac{B(z,\boldsymbol{\beta})}{A(z,\boldsymbol{\beta})}u(t-k) + \frac{1}{A(z,\boldsymbol{\beta})}e(t)$
AutoRegressive Moving Average with an eXogenous variable (ARMAX)	$y(t) = \frac{B(z,\boldsymbol{\beta})}{A(z,\boldsymbol{\beta})}u(t-k) + \frac{C(z,\boldsymbol{\beta})}{A(z,\boldsymbol{\beta})}e(t)$
Output Error (OE)	$y(t) = \frac{B(z,\boldsymbol{\beta})}{F(z,\boldsymbol{\beta})}u(t-k) + e(t)$
Finite Impulse Response (FIR)	$y(t) = B(z, \boldsymbol{\beta})u(t-k) + e(t)$
Box-Jenkins (BJ)	$y(t) = \frac{B(z,\boldsymbol{\beta})}{F(z,\boldsymbol{\beta})}u(t-k) + \frac{C(z,\boldsymbol{\beta})}{D(z,\boldsymbol{\beta})}e(t)$

Table 3: Model classes.

where the cost function takes the form of:

$$J_n(\boldsymbol{\beta}) = \sum_{t=1}^{n} \left(y(t) - \hat{y}(t|t-1, \boldsymbol{\beta})\right)^2 \ . \tag{1.80}$$

Note that the parameters vector is obtained by exploiting a finite length dataset $\mathcal{D}$, however the goal is to learn a model that performs well on unseen data. To estimate this performance, the so-called *out-of-sample error* is evaluated. This corresponds to:

$$E_{out} = \mathbb{E}\left[\left(\frac{1}{H(z,\hat{\boldsymbol{\beta}})}\tilde{y}(t) - \frac{G(z,\hat{\boldsymbol{\beta}})}{H(z,\hat{\boldsymbol{\beta}})}\tilde{u}(t)\right)^2\right] \ . \tag{1.81}$$

E_{out} cannot be computed using $\mathcal{D}$, therefore a new dataset, called *test set*, is needed. This is denoted as $\mathcal{D}_{test} = \{\tilde{u}(i), \tilde{y}(i)\}_{i=1}^{n_T}$, with n_T is the number of the data in the test set. The higher n_T is, the more accurate the estimate of E_{out} will be. The counterpart of the out-of-sample error is *in-sample-error*, computed with $E_{in} = J_n(\hat{\boldsymbol{\beta}})$, by exploiting $\mathcal{D}$. This indicator, taken alone, is a bad estimation of the out-of-sample performance.

Figure 24 depicts the out-of-sample error and the in-sample error curves. As can be seen, these errors vary as the complexity of the model increases. In particular, by fixing a model class, when the model complexity is:

- *High*: the estimated model, usually, approximates the plant well. This means E_{in} is low. However, the complex model fits also the observed error dynamics[16] which do not belong to the true system. This results into a high E_{out} since the error dynamics depend on the realizations of the measured data. Thus, changing the dataset, $E_{out} >> E_{in}$. This phenomenon is called *variance*.

- *Low*: the estimated model, usually, misses the relevant dynamics of the plant and therefore it results in an high E_{in}. This phenomenon is called *bias*.

The minimization of the out-of-sample error takes the name of *bias-variance trade-off*. Unfortunately, the bias and variance cannot be computed, because they depend on the input and target probability distribution [87, Chapter 2]. Appendix A reports a conceptual tool called *bias-variance decomposition*, which explains the connection between bias, variance and the expected E_{out}. Specifically, this trade-off helps us to avoid the so-called *overfitting* phenomenon. This occurs when, as the complexity of the model increases, E_{out} increases and E_{in} decrease. Figure 24 highlights in red the overfitting area. The main cause of overfitting is model variance [87, Chapter 4].

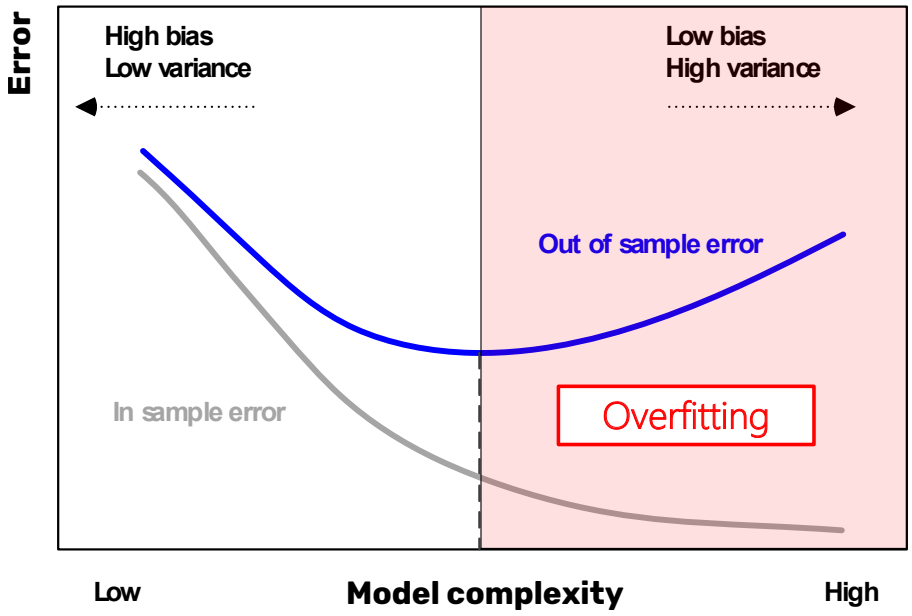


Figure 24: Out-of-sample error vs in-sample error curves, the red area highlights the overfitting phenomenon.

[16]Remember that the measured data are noisy (see (1.73)).

The main idea behind the bias-variance trade-off is that the model complexity must be correlated with the number of data in $\mathcal{D}$ and not with the target dynamic system complexity. In light of this, there is a heuristics rule which ties the number of data that the PEM needs n with respect to the number of parameters m:

$$n \geq 10 \cdot m \, . \tag{1.82}$$

This is simple, but shows a lot of limits. In literature, there are more reliable methods which solve the bias-variance trade-off, such as: Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), Cross Validation (CV) and *regularization*. These are described in Appendix A. Specifically, the first two methods obtain a threshold that selects the best model complexity, by comparing some indicators between a pool of different model complexity, but with same model class. Instead, the Regularization is a constrained version of (1.79), that reduce drastically the variance at the cost of an introduction of a small bias.

Remark 1.6

Note that, if we assume that $e(t)$ is a Gaussian white noise, PEM coincides with the Maximum Likelihood (ML) approach [3].

1.4.2 Kernel-based system identification

The main problem of the parametric model estimation is to choose the model family, which can cause unintended phenomena. The *Reproducing Kernel Hilbert Space (RKHS)* framework is employed to perform a time-domain non-parametric identification that ties input and output signals as $y = g(x)$, where g is not constrained into a specific parametric structure $g_{\beta} \in \mathbb{R}^{m \times 1}$, but it is searched in a infinite-dimensional function space Γ, which defines the hypothesis space.

By considering a function $g \in \Gamma$, a static non-parametric learning problem is defined as:

$$\hat{g}_0 \equiv \arg \min_{g \in \Gamma} \left(\sum_{i=1}^{n} (y_i - g(\boldsymbol{x}_i))^2 + \lambda J(g) \right) , \tag{1.83}$$

where the dataset of length $n \in \mathbb{N}^+$ is composed of $(\boldsymbol{x}_i, y_i) \in \Gamma \subseteq \mathbb{R}$, with $\boldsymbol{x}_i = \boldsymbol{x}(t_i) \in \mathbb{R}^{n \times 1}$ and $y_i = y(t_i) \in \mathbb{R}$. These are the i-th observed data which belong, respectively, to the regression matrix $\boldsymbol{\Phi} \in \mathbb{R}^{m \times n}$ and output vector $\boldsymbol{y} \in \mathbb{R}^{n \times 1}$. The regression matrix is the composition of the observed regressors $\boldsymbol{x}$ stacked in m rows. This formulation is similar to (A.10), indeed: the hyperparameter $\lambda \in \mathbb{R}^+$ modulates the regularization strength, $J(g)$ is the regularization term and the first term measures the function's fit to data. If Γ is a RKHS, the problem (1.83) becomes:

$$\hat{g}_0 = \arg \min_{g \in \mathscr{H}} \left(\sum_{i=1}^{n} (y_i - g(\boldsymbol{x}_i))^2 + \lambda \left\| g \right\|_{\mathscr{H}}^2 \right) . \tag{1.84}$$

To understand this equality, the RKHS definition must enunciated [88]:

Definition 1.10: Reproducing Kernel Hilbert Space (RKHS)

A **Reproducing Kernel Hilbert Space (RKHS)** is a Hilbert space $\mathscr{H}$ (its definition is reported in Appendix B) where its element $g : \Omega \to \mathbb{R}$ are functions, such that the pointwise evaluations are continuous linear functional L_x on $\mathscr{H}$, i.e.:

$$\forall x \in \Omega, \quad \exists L_x < \infty : |g(x)| \leq L_x \|g\|_{\mathscr{H}}, \quad \forall g \in \mathscr{H} ; \tag{1.85}$$

with Ω is a non-empty set.

The term $\|g\|^2_{\mathscr{H}}$ is computable since that the Hilbert space is endowed with an inner product $\langle \cdot, \cdot \rangle_{\mathscr{H}}$, which is complete with respect to the induced norm $\|g\|^2_{\mathscr{H}} = \langle g, g \rangle_{\mathscr{H}}$ (see Appendix B).

An important property that help us to solve the problem (1.84), is that a RKHS is linked with a *reproducing kernel*, defined in Definition 1.11.

Definition 1.11: Positive semidefinite kernel, Reproducing kernel and Kernel section

Let Ω denote a non-empty set. A symmetric function $\mathcal{K} : \Omega \times \Omega \to \mathbb{R}$ is a **positive semidefinite kernel**, if, for any finite $p \in \mathbb{N}$, it holds:

$$\sum_{i=1}^{p} \sum_{j=1}^{p} b_i b_j \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0, \quad \forall (\boldsymbol{x}_k, b_k) \in (\Omega, \mathbb{R}), k = 1, 2, ..., p \, . \tag{1.86}$$

If a semidefinite positive kernel holds the reproducing property:

$$g(x) = \langle g, \mathcal{K}_x \rangle_{\mathscr{H}}, \quad \forall (x, g) \in (\Omega, \mathscr{H}) \, . \tag{1.87}$$

The kernel is called **Reproducing kernel**.

The notation $\mathcal{K}_x \in \mathscr{H}$ represents the **kernel section** centered in x. It corresponds to $\mathcal{K}_x(a) = \mathcal{K}(\boldsymbol{x}, a), \forall a \in \Omega^a$.

[a] The kernel section will be written as $\mathcal{K}_x(a)$ or $\mathcal{K}_x$.

In particular, the following theorem describes the connection between RKHS and a reproducing kernel [89]:

Theorem 1.4: Moore-Aronszajn theorem

A RKHS corresponds to a unique reproducing kernel. Conversely, given a reproducing kernel defines a unique RKHS.

The Moore-Aronszajn theorem, in Theorem 1.4, explains that, if we define a reproducing kernel, we also know that exists a related RKHS, and by encoding the desired characteristic to the kernel function, we also define the properties of the functions that belong to the RKHS.

If we define g in a RKHS, the functions that belong to this space are infinite since the space is infinite-dimensional. The so-called *Representer theorem* defines how the regularization problem (1.84), that exploits the RKHS, admits a solution with a finite-dimensional representation [90]:

Theorem 1.5: Representer theorem

If $\mathscr{H}$ is a RKHS, the minimizer of (1.84) is:

$$\hat{g}_0 = \sum_{i=1}^{n} c_i \mathcal{K}_{x_i} \,, \tag{1.88}$$

where $\boldsymbol{c} = [c_1, c_2, ..., c_n]^\top \in \mathbb{R}^{n \times 1}$

Thus, the minimizer corresponds to a linear combination of $\mathcal{K}_{x_i}$, called also *basis functions*. Hence, by applying the Representer theorem, the kernel choice defines also the resulting functions properties. Note that $\boldsymbol{c}$ is a vector of length n, therefore the problem of the infinite-dimensional estimation boils down to a n coefficients estimation. The estimator (1.88) is also called as *regularization network* [91] or *least square support vector machine* [92].

The Representer theorem (1.88) gives us a reformulation of $\|g\|^2_{\mathscr{H}}$:

$$\begin{aligned} \|g\|^2_{\mathscr{H}} &= \langle g, g \rangle_{\mathscr{H}} \\ &= \langle \textstyle\sum_{i=1}^{n} c_i \mathcal{K}_{x_i}, \sum_{j=1}^{n} c_j \mathcal{K}_{x_j} \rangle_{\mathscr{H}} \\ &= \textstyle\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j \langle \mathcal{K}_{x_i}, \mathcal{K}_{x_j} \rangle_{\mathscr{H}} \\ &= \boldsymbol{c}^\top \boldsymbol{K} \boldsymbol{c} \end{aligned} \tag{1.89}$$

$\boldsymbol{K} \in \mathbb{R}^{n \times n}$ is a positive semidefinite matrix, such that $K_{ij} = \mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. This is called *Kernel matrix* (or *Gram matrix*). Thus:

$$\begin{aligned} \boldsymbol{g} &= [g(x_1), g(x_2), ..., g(x_n)] \\ &= \left[\sum_{i=1}^{n} c_i \mathcal{K}_{x_i}(x_1), ..., \sum_{i=1}^{n} c_i \mathcal{K}_{x_i}(x_n)\right] \\ &= \left[\sum_{i=1}^{n} c_i \mathcal{K}(\boldsymbol{x}_i, x_1), ..., \sum_{i=1}^{n} c_i \mathcal{K}(\boldsymbol{x}_i, x_n)\right] \\ &= \boldsymbol{c}^\top \boldsymbol{K} \end{aligned} \tag{1.90}$$

Thanks to (1.89) and (1.90), the problem (1.84) can be rewrite as:

$$\hat{\boldsymbol{c}} = \arg \min_{\boldsymbol{c} \in \mathbb{R}^{n \times 1}} \left\| \boldsymbol{y} - \boldsymbol{c}^\top \boldsymbol{K} \right\|_2^2 + \lambda \boldsymbol{c}^\top \boldsymbol{K} \boldsymbol{c} \,. \tag{1.91}$$

This is a quadratic optimization problem that can be solved in closed form. Specifically, the problem (1.91) can be rewritten by exploiting the partial derivatives with respect to $\boldsymbol{c}$ to zero, i.e.:

$$(\boldsymbol{K} + \lambda \boldsymbol{I}_n)\hat{\boldsymbol{c}} = \boldsymbol{y} \,. \tag{1.92}$$

Furthermore, if the kernel is non-degenerate, then $\boldsymbol{K}$ is positive definite and therefore invertible. Thus, we can write the final form of the estimator as:

$$\hat{\boldsymbol{c}} = (\boldsymbol{K} + \lambda \boldsymbol{I}_n)^{-1} \boldsymbol{y} \,. \tag{1.93}$$

As already mentioned, the reproducing kernel choice is a very crucial step to perform a correct estimation. Some examples of reproducing kernels are reported in Table 4.
If the employed kernel is a linear one, i.e. $\mathcal{K}(\boldsymbol{x}, \boldsymbol{z}) = \boldsymbol{x}^\top \cdot \boldsymbol{z}$ and if the kernel matrix is chosen as $\boldsymbol{K} = \boldsymbol{\Phi}^\top \boldsymbol{P} \boldsymbol{\Phi}$ with $\boldsymbol{P} \in \mathbb{R}^{m \times m}$ is symmetric semidefinite positive, then, by employing the theorem 1.4, the space RKHS contains only linear functions $g : \mathbb{R}^{m \times 1} \to \mathbb{R}$. Thus, the problem (1.91) becomes:

$$\hat{\boldsymbol{c}} = \arg \min_{\boldsymbol{c} \in \mathbb{R}^{n \times 1}} \left\| \boldsymbol{y} - \boldsymbol{c}^\top \boldsymbol{\Phi}^\top \boldsymbol{P} \boldsymbol{\Phi} \right\|_2^2 + \lambda \boldsymbol{c}^\top \boldsymbol{\Phi}^\top \boldsymbol{P} \boldsymbol{\Phi} \boldsymbol{c} \,. \tag{1.94}$$

Furthermore, by defining $\boldsymbol{\beta} \equiv \boldsymbol{c}^\top \boldsymbol{\Phi}^\top \boldsymbol{P}$, we obtain:

$$\hat{\boldsymbol{\beta}} = \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^{m \times 1}} \|\boldsymbol{y} - \boldsymbol{\beta} \boldsymbol{\Phi}\|_2^2 + \lambda \boldsymbol{\beta} \boldsymbol{\Phi} \boldsymbol{c} \tag{1.95a}$$

$$= \arg \min_{\boldsymbol{\beta} \in \mathbb{R}^{m \times 1}} \|\boldsymbol{y} - \boldsymbol{\beta} \boldsymbol{\Phi}\|_2^2 + \lambda \boldsymbol{\beta} \boldsymbol{P}^{-1} \boldsymbol{\beta} \tag{1.95b}$$

Name	*Structure*
Constant kernel	$\mathcal{K}(\boldsymbol{x}, \boldsymbol{z}) = 1$
Linear kernel	$\mathcal{K}(\boldsymbol{x}, \boldsymbol{z}) = \boldsymbol{x}^\top \cdot \boldsymbol{z}$
Gaussian kernel	$\mathcal{K}(\boldsymbol{x}, \boldsymbol{z}) = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{z}\|^2}{2\sigma^2}}$
Polynomial kernel	$\mathcal{K}(\boldsymbol{x}, \boldsymbol{z}) = (\boldsymbol{x}^\top \cdot \boldsymbol{z} + 1)^d$

Table 4: Some examples of reproducing kernel structure.

This problem is equal to Regularized Least Squares (ReLS) (A.15), therefore has the same closed forms, i.e.:

$$\hat{\boldsymbol{\beta}} = \boldsymbol{P}\boldsymbol{\Phi}\left(\boldsymbol{\Phi}^\top \boldsymbol{P}\boldsymbol{\Phi} + \lambda \boldsymbol{I}_n\right)^{-1} \boldsymbol{y} \quad \text{or} \tag{1.96a}$$

$$= \left(\boldsymbol{P}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \lambda \boldsymbol{I}_m\right)^{-1} \boldsymbol{P}\boldsymbol{\Phi}\boldsymbol{y} \tag{1.96b}$$

Remark 1.7

The solution of RKHS regularization via linear kernel (1.96) corresponds to the solution of regularized FIR estimation (A.15) and the MAP (A.17b) of the Bayesian linear model problem with Gaussian prior on the parameters. Specifically, the RKHS is a generalization of the FIR regularized estimation.

The kernel and the regularization parameters usually are contained in a hyperparameters vector $\boldsymbol{\eta}$. This can be estimated using the Marginalized Likelihood (MargLik) (A.18), as for the regularized parametric estimation. This operation is the counterpart of the model order selection in the classical parametric approach [6].

1.4.3 Model estimation uncertainty

The identified models are affected by an error composed of variance and bias (see Appendix A). The method which accounts for both variance and bias is *Robust identification*. This is described in the next subsection. By choosing the correct model family, a full-order model and a sufficiently

exciting input signal, the estimated model has zero bias. As already described, the model family is defined by the order and the model class (for PEM). Since, the model class is chosen by the user, PEM does not always guarantee the identified model has not some bias, instead the regularized kernel methods return a low bias model. Thus, the uncertainties derived from the bias are not considered in this subsection. In this way, we assume that the unmodeled dynamics do not arise and therefore the remaining sources of uncertainty are: unpredictable events and poor quality of the available data, see Section 1.1. Unpredictable events cannot be considered since they are not identifiable from the available information. Instead, the last source of uncertainty is highly tied with the variance error. In fact, in literature, the variance error is called also the *noise-induced error*. Usually, this error is introduced by the sensors that measure the signals. If the data are noiseless or the number of data is infinite, the variance error is equal to zero. Some system identification methodologies assume that the dataset is noise-free, therefore the uncertainty estimation with these cannot be evaluated. As described in [93], the uncertainty estimation methods assume different prior information, for instance: the so-called $\mathcal{H}_\infty$ *identification* needs of some frequency data at certain frequencies, with their confidence intervals, acquired by the real system [94, 95]. Another example is PEM, it finds a "soft" uncertainty bound, with given probability confidence and some assumption on the cost function. Instead, by applying the *Set membership* methodology, the main assumption is that the noise is unknown but bounded.

The uncertainty analysis in the system identification literature is well known for the PEM and therefore we proceed to review the uncertainty with this standard approach. In light of this, the relationship between the other identification methods and the model estimation uncertainty is reviewed. To obtain a bias free model we should impose some assumptions, these are described in Assumption 1.1.

Assumption 1.1: Bias-free assumptions

As already sad, we assume that:

- $\mathcal{S}_T \in \mathcal{M}$, with $\mathcal{S}_T$ denotes the plant family:

$$\mathcal{S}_T : y(t) = G_T(z)u(t) + H_T(z)e(t)\,, \tag{1.97}$$

  where $G_T(z), H_T(z)$ are the true system's transfer functions

- Number n of the sensed data is finite;
- $e(t) \sim (0, \sigma_e^2)$ is a white noise;
- An unique global minimum of (1.79) exists.

The experimental design plays an important role to guarantee the last assumption. The input signal must be enough exciting to prevent the bias phenomenon. Even if $\mathcal{S}_T \in \mathcal{M}(\boldsymbol{\beta})$, but the experimental design is poor, the identified model is endowed of bias. For instance, the main relationship between the input signal and the PEM cost function is:

Proposition 1.1: Experimental design rule

If a signal is sufficiently exciting, then the applied identification criterion (such as: PEM) has an unique solution $\boldsymbol{\beta}_{ideal} = \boldsymbol{\beta}_T$, when $\mathcal{S}_T \in \mathcal{M}(\boldsymbol{\beta})$. A signal $u(t)$ is said sufficiently exciting if $n_s \geq m$, where n_s is the order of the input signal[a] and m is the number of parameters. Usually, the order n_s is computed by plotting the power spectrum of $u(t)$.

[a]For instance: white noise has order $n_s = \infty$ and a sinusoidal signal $n_s = 2$.

Remember that, by having a stochastic noise $e(t)$, which corrupts the output data, $\hat{\boldsymbol{\beta}}$, explicated in (1.79), is a random variable. Hence, changing the dataset $\mathcal{D}$, $\hat{\boldsymbol{\beta}}$ changes too. Since the estimated model has no bias, the average parameters estimation is asymptotically $\mathbb{E}\left[\hat{\boldsymbol{\beta}}\right] = \boldsymbol{\beta}_T$, where $\boldsymbol{\beta}_T \in \mathbb{R}^{m\times 1}$ represents the real parameters vector. Therefore, the distribution of $\hat{\boldsymbol{\beta}}$, obtained with the PEM estimation, converges asymptotically to:

$$\hat{\boldsymbol{\beta}} \sim \mathcal{N}\left(\boldsymbol{\beta}_T, \bar{\boldsymbol{P}}_\beta\right) . \tag{1.98}$$

$\bar{\boldsymbol{P}}_\beta \in \mathbb{R}^{m\times m}$ is the covariance matrix of the parameters vector, defined as:

$$\bar{\boldsymbol{P}}_\beta \equiv \mathbb{E}\left[\left(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}_T\right)\left(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}_T\right)^\top\right] = \frac{\sigma_e^2}{n}\bar{\boldsymbol{R}}_\beta^{-1} . \tag{1.99}$$

$\bar{\boldsymbol{R}}_\beta \in \mathbb{R}^{m\times m}$ represents the asymptotic autocorrelation of $\boldsymbol{\beta}_T$, computed by:

$$\bar{\boldsymbol{R}}_\beta = \mathbb{E}\left[\boldsymbol{\Psi}(t, \boldsymbol{\beta}_T)\boldsymbol{\Psi}(t, \boldsymbol{\beta}_T)^\top\right] . \tag{1.100}$$

$\boldsymbol{\Psi}(t, \boldsymbol{\beta}_T) \in \mathbb{R}^{m\times 1}$ is the partial derivative of the one step identification error $\epsilon_1(t, \boldsymbol{\beta}) = y(t) - \hat{y}(t|t-1, \boldsymbol{\beta})$:

$$\boldsymbol{\Psi}(t, \boldsymbol{\beta}_T) = -\frac{d}{d\boldsymbol{\beta}}\epsilon_1(t, \boldsymbol{\beta})\bigg|_{\boldsymbol{\beta}=\boldsymbol{\beta}_T} . \tag{1.101}$$

If $v(t) = H_T(z)e(t)$ is a gaussian noise, PEM is asymptotically statistically efficient, therefore the equation (1.99) corresponds to the Cramer-Rao lower bound [2, Chapter 7]. All elements that compose the equation (1.99) depend on the real parameters vector $\boldsymbol{\beta}_T$, therefore in practice (1.99) is not computable. Thus, an approximation is necessary, so $\hat{\boldsymbol{R}}_\beta$ becomes:

$$\hat{\sigma_e}^2 = \frac{1}{n}\sum_{t=1}^{n}\left(y(t) - \hat{y}(t|t-1, \hat{\boldsymbol{\beta}})\right)^2 ; \tag{1.102}$$

$$\hat{\boldsymbol{R}}_{\beta} = \frac{1}{n}\sum_{t=1}^{n} \boldsymbol{\Psi}(t, \hat{\boldsymbol{\beta}})\boldsymbol{\Psi}(t, \hat{\boldsymbol{\beta}})^{\top} . \tag{1.103}$$

Furthermore, in [2, Chapter 7], it is proven that:

$$\bar{\boldsymbol{P}}_{\beta} \approx \frac{\hat{\sigma_e}^2}{n}\hat{\boldsymbol{R}}_{\beta}^{-1} . \tag{1.104}$$

Notice that:

- By increasing n decrease $\bar{\boldsymbol{P}}_{\beta}$;
- By increasing $\hat{\sigma}_e^2$ increase $\bar{\boldsymbol{P}}_{\beta}$.

To clarify these two observations, the Example 1.2 explains this phenomenon.

Example 1.2: Example of covariance in PEM identification at varying the properties of dataset $\mathcal{D}$

Given the true system:

$$\mathcal{S}_T : \quad y(t) = \frac{0.6z^{-1}}{1 + 0.4z^{-1}}u(t) + \frac{1}{1 + 0.4z^{-1}}e(t) \tag{1.105}$$

and ARX model of order ARX$(1, 1, 1)$:

$$\mathcal{M} : \quad G(z, \boldsymbol{\beta}) = \frac{bz^{-1}}{1 + az^{-1}} \quad H(z, \boldsymbol{\beta}) = \frac{1}{1 + az^{-1}} ; \tag{1.106}$$

with: $\boldsymbol{\beta} = [a, b]^{\top}$, $u(t)$ and $e(t)$ are white noise signals. All assumptions to neglect the bias are respected. We feed the system with different numbers of data n and noise variance σ_e^2. Then, we estimate the parameters vector 50 times for each conditions, by varying the $u(t)$ and $e(t)$ realizations. To perform the system identification, we employ the PEM method and the dataset is cut off from the transient data. Furthermore, the sampling time is $T_s = 1$ and the variance of the input is $\sigma_u^2 = 0.4$. The next figures depict the true values of a and b, denoted by the red circle and the blue crosses represent the 50 estimations. In particular:

- Figure 25 represents $\hat{\boldsymbol{\beta}}$ with $n = 200$ and $\sigma_e^2 = 0.4$;
- Figure 26 represents $\hat{\boldsymbol{\beta}}$ with $n = 200$ and $\sigma_e^2 = 1$;
- Figure 27 represents $\hat{\boldsymbol{\beta}}$ with $n = 1300$ and $\sigma_e^2 = 0.4$;
- Figure 28 represents $\hat{\boldsymbol{\beta}}$ with $n = 1300$ and $\sigma_e^2 = 1$.

As we expect, the variance increases by increasing σ_e^2 and decreases with high n.

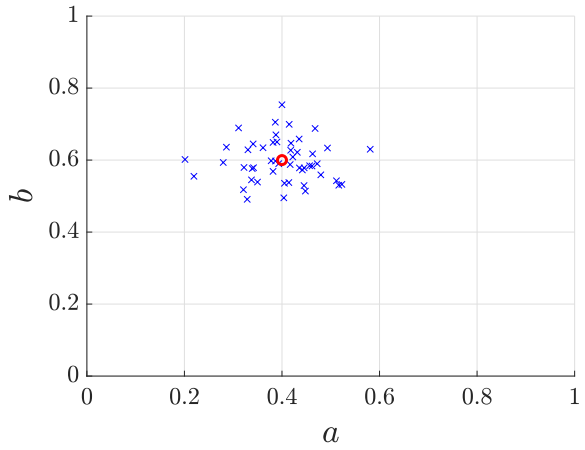


Figure 25: $\hat{\beta}$ estimation with low number of data and low variance of noise.

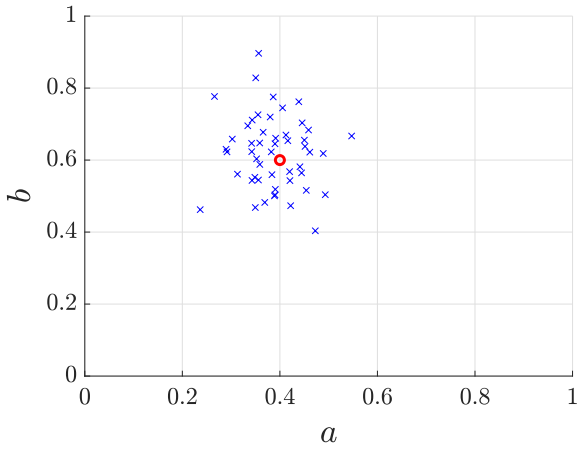


Figure 26: $\hat{\beta}$ estimation with low number of data and high variance of noise.

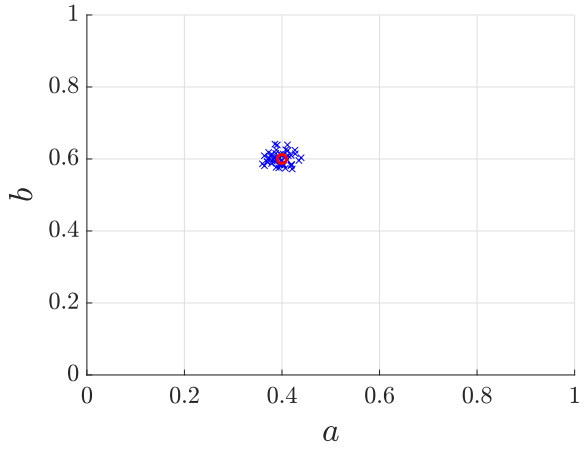


Figure 27: $\hat{\beta}$ estimation with high number of data and low variance of noise.

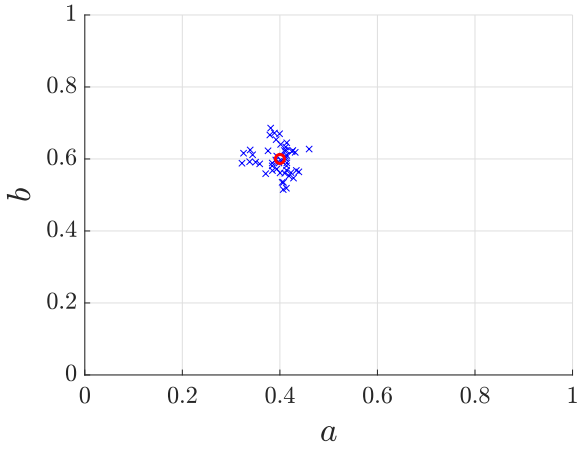


Figure 28: $\hat{\beta}$ estimation with high number of data and high variance of noise.

The Example 1.2 reports the solutions in the parameters space. The confidence interval in the parameter space is defined as:

$$C_\beta(\alpha) = \left\{ \boldsymbol{\beta} \in \mathbb{R}^{m\times 1} \middle| \left(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}\right)^\top \bar{\boldsymbol{P}}_\beta^{-1} \left(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}\right) \leq \alpha \right\} ; \tag{1.107}$$

with:

- α is the size of the confident region and it is such that $Pr(\mathcal{X}^2(m) \leq \alpha) = p_\beta$;
- p_β is the probability that the confidence interval contains $\boldsymbol{\beta}_T$;
- $\mathcal{X}^2 \sim \left(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}\right)^\top \bar{\boldsymbol{P}}_\beta^{-1} \left(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}\right)$, where $\mathcal{X}^2(m)$ is the chi-square probability density function with m degrees of freedom.

C_β in the parameters space is an ellipsoid centered at the identified parameters vector $\hat{\boldsymbol{\beta}}$ and shaped by $\bar{\boldsymbol{P}}_\beta$. Usually $p_\beta = 0.95$: it means that the real parameters vector is in C_β with a probability of 95%. An example of this variance representation is described in 1.3.

Example 1.3: Cont'd of Example 1.2

The confidence interval of the previous example with $p_\beta = 0.95$, represented in the parameters space, is depicted in:

- Figure 29 represents $\hat{\boldsymbol{\beta}}$ with $n = 200$ and $\sigma_e^2 = 1$;
- Figure 30 represents $\hat{\boldsymbol{\beta}}$ with $n = 1300$ and $\sigma_e^2 = 0.4$.

The cross represents the identified parameters and so the center of the ellipsoid.

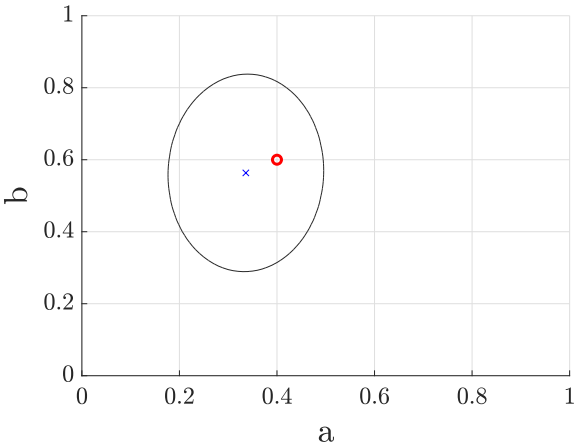


Figure 29: Confidence interval in the parameters space with low number of data and high variance of noise.

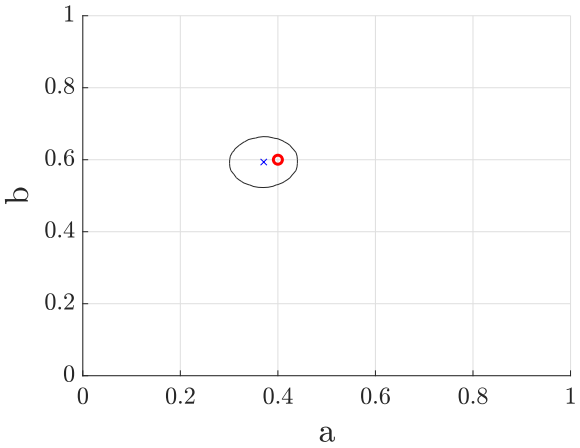


Figure 30: Confidence interval in the parameters space with high number of data and low variance of noise.

The uncertainty of PEM can be also represented in the frequency domain by exploiting the transfer function space. This is possible since the parameters vector $\hat{\boldsymbol{\beta}}$ is a random variable vector, therefore also the transfer function $G(e^{j\omega}, \hat{\boldsymbol{\beta}}) = \hat{G}_0(e^{j\omega})$ is a random variable. So, the covariance matrix of the transfer function in frequency domain is:

$$\bar{\boldsymbol{P}}_\beta(\omega) = \mathrm{Cov}\left(G\left(e^{j\omega}, \hat{\boldsymbol{\beta}}\right)\right) = \mathbb{E}\left[\left|G(e^{j\omega}, \hat{\boldsymbol{\beta}}) - G(e^{j\omega}, \boldsymbol{\beta}_T)\right|^2\right] . \tag{1.108}$$

Also in this domain, the covariance matrix of transfer function can be estimated using $\bar{\boldsymbol{P}}_\beta$ and $\hat{\boldsymbol{\beta}}$:

$$\mathrm{Cov}\left(G\left(e^{j\omega}, \hat{\boldsymbol{\beta}}\right)\right) \approx \boldsymbol{\Lambda}_G\left(e^{j\omega}, \hat{\boldsymbol{\beta}}\right) \bar{\boldsymbol{P}}_\beta \boldsymbol{\Lambda}_G^*\left(e^{j\omega}, \hat{\boldsymbol{\beta}}\right) ; \tag{1.109}$$

with $\boldsymbol{\Lambda}_G^\top\left(e^{j\omega}, \hat{\boldsymbol{\beta}}\right) = \frac{d}{d\boldsymbol{\beta}} G(e^{j\omega}, \hat{\boldsymbol{\beta}}) \in \mathbb{C}^{m\times 1}$. The same considerations of parameters domain are valid in frequency domain.

By assuming: $\mathcal{S}_T \in \mathcal{M}(\boldsymbol{\beta})$, $u(t)$ and $e(t)$ are uncorrelated, the authors in [86, Chapter 9] proposed an approximation of (1.108) as:

$$\mathrm{Var}(G(e^{j\omega})) \approx \frac{n_x}{n} \frac{\Gamma_v(\omega)}{\Gamma_u(\omega)} \; ; \tag{1.110}$$

with:

- $\Gamma_v(\omega)$ is the power spectral density of noise $v(t)$;
- $\Gamma_u(\omega)$ is the input density spectrum;
- n_x is the number of states of the system $G(z, \boldsymbol{\beta})$.

This is valid only if n_x is large, the real system can be described as a piecewise constant function of frequency ω and these constants are independent over the different frequency intervals. Due to these considerations, often, the approximation (1.110) cannot be applicable. An improvement of (1.110) is proposed by [96].

As for the covariance of the parameters, the transfer function covariance has a graphical representation. By denoting the frequency response of the model $G(z, \boldsymbol{\beta})$ as:

$$\boldsymbol{a}\left(e^{j\omega}, \boldsymbol{\beta}\right) \equiv \begin{bmatrix} \mathrm{Re}\left(G\left(e^{j\omega}, \boldsymbol{\beta}\right)\right) \\ \\ \mathrm{Im}\left(G\left(e^{j\omega}, \boldsymbol{\beta}\right)\right) \end{bmatrix} \tag{1.111}$$

and by using the same assumptions as before, the distribution of $\boldsymbol{a}\left(e^{j\omega}, \boldsymbol{\beta}\right)$ is Gaussian:

$$\boldsymbol{a}\left(e^{j\omega}, \hat{\boldsymbol{\beta}}\right) \sim \mathcal{N}\left(\boldsymbol{a}\left(e^{j\omega}, \boldsymbol{\beta}_T\right), \bar{\boldsymbol{P}}_\beta(\omega)\right) \; . \tag{1.112}$$

So, the confidence region can be easily built by drawing an ellipse $C_\beta(\alpha, \omega)$ for each frequency of $\boldsymbol{a}(e^{j\omega}, \boldsymbol{\beta}_T)$ in the Nyquist plane. This guarantees that $\boldsymbol{a}(e^{j\omega}, \boldsymbol{\beta}_T)$ is inside $C_\beta(\alpha, \omega)$, $\forall\omega$, with probability p_β. The confidence region in frequency domain becomes:

$$C_\beta(\alpha, \omega) = \left\{ \boldsymbol{a} \middle| \left(\boldsymbol{a} - \boldsymbol{a}(e^{j\omega}, \hat{\boldsymbol{\beta}})\right)^\top \bar{\boldsymbol{P}}_\beta^{-1} \left(\boldsymbol{a} - \boldsymbol{a}(e^{j\omega}, \hat{\boldsymbol{\beta}})\right) \leq \alpha \right\} \; . \tag{1.113}$$

The Example 1.3 shows the confidence interval representation of the problem described in Example 1.2.

Example 1.4: Cont'd of Example 1.2

The confidence interval of the previous example with $p_\beta = 0.95$, represented in frequency response space, is depicted in:

- Figure 31 represents $\hat{\beta}$ with $n = 200$ and $\sigma_e^2 = 1$;
- Figure 32 represents $\hat{\beta}$ with $n = 1300$ and $\sigma_e^2 = 0.4$.

The true system is represented in red line, the identified model in blue line and the uncertainty region in black lines.

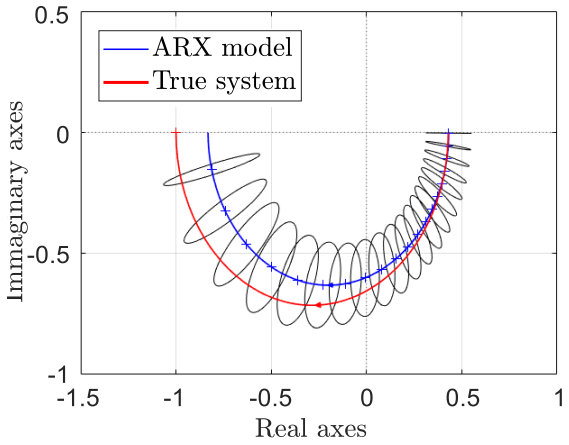


Figure 31: Confidence interval in frequency domain with low number of data and high variance of noise.

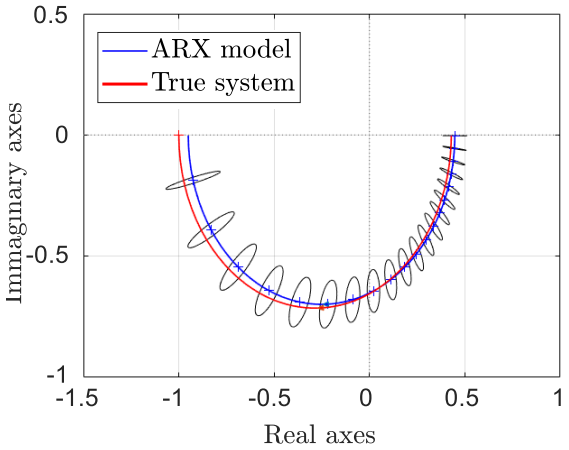


Figure 32: Confidence interval in frequency domain with high number of data and low variance of noise.

The authors of [97] have shown that this methodology does not maintain the probability level of the region $C_\beta(\alpha, \omega)$ with respect to the parameters domain. They have proposed a two steps projection to obtain the uncertainty in the Nyquist space that maintains the correct membership probability of the real transfer function. Instead, the authors of [98] have explained how the variance changes by choosing a general model (such as Box Jenkins) or by varying the signal-to-noise ratio and not only the noise variance. This leads to selecting a correct input variance, during the experimental design, by the noise variance knowledge.

Kernel system identification, as for PEM, suffers from the variance of the identification since the available dataset is noised. By assuming:

- β is a Gaussian random variable $\beta \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, with zero mean and covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{m \times m}$,
- $e(t)$ is Gaussian and independent from β,
- $e(t)$ and $\mathbf{\Phi}$ are known,

the y and β are jointly Gaussian variable [99, Chapter 2], as described in A, therefore the parameters vector can be seen as a posterior distribution $\beta|y \sim \mathcal{N}\left(\hat{\beta}, \hat{\mathbf{\Sigma}}\right)$. Mean and variance are compute

respectively by the closed forms (A.17b) (A.17c). In this way, kernel-based identification is endowed with the uncertainty represented by the variance of the posterior distribution. This representation is enough even if we consider all uncertainty sources (bias and variance) since the resulting model has low bias.

As for PEM, the input design can induce a variation in the covariance of the posterior $\hat{\boldsymbol{\Sigma}}$ (A.17c). In [100], the authors have explained how the input can be shaped when the kernel choice is known before the experiment design step. This is reasonable since most kernel methods employ the Empirical Bayes Method to tune the hyperparameters (Appendix A). The same is done in frequency domain, in [8].

Notice that PEM and kernel based produce a noise-induced bound called *soft bounds* (or *probabilistic bounds*), this is defined as:

$$\mathrm{Var}(G(e^{j\omega})) \in \mathcal{N}(0, W) \,, \tag{1.114}$$

A further bound is the so-called *hard bounds*:

$$\left\|\mathrm{Var}(G(e^{j\omega}))\right\|_l \leq O \,, \tag{1.115}$$

where $l = 1, 2, \infty$ and O is a finite number or a transfer function evaluated in frequency domain.

A system identification methodology that produces a hard bound of the variance is called *Set Membership estimation* [4]. The main assumption is that $e(t)$ is unknown but bounded. the signal $e(t)$ can be energy bounded or component-wise bounded. Specifically, the norm applied to $e(t)$ shapes the uncertainty set $\mathcal{B}_e$ in the error space. The two most used norm choices are:

- ∞-norm: which shapes $\mathcal{B}_e^{\infty}$ as a cube, in the measurement spaces. The formalization of the uncertainty space is:

$$\mathcal{B}_e^{\infty} = \left\{ \boldsymbol{e} \in \mathbb{R}^{n\times 1} : \|\boldsymbol{e}\|_{\infty} \leq \epsilon \right\} \,. \tag{1.116}$$

  where $\boldsymbol{e} = [e(1), e(2), ..., e(n)]$ and the cube side is equal to 2ϵ.

- 2-norm: which shapes $\mathcal{B}_e^2$ as a sphere, in the measurement spaces. It is defined as:

$$\mathcal{B}_e^2 = \left\{ \boldsymbol{e} \in \mathbb{R}^{n\times 1} : \|\boldsymbol{e}\|_2 \leq \epsilon \right\} \,. \tag{1.117}$$

  The sphere radius is ϵ.

The uncertainty space $\mathcal{B}_e$ is represented in measurement space by placing the center of $\mathcal{B}_e$ in the measured values $\boldsymbol{y}$ and by constructing a sphere or cube according to the type of norm. This representation is called *Measurement Uncertainty Set (MUS)*. Another graphical portrayal of $\mathcal{B}_e$, called *Estimated Uncertainty Set (EUS)*, corresponds to a projection into the parameters spaces [5].

This is done by employing a function $\boldsymbol{A} : \mathcal{R}^n \rightarrow \mathcal{R}^m$ that makes the projection. The norm choice has also an implication in the form of MUS and EUS, therefore, we highlight the distinction with:

- ∞-norm: the sets are denoted MUS^{∞} and EUS^{∞};
- 2-norm: the sets are denoted MUS^2 and EUS^2.

The first type is depicted in Figure 33, as illustrated at the left of the image, highlighted in red, is reported MUS^{∞}, its shape is a cube centered in $\boldsymbol{y}$, instead EUS^{∞} is a polytope. The second type is reported in Figure 34, MUS^2 is in the left graph, highlighted in green. As we expected, the shape of MUS^2 is a sphere, instead EUS^2 is an ellipse. The EUS size gives us an indication of the "quality" of the estimation. The blue arrow, in both MUS$^{\infty}$ and MUS^2, represents the noise vector $\boldsymbol{e} = \boldsymbol{y}_T - \boldsymbol{y}$, where $\boldsymbol{y}_T$ represents the true output values.
The range of each parameter β_j is defined as $\hat{\beta}_j^l \leq \beta_{T_j} \leq \hat{\beta}_j^u$, since the real parameters vector is $\boldsymbol{\beta}_T \in$ EUS, by assuming zero bias and by defining the *Estimate Uncertainty Interval (EUI)* as:

$$\text{EUI}_j^a = \left[\min_{\boldsymbol{\beta} \in \text{EUS}^a} \beta_j, \max_{\boldsymbol{\beta} \in \text{EUS}^a} \beta_j \right] = \left[\hat{\beta}_j^l, \hat{\beta}_j^u \right] \subset \mathbb{R}, \quad j = 1, ..., m \,, \tag{1.118}$$

with $a = 2, \infty$. The upper bound of the estimation error of the parameters is:

$$\left| \hat{\beta}_j - \beta_{T_j} \right| \leq \frac{\hat{\beta}_j^u - \hat{\beta}_j^l}{2} \,. \tag{1.119}$$

Therefore, the symmetric center of EUS is $\hat{\boldsymbol{\beta}}$, i.e. the mean of the set. This highlights that if $\boldsymbol{y}_T$ is in MUS, the corresponding EUS contains $\boldsymbol{\beta}_T$.
The EUS is not the smallest set in the parameters domain that contains $\boldsymbol{\beta}_T$. To find this, we have to construct a further set that is consistent with both prior knowledge and measured data, called *Feasible Parameter Set (FPS)*. If the parameter estimation is computed through the Least Square optimization, therefore the output is $\boldsymbol{y} = \boldsymbol{\Phi}^{\top} \hat{\boldsymbol{\beta}} + \boldsymbol{e}$ and $\boldsymbol{A}$ becomes $\boldsymbol{A} = \left(\boldsymbol{\Phi}^{\top} \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi}^{\top}$. Doing so, FPS is defined as:

$$\text{FPS}^a = \left\{ \boldsymbol{\beta} \in \mathbb{R}^{m \times 1} : \left(\boldsymbol{y} - \boldsymbol{\Phi}^{\top} \boldsymbol{\beta} \right) \in \mathcal{B}_e^a \right\} \,, \tag{1.120}$$

where $a = 2, \infty$. Hence, a general parameters vector $\boldsymbol{\beta}$, that is in FPS, is said to be feasible. The relationship that ties the Estimated Uncertainty Set with the Feasible Parameter Set is FPS$^a \subseteq$ EUS^a [101]. From FPS, it is possible to create its parameters interval by computing:

$$\text{PUI}_j^a = \left[\min_{\boldsymbol{\beta} \in \text{FPS}^a} \beta_j, \max_{\boldsymbol{\beta} \in \text{FPS}^a} \beta_j \right] = \left[\beta_j^l, \beta_j^u \right] \in \mathbb{R}, \quad j = 1, ..., m \,, \tag{1.121}$$

with $a = 2, \infty$.

Since $\text{FPS}^a \subseteq \text{EUS}^a$, therefore $\text{PUI}_j^a \subseteq \text{EUI}_j^a$. Thank to this, it is demonstrable that FPS contains $\boldsymbol{\beta}_T$, since:

$$\hat{\beta}_j^l \leq \beta_j^l \leq \beta_{T_j} \leq \beta_j^u \leq \hat{\beta}_j^u \,. \tag{1.122}$$

β_j^l and β_j^u can be found by a linear programming problem.

The optimal estimation $\hat{\boldsymbol{\beta}}^{opt}$ of the Set membership identification is given by the minimization of the estimated error $\epsilon(\hat{\boldsymbol{\beta}})$ with an estimated $\hat{\boldsymbol{\beta}}$, defined as:

$$\epsilon(\hat{\boldsymbol{\beta}}) = \sup_{\boldsymbol{\beta} \in \text{FPS}} \left\| \boldsymbol{\beta} - \hat{\boldsymbol{\beta}} \right\| \,. \tag{1.123}$$

In particular, $\hat{\boldsymbol{\beta}}^{opt}$ is such that $\epsilon(\hat{\boldsymbol{\beta}}^{opt}) \leq \hat{\boldsymbol{\beta}} \quad \forall \hat{\boldsymbol{\beta}} \in \mathbb{R}^{m \times 1}$. So, the optimal estimation is the central estimation of FPS , i.e. the mean $\hat{\beta}_j^{opt} = \hat{\beta}_j^C = \frac{\beta_j^u - \beta_j^l}{2}$.

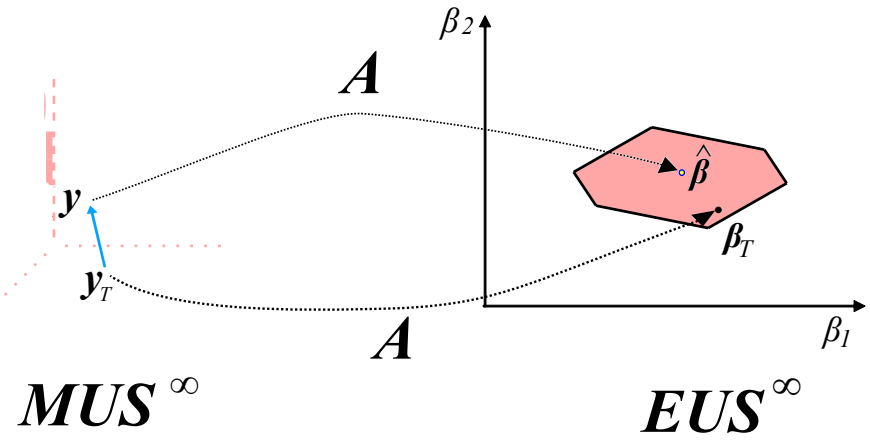


Figure 33: Graphical representations of $\mathcal{B}_e^{\infty}$ in the measurements space and parameters space.

Often, the bound choice for linear systems is 2-norm, since the ellipse is easier to handle than the polytope [102].

Figure 35 represents the shape of the distribution in the stochastic case and the Set membership. The Set membership probability density of the noise $e(t)$ is unknown but bounded, instead using the stochastic model is unbounded, but assumed known (in this case Gaussian). In [103], a combination of the probability densities between the stochastic identification and the Set membership identification is studied, to model a multi-sensor system.

Remark 1.8

The uncertainty sources are mainly the same between the robust control world and the system identification literature, but there is a gap between the uncertainty representations of the two kinds of literature.

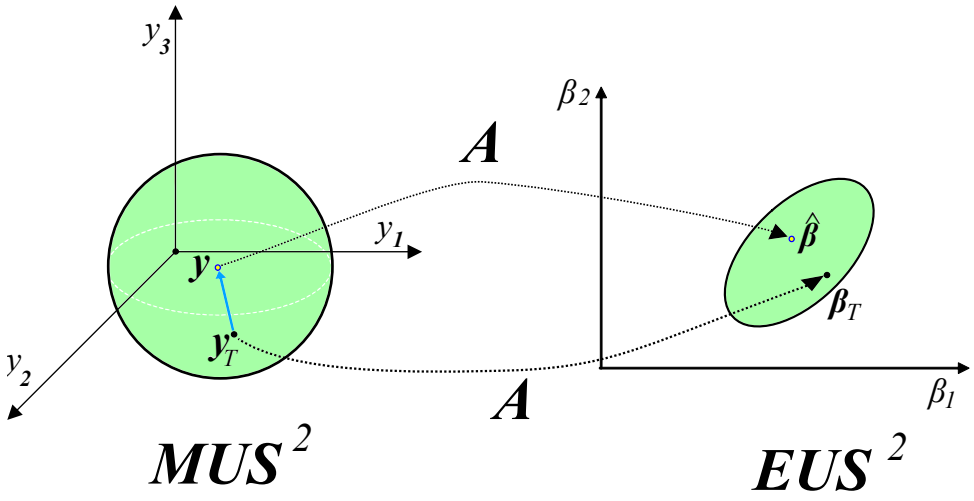


Figure 34: Graphical representations of $\mathcal{B}_e^2$ in the measurements space and parameters space.

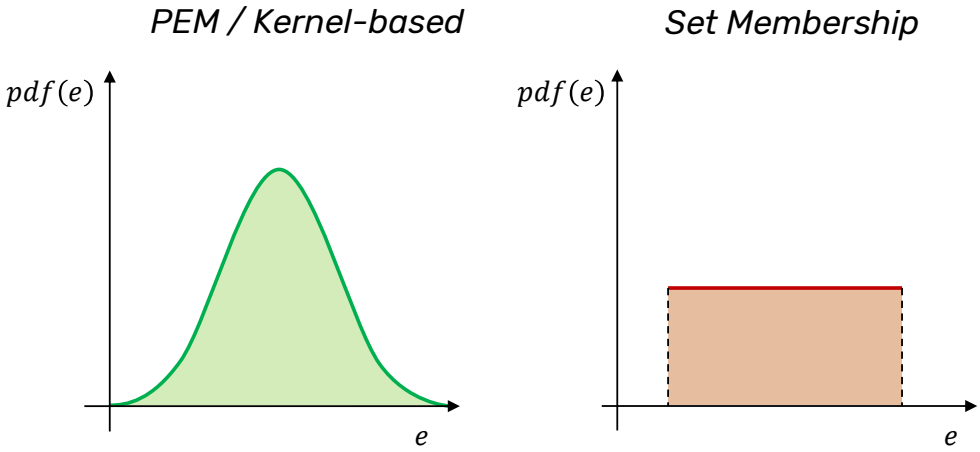


Figure 35: Analysis of the probability density function of all possible values of the noise.

Robust identification

The *Robust identification* identifies the nominal model and *error model*, i.e. a model that represents both bias and variance. The most famous robust modeling approaches are: Stochastic Embedding (SE), Model Error Modeling (MEM) and Set Membership [10]. Note that kernel-based system identification is not applied in this literature because the main assumption is low bias, therefore the error model is approximated by the variance error.

Stochastic Embedding [104] defines the true system frequency response $G_T(j\omega)$ as a random variable: $G_T(j\omega) = \hat{G}_0(j\omega) + \Delta_G(j\omega)$, where $\Delta_G(j\omega)$ is the true *model error model* (i.e. the bias term) designed as a random variable independent from the identified nominal model $\hat{G}_0(j\omega)$. *The uncertainty region of the model error model is the variance term.* If we consider that the data are noisy, the model of the true system, evaluated in frequencies ω_k is $\hat{\hat{G}}(j\omega_k) = G_T(j\omega_k) + e_k, \quad k = 1, ..., m$, with e_k is

the noise of the frequency response observations independent from $G_T(j\omega)$ and $\Delta_G(j\omega)$. Hence, by combining the noise term and the bias term, the model can be represented as:

$$\hat{\hat{G}}(j\omega_k) = \hat{G}_0(j\omega_k) + \Delta_G(j\omega_k) + e_k,\ k = 1, ..., m\ . \tag{1.124}$$

The authors of [9] use the linear regression to parametrize the nominal model as $\hat{G}_0(\boldsymbol{\beta}) = \sum_{i=1}^{m} o_i\beta_i$, where: $\boldsymbol{O} = [o_1, ..., o_m]$ are m-dimensional vector of orthonormal basis functions and $\boldsymbol{\beta} \in \mathbb{R}^m$ is the parameters vector. Furthermore, the magnitude of model error model is greater as ω increases. This is accounted by the basis functions. Hence, the bias error can be modeled as $\boldsymbol{O}\bar{\boldsymbol{\beta}}\Lambda$, where Λ is a random walk process over ω and $\bar{\boldsymbol{\beta}} \in \mathbb{R}^{m\times 1}$ a priori known parameters.
In light of this, the identification procedure is composed of:

- Perform a pointwise Least-Squares, which delivers $\hat{\hat{G}}(j\omega_k)$, where: $k = 1, ..., n_f$[17] and the input signal is enough exciting for the system;
- Compute the statistical properties of the output noise e_k, usually assumed Gaussian;
- Choose a set of basis functions $\boldsymbol{O}$;
- Estimate $\boldsymbol{\beta}$ and Λ, with $\hat{\hat{G}}(j\omega_k, \boldsymbol{\beta}) = \boldsymbol{O}\boldsymbol{\beta} + \boldsymbol{O}\bar{\boldsymbol{\beta}}\Lambda + e_k$ and chosen $\boldsymbol{O}$. Specifically:
  - $\hat{\boldsymbol{\beta}}$ is found based on the knowledge of $\hat{\hat{G}}(j\omega_k)$. So, the nominal model is a least-square approximation of $\hat{\hat{G}}(j\omega_k)$ in the subspace spanned by the basis functions;
  - When no a priori information on the bias term is available, the model error model parameters vector is chosen as $\bar{\boldsymbol{\beta}} = \hat{\boldsymbol{\beta}}$;
  - Λ is chosen in such a way that the variance of $\Delta_G(j\omega)$ increases linearly after a chosen critical frequency. Doing so, the undermodeling can be represented as a multiplicative error;
  - Compute the statistical properties of $\boldsymbol{O}\bar{\boldsymbol{\beta}}\Lambda$.

The second approach, MEM, estimates the nominal model by adopting PEM method [1]. After, the model error modeling is done by computing the residual signal as:

$$\epsilon(s) = Y(s) - \hat{G}_0(s)U(s)\ . \tag{1.125}$$

Then, the model error model $G_e(s)$ is identified using input and error observations. This represents the model of the bias term. The confidence region of $G_e(s)$ is given by exploiting the uncertainty region

[17] n_f is the number of frequency samples.

of $\hat{G}_0(s)$, identified by PEM (see Section 1.1), centered in $G_e(s)$ and truncated at 99% percentile. This is valid only if the model is unfalsified, which means:

Definition 1.12: Unfalsified model

A *falsifying test* is a test where the measured real system output is compared with the signal estimated by the identified model and fed by the same chosen input[a]. If this difference is low, then the model is called **unfalsified**. Thus, if the model error model is unfalsified, the model explains correctly the relation between the residual and the input signal.

[a]The dataset employed to perform this test differs from the system identification dataset.

The final model, i.e. the identified model that accounts also the bias and the variance, is computed by frequency sampling the model error model and after that, these are added to the model $\hat{G}_0(s)$ (sampled with the same frequency vector). The construction of the final model uncertainty region can be done in two ways [10]:

- The uncertain region of the final model corresponds to the uncertainty region of model error model centered in the final model, depicted in Figure 36;

- Endowing the frequency response of the final model with a symmetric uncertain region. It is more conservative than the first option. The values of the uncertainty region boundary correspond the maximum value between the distances from the lower and upper uncertainty boundary of the uncertainty region, found with the first way, to the final model. This is done for each sampled frequency.

Note that the nominal model is unfalsified only if this is inside the final model uncertainty region. It corresponds to saying that the uncertainty region centered in the model error model contains the 0 value.

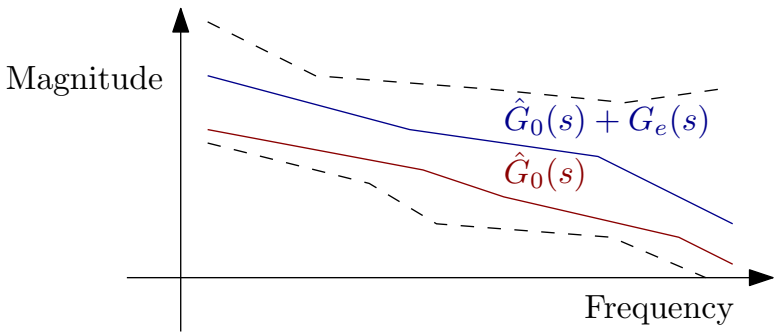


Figure 36: Graphical final model (nominal model plus model error model) endowed of the uncertainty region (dashed lines).

Another observation is that, as already said, the residual can be seen as a sum of bias and variance terms, as:

$$\epsilon(s) = \Delta_G(s)U(s) + E(s) \; . \tag{1.126}$$

where $E(s)$ is the laplace transform of the signal $e(t)$. Therefore, the model error model $G_e(s)$ endowed with its uncertainty is an estimation of $\Delta_G(s)$. The model family must be enough flexible to obtain an unbiased model, but the flexibility must be not unnecessary flexible to avoid the overfitting phenomenon. Hence, the model family choice is not trivial.

The third methodology employs the Set membership estimation in a model error modeling framework [11]. The procedure assumes the unknown but bounded of both residual and noise signals. The strategy relies on: identifying the nominal model using the Set membership estimation, then computing the residual as in (1.125). The third step is critical because it chooses the noise bound and the model error model structure. After that, the nominal model error $G_e(s)$ is identified. Specifically, if $G_e(s)$ is parametrized as: $G_e\left(s;\bar{\bar{\boldsymbol{\beta}}}\right) = \sum_{i=1}^{\bar{\bar{m}}} \bar{\bar{o}}_i\bar{\bar{\beta}}_i$, then the Feasible Parameter Set of the model error model is given by:

$$\text{FPS}_e^a = \left\{\bar{\bar{\boldsymbol{\beta}}} \in \mathbb{R}^{\bar{\bar{m}}\times 1} \left\| \epsilon(s) - G_e\left(s;\bar{\bar{\boldsymbol{\beta}}}\right)\right\|_a \leq \nu\right\} \; ; \tag{1.127}$$

where:

- $\|e\|_a \leq \nu$ for a given $\nu > 0$;
- $\bar{\bar{\boldsymbol{O}}} = [\bar{\bar{o}}_1, ..., \bar{\bar{o}}_{\bar{\bar{m}}}]$ is the vector of basis function of the model error model;
- $\bar{\bar{\boldsymbol{\beta}}} \in \mathbb{R}^{\bar{\bar{m}}\times 1}$ is the parameters vector of the model error model;
- $a = 2, \infty$.

Thus, the model error model is identified using optimal or suboptimal estimator based on (1.127). The last step consists of mapping the nominal model plus the model error model onto the frequency domain [10].

A huge problem in practical applications is that the bound ν is set by the knowledge of the user, so it may be too conservative or too small, leading to an empty FPS_e^a.

Remark 1.9

The Robust identification allows modeling the bias error and the variance error terms in an undermodeling setting. Some critical choices, such as the model family of the model error model in SE or MEM or even the bound of the input noise in model error modeling, with the Set membership method, make this methodology heavily dependent on the prior knowledge of the real system.

1.5 Conclusions

Different kinds of literature are reviewed focusing on the role of the uncertainty, from the modeling to the usage passing through the representation. Figure 37 depicts a scheme of the concepts described in the Chapter 1. The straight arrows and lines represent a direct connection between the terms/nodes, instead the dashed arrows represent a conceptual connection. Specifically, the upper arrow between the uncertainty and LFT corresponds to the uncertainty derived from the user's knowledge, whereas the lower dashed arrow represents the connection between the uncertainty derived from the identification methodologies. This distinction means that the representation of the upper case is well defined in the literature, instead the lower case has a gap between the two uncertainties representations. The other dashed line represents the connection between the signal-based fault detection and robust fault detection: these are not directly connected, but the signal-based one belongs to the more general fault detection approach.

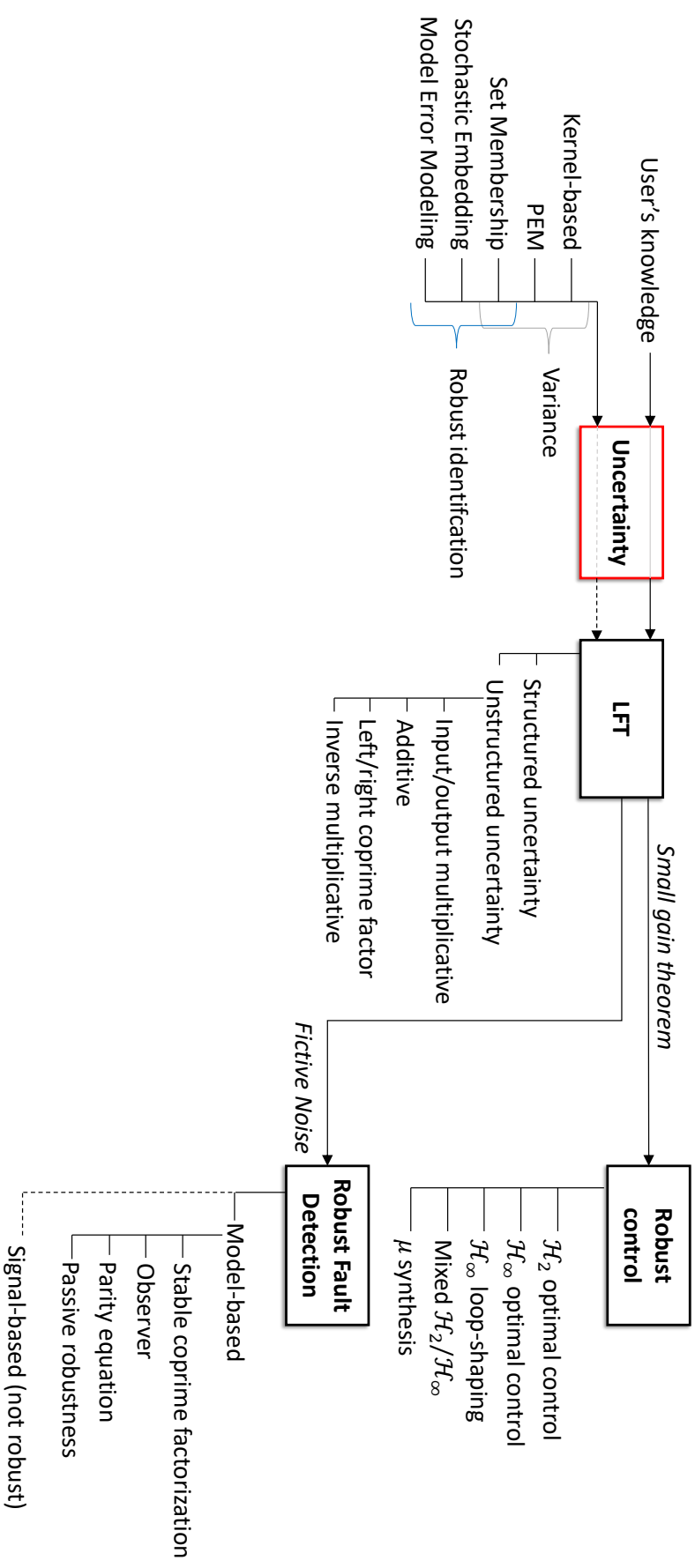


Figure 37: Graphical resume of the state of the art.

Part I

Theoretical contributions

Chapter 2. Data-driven mixed-sensitivity control of LTI systems with automated weighting functions selection

In this chapter, we present our theoretical contribution. The idea is to design a robust controller through $\mathcal{H}_\infty$ loop-shaping, specifically using a declination called *S/T mixed-sensitivity approach*. The aim is to bridge the gap between system identification and robust control synthesis, where uncertainty regions provided by the estimation methods can be used to define an uncertainty set for robust stability. Typically, a model error model is used to represent the modeling bias along with its variance (see 1.4.3). Instead, we propose to employ kernel-system identification to perform a low-bias Robust identification. Doing so, the user is relieved from:

- The model family selection;
- The model error modeling choices;
- The weights functions design of the S/T mixed-sensitivity algorithm.

Furthermore, the weight functions are data-driven designed with the aim to obtain both robust stability-nominal performance and robust performance. For both design problems, we compare our approach with PEM in a simulation benchmark. Results show how kernel methods provide a more reliable uncertainty representation for robust control, due to their low bias modeling capability. In Chapter 3, we describe also an extension of the proposed method, where the goal becomes to deal with multiple uncertainty sources.

2.1 Motivation

Mixed-sensitivity control design [19] refers to a class of optimization based control problems where the sensitivity function is shaped along with other closed-loop transfer functions [16, Chapter 9]. Standard results in robust control theory [26] link the mixed-sensitivity problem to the design of controllers that guarantee robust stability and performance under bounded uncertainties [24, Chapter 4].

However, the S/T mixed-sensitivity approach requires the *selection of stable and proper transfer functions that define stability and performance requirements* of the feedback system, and finding suitable instances of such weights is known to be the most critical task within this framework. To this purpose, some general guidelines for the tuning of such weights are given in the reference texts

[105, 106], while specific ones for second-order plants and for tracking sinusoidal signals are provided, respectively, in [107] and [108]. A common suggestion is to first define the weighting transfer function for robust stability $W_T(z)$ based on such guidelines, and then manually aim for an adequate weight $W_S(z)$ for control performance [109, 110].

Although this rationale might sometimes lead to acceptable results in practice, its drawback is that $W_S(z)$ depends on the previously fixed $W_T(z)$, so that - if this weight is too conservative- the designed controller will exhibit poor performance. The authors of [111] faced the problem of tuning $W_T(z)$ based on the discrepancy among the models identified at several operating regimes. The uncertainty information provided by the identification approach is not leveraged, and the selection of the number and type of operating points is a user choice. The weight $W_S(z)$ is still manually tuned based on the reconstructed $W_T(z)$. The joint optimization of $W_T(z)$ and $W_S(z)$, along with the controller, is faced in [112, 113, 114] under a robust performance constraint.

All the above approaches are based on the key assumption that a mathematical description of the system is available for control design. When this is not the case, a model is usually identified from data through black-box approaches using some collections of measurements. Since identification is here motivated by robust-control design, the model order is usually selected to be relatively low [10] in order to keep the complexity of the controller limited. However, in this way, undermodeling errors become usually not negligible and input/output data prefiltering has to be employed to shape the arising bias error, although the tuning of the prefilter band is nontrivial [115]. If not properly taken into account, the modeling bias can compromise the uncertainty region that might not contain the true system, thus jeopardizing the design of robust controllers. As an alternative solution, kernel-based learning methods can be employed to identify an *high-order low-bias model*. Kernel-based methods are regularized approaches endowed with the Bayesian interpretation of Gaussian Processes (GP), so that a posterior Gaussian distribution on the plant impulse response can be obtained. The use of the kernel/GP approach allows one to get rid of the bias error (which is still present but practically negligible) and of its modeling, only assuming a Gaussian output noise distribution (see also the recent works [116, 117]).

This work investigates the use of kernel-based learning methods for S/T mixed-sensitivity $\mathcal{H}_\infty$ control design as well as the opportunities offered by this approach for the selection of the weighting functions, see Figure 38. More specifically, the derived uncertainty measure is used to determine the stability weight $W_T(z)$, based on which $W_S(z)$ is tuned by a multi-objective optimization for maximum attainable control performance. In the proposed approach, the data-driven design of the mixed-sensitivity weights has thus been *automated* and made *non-iterative* by reducing the user choices to the minimum.

Specifically, only some configuration parameters must be selected by the user. The automated design of $W_T(z)$ relies on a randomized algorithm rationale that depends on an *accuracy* and *confidence* levels for the reconstruction of the uncertainty bound. The design is first performed non-parametrically in the frequency domain, so that it is necessary to specify a discrete *frequency grid* for the transfer function evaluation. Then, a parametric model for $W_T(z)$ is fit by specifying a *model order*. The automated design of $W_S(z)$ is performed solving an optimization problem requiring the specification of the maximum allowed *settling time* and minimum allowed *overshoot*, in terms of closed-loop step-response. The optimization problem is considered feasible depending on a *slack threshold*.
Differently from the reviewed literature:

- We jointly consider a robust stability/nominal performance (or robust performance) control objective, without limiting the use of the uncertainty set provided by identification to the robust stability aim. Rather, it serves for the purpose of $W_S(z)$ optimization;
- As little as possible conservatism is encoded in $W_T(z)$, since it directly derives from the kernel-based identification, where bias is negligible, so that only $W_S(z)$ is optimized for the maximum attainable performance without affecting robust stability.

Remark 2.1

Note that, as said by Hjalmarsson in [119], for the *separation principle* when the identification procedure is done to obtain a controller, the user should *first try to model as well as possible*. After that any simplification can be performed without jeopardizing the statistical accuracy. So, by the kernel-based system identification, the best accurate model is identified and then the resulting non parametric function is projected into an high-order Finite Impulse Response model to obtain a finite order model without loosing any statistical accuracy.

Usually, the $\mathcal{H}_\infty$ approaches lead to a controller of the same order of the augmented model (nominal model and weighting filters). Note that by exploiting the kernel-based system identification the resulting model is an *high-order low-bias model*, so it could jeopardizing the applicability of the proposed method to a real world problem. To solve this, we have employed the `hinfstruct` optimization method, doing so the user fixes the control structures as desired.

2.1.1 S/T mixed-sensitivity loop-shaping

Consider the unknown stable LTI SISO plant $G_T(z)$, where z^{-1} denotes the backward shift operator, and two stable proper scalar weighting functions $W_S(z), W_T(z)$. The control aim considered in this work

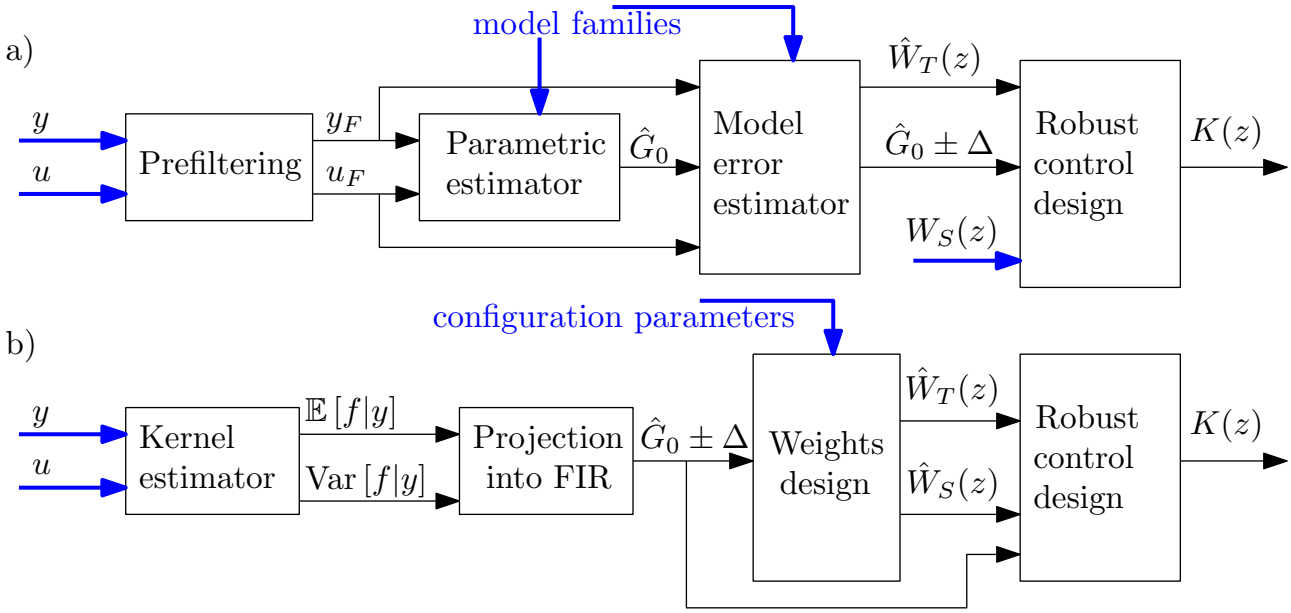


Figure 38: (a) Identification for robust control approach common to many benchmark methodologies, like stochastic embedding (SE) [118], model-error modeling (MEM) [86] and Set membership (SM). The system input and output are denoted by $u(t)$ and $y(t)$ respectively, with $u_F(t), y_F(t)$ being their filtered versions. $\hat{G}_0(z)$ denotes an estimated model of the plant, while $\Delta(z)$ is the estimated model uncertainty. (b) Our approach. $\mathbb{E}[f(t)|y(t)]$ and $\mathrm{Var}[f(t)|y(t)]$ denote respectively the posterior mean and autocovariance of the impulse response. $W_T(z)$ and $W_S(z)$ are weights functions in the S/T mixed-sensitivity rationale, and $K(z)$ is the designed robust controller. Blue lines indicate the information needed from the user.

is to design a LTI fixed-order controller $K(z, \boldsymbol{\rho})$, parametrized by the parameters vector $\boldsymbol{\rho} \in \mathbb{R}^{n_\rho \times 1}$, so as to minimize

$$J(\boldsymbol{\rho}, G_T) := \|\boldsymbol{H}(z, \boldsymbol{\rho})\|_\infty = \left\| \begin{matrix} W_S(z)S(z, \boldsymbol{\rho}) \\ W_T(z)T(z, \boldsymbol{\rho}) \end{matrix} \right\|_\infty , \tag{2.1}$$

where $S(z, \boldsymbol{\rho}) = [1 + K(z, \boldsymbol{\rho})G_T(z)]^{-1}$ is the sensitivity function and $T(z, \boldsymbol{\rho}) = 1 - S(z, \boldsymbol{\rho})$ is the complementary sensitivity function of the closed-loop system. The choice of $W_S(z), W_T(z)$ is critical and directly influences $K(z, \boldsymbol{\rho})$.

The S/T mixed-sensitivity problem (2.1) is often employed in robust control design for robust stability against multiplicative uncertainty at plant output [16, Chapter 9]. Let $\Delta(z)$ a stable transfer function that satisfies the bounded real condition $\|\Delta(z)\|_\infty \leq 1$, and consider a multiplicative output uncertainty model set:

$$G_u(z) := \big(1 + \Delta(z)W_o(z)\big)G_T(z) , \tag{2.2}$$

where $G(z)$ denotes a the perturbed SISO plant model and $\Delta(z)$ describes a normalized bounded frequency-domain uncertainty with $W_o(z)$ its frequency magnitude [16, Chapter 7],[26, Chapter 9]

(for further information see Section 1.1). As described in Section 1.2, this method is declined from more general $\mathcal{H}_\infty$ loop-shaping.

The cost function (2.1) is similar to (1.39) by letting $W_d(z) = W_S(z)$[1]. This represents the performance test. Instead, the robust stability test for the multiplicative uncertainty (described in Section 1.1) is obtained as $\|W_T(z)T(z,\boldsymbol{\rho})\|_\infty$, this results by choosing $W_o(z) = W_T(z) = W_n(z)G_T(z)$ (where $W_n(z)$ is defined in (1.39) and $W_o(z)$ is defined in (1.17)). With this representation: $W_T(z)$ shapes the uncertainty and $W_S(z)$ shapes the performance requirements.

If, in (2.1), we consider also the $\left\|W_Q(z)Q(z)\right\|_\infty$ term, the problem shifts to $S/T/KS$ mixed-sensitivity loop-shaping. Doing so, we can shape some desired control performance requirements by choosing properly $W_Q(z)$.

Under these settings, the problem (2.1) is solved imposing that [24, Chapter 4]

$$J(\boldsymbol{\rho}, G_T(z)) < 1, \text{ for robust stability and nominal performance} \tag{2.3a}$$

$$J(\boldsymbol{\rho}, G_T(z)) < 1/\sqrt{2}, \text{ for robust performance} \tag{2.3b}$$

The demonstration is the following proposition:

Proposition 2.1: Design a robust performance controller by employing the mixed-sensitivity cost function (2.1)

Before explaining how the robust performance can be obtained with the employment of the mixed-sensitivity loop-shaping, we recap the sufficient and necessary conditions of the stability and performance aims with multiplicative output uncertainty:

- Robust stability test:

$$\|W_T(z)T(z,\boldsymbol{\rho})\|_\infty < 1,\ \|\Delta(z)\|_\infty \leq 1\,, \tag{2.4}$$

  this corresponds to: $\left|T(e^{j\omega},\boldsymbol{\rho})\right| < \left|\frac{1}{W_T(e^{j\omega})}\right|\ \forall\omega$.

- Nominal performance test:

$$\|W_S(z)S(z,\boldsymbol{\rho})\|_\infty < 1\,, \tag{2.5}$$

- Robust stability and nominal performance test:

$$\|W_T(z)T(z,\boldsymbol{\rho})\|_\infty < 1\ \&\ \|W_S(z)S(z,\boldsymbol{\rho})\|_\infty < 1, \quad \|\Delta(z)\|_\infty \leq 1\,, \tag{2.6}$$

- Robust performance test[a]:

$$\left\|\boldsymbol{H}(e^{j\omega},\boldsymbol{\rho})\right\|_1 = \left|W_T(e^{j\omega})T(e^{j\omega},\boldsymbol{\rho})\right| + \left|W_S(e^{j\omega})S(e^{j\omega},\boldsymbol{\rho})\right| < 1\ \forall\omega\,, \tag{2.7}$$

[1]This description is written with discrete time systems, but is also valid for continuos time system.

this is a sufficient and necessary condition resulting from:

$$\|W_T(z)T(z,\boldsymbol{\rho})\|_\infty < 1 \;\&\; \left\| \frac{W_S(z)S(z,\boldsymbol{\rho})}{1+\Delta(z)W_T(z)T(z,\boldsymbol{\rho})} \right\|_\infty < 1,\ \|\Delta(z)\|_\infty \leq 1 \tag{2.8}$$

where the second inequality is found by considering the uncertain system in the sensitivity functions:

$$\frac{1}{1+\Big(1+\Delta(z)W_T(z)\Big)G_T(z)K(z,\boldsymbol{\rho})} = \frac{S(z)}{1+\Delta(z)W_T(z)T(z,\boldsymbol{\rho})} \tag{2.9}$$

By minimizing the cost function (2.1), the resulting controller $K(z,\hat{\boldsymbol{\rho}})$ guarantees the robust stability and nominal performance if $K(z,\hat{\boldsymbol{\rho}})$ comply with:

$$\|\boldsymbol{H}(z,\hat{\boldsymbol{\rho}})\|_\infty < 1\ . \tag{2.10}$$

This equation (2.10) corresponds to guarantee (2.6).

The robust performance test is quite similar to the cost function (2.1), since the only variation consists of the type of the norm. So, the general relationships between the norms help us to understand how to translate the inequality (2.10) for robust performance aim.

By considering the Cauchy-Schwartz inequality [120, Chapter 2], the norms inequalities can be written as:

$$\|\boldsymbol{d}\|_\infty \leq \|\boldsymbol{d}\|_2 \leq \|\boldsymbol{d}\|_1 \leq \sqrt{l}\,\|\boldsymbol{d}\|_2\ , \tag{2.11}$$

where $\boldsymbol{d} \in \mathbb{C}^{l\times 1}$. Figure 39 shows the rationale of the norm inequalities. In light of this, the robust performance test $\left\|\boldsymbol{H}(e^{j\omega},\boldsymbol{\rho})\right\|_1 < 1 \quad \forall\omega$ can be approximated by employing the 2-norm as:

$$\left\|\boldsymbol{H}(e^{j\omega},\boldsymbol{\rho})\right\|_2 < \frac{1}{\sqrt{2}} \cong 0.707 \quad \forall\omega\ . \tag{2.12}$$

where $\boldsymbol{H}(e^{j\omega},\boldsymbol{\rho}) \in \mathbb{C}^{2\times 1}$. Furthermore, to tie the equation approximate robust stability test (2.12) with the mixed sensitivity cost function (2.1), we need to consider the computation of the infinity norm for a dynamic system, computed as:

$$\|\boldsymbol{H}(z,\boldsymbol{\rho})\|_\infty = \max_\omega\ \bar{\sigma}\left(\boldsymbol{H}(e^{j\omega},\boldsymbol{\rho})\right)\ , \tag{2.13}$$

where $(\boldsymbol{H}(e^{j\omega},\boldsymbol{\rho})$ returns a vector of complex number evaluated in ω. By considering SISO systems, the infinity norm computation (2.13) becomes:

$$\|\boldsymbol{H}(z,\boldsymbol{\rho})\|_\infty = \max_\omega\ \left\|\boldsymbol{H}(e^{j\omega},\boldsymbol{\rho})\right\|_2\ . \tag{2.14}$$

This resolution shows a direct relation between the infinity norm and the 2-norm of a SISO dynamic system. Thus, by considering the equations (2.14) and (2.12), the robust performance controller can be obtained with the S/T mixed-sensitivity loop-shaping[b], if $K(z, \hat{\boldsymbol{\rho}})$ comply with:

$$\|\boldsymbol{H}(z, \boldsymbol{\rho})\|_\infty < \frac{1}{\sqrt{2}} . \tag{2.15}$$

Note that, this is a conservative approximation of the real robust performance test, because there exist some solutions that meet the robust performance test (2.7), but not agree with (2.15), as depicted in Figure 39.

[a]The robust performance test shows that the prerequisites are nominal performance and robust stability.
[b]With $S/T/KS$ mixed-sensitivity the robust performance is obtained by bounding the cost function with $\frac{1}{\sqrt{3}}$.

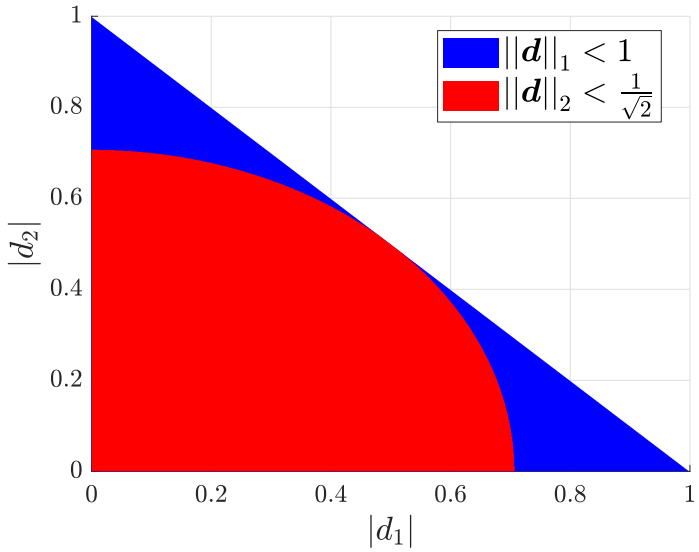


Figure 39: Comparison between $\|d\|_1 < 1$ and $\|d\|_2 < 1/\sqrt{2}$, where $\boldsymbol{d} \in \mathbb{C}^2$.

Since $G_T(z)$ is assumed to be unknown, practical use of (2.1) for robust control under the uncertainty description (2.2) requires the development of a plant model. Here we focus on the case where a data-driven model $\hat{G}_0(z)$ is identified from a set of n input/output data $\mathcal{D} = \{u(t), y(t)\}_{t=1}^n$ collected from an open-loop experiment on the plat, so that

$$y(t) = G_T(z)u(t) + e(t), \tag{2.16}$$

where n is the number of measurements and $e(t)$ is a random measurement noise satisfying the following assumption.

Assumption 2.1: Bounded noise assumption

The noise $e(t)$ in (2.16), acting at plant output, is possibly stochastic and norm-bounded with $|e(t)| < \bar{\delta}_e$, $\forall t$.

The randomness in $e(t)$ influences the estimate of $\hat{G}_0(z)$, thereby acting as a source of model uncertainty. We describe such uncertainty as in (2.2), where the boundedness of $\Delta(z)W(z)$ derives from the bounded nature of $e(t)$. The aim of this work is to provide an automatic data-driven tuning of the weight functions $W_S(z), W_T(z)$ to design a controller $K(z, \boldsymbol{\rho})$ by minimizing the mixed-sensitivity cost $J(\boldsymbol{\rho}, \hat{G}_0)$ in (2.1), with $\hat{G}_0(z)$ in place of $G_T(z)$, so that the feedback system $T(z)$ is robust to the modeling uncertainty endowed in the identification process under Assumption 2.1, described by the uncertain model (2.2).

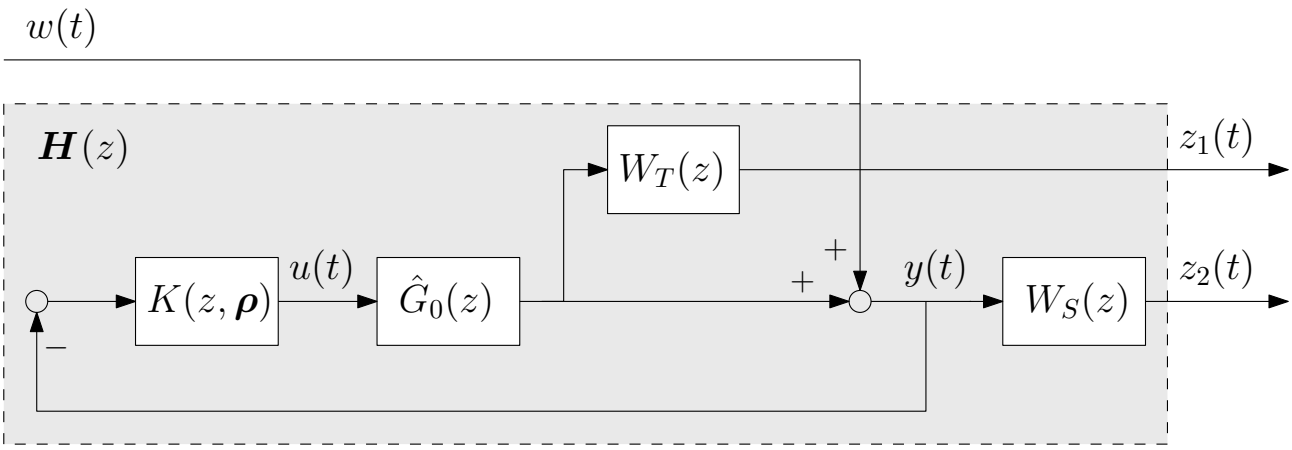


Figure 40: S/T mixed-sensitivity scheme with multiplicative uncertainty and identified model $\hat{G}_0(z)$ for the design of the controller $K(z, \rho)$. The term $W_T(z)$ represents the uncertainty weight function in the identification of $\hat{G}_0(z)$. Instead, the term $W_S(z)$ depicts the performance specification.

Figure 40 depicts the considered feedback system model with output multiplicative uncertain systems. This model can be employed to design a controller which:

- Attenuates the disturbance $d(t)$, with $w(t) = d(t) = \Delta(z)z_1(t)$. In this configuration the reference signal is $r(t) = 0\ \forall t$;

- Tracks a reference signal, with $w(t) = -r(t)$.

In our work, we consider the tracking problem.

Proof 2.1: Lower LFT representation for S/T mixed-sensitivity loop-shaping

By considering the lower LFT equation (1.37) and by studying the resulting traditional control synthesis scheme (depicted in Figure 13):

$$\boldsymbol{N}(z) = \left[\begin{array}{c|c} W_S(z) & -W_S(z)\hat{G}_0(z) \\ 0 & W_T(z)\hat{G}_0(z) \\ \hline 1 & -\hat{G}_0(z) \end{array}\right]. \tag{2.17}$$

By substituting the elements of the matrix $\boldsymbol{N}(z)$ in the lower LFT form, we obtain:

$$\begin{aligned} F_l(\boldsymbol{N}(z), K(z)) &= \begin{bmatrix} W_S(z) \\ 0 \end{bmatrix} + \begin{bmatrix} -W_S(z)\hat{G}_0(z) \\ W_T(z)\hat{G}_0(z) \end{bmatrix} K(z)\left(1+\hat{G}_0(z)K(z)\right)^{-1} \\ &= \begin{bmatrix} W_S(z) \\ 0 \end{bmatrix} + \begin{bmatrix} -W_S(z)\hat{G}_0(z)K(z)\hat{S}_0(z) \\ W_T(z)\hat{G}_0(z)K(z)\hat{S}_0(z) \end{bmatrix} \\ &= \begin{bmatrix} W_S(z)\left(1-\hat{T}_0(z)\right) \\ W_T(z)\hat{T}_0(z) \end{bmatrix} \\ &= \begin{bmatrix} W_S(z)\hat{S}_0(z) \\ W_T(z)\hat{T}_0(z) \end{bmatrix} \end{aligned} \tag{2.18}$$

where $\hat{S}_0(z) = \frac{1}{1+K(z)\hat{G}_0(z)}$ and $\hat{T}_0(z) = \frac{K(z)\hat{G}_0(z)}{1+K(z)\hat{G}_0(z)}$.

This is equal to the cost function of the S/T mixed-sensitivity loop-shaping.

Assumption 2.1 would call for a Set membership identification approach, in this framework there is no guarantee that $\hat{G}_0(z)$ is close to $G_T(z)$ in L_2-norm [10], which is required for the minimization of $J(\boldsymbol{\rho}, \hat{G}_0)$ to lead to the required robust control design, since modeling bias might still be present. Thus, we investigate the research problem using stochastic approaches, were $\hat{G}_0(z) \approx G_T(z)$ if the model is flexible enough. We focus our description on low-bias kernel-based methods, where the only assumption is on the gaussianity of the output noise $e(t)$.

Assumption 2.2: Data-generating system and model assumptions

In Assumption 2.1 we assume a bounded-amplitude noise, this prior information is not leveraged by standard kernel methods, which instead assume a Gaussian output noise. However, the price paid for ignoring this information is way lower than the price paid by using a model with high bias in robust control design routines, as shown in Section 2.3.

2.2 Data-driven design of mixed-sensitivity weights

This section shows the automatic design of mixed-sensitivity weights. Specifically, it employs the uncertainty information given by regularized kernel identification from a Bayesian regression point of view considering finite impulse response (FIR) models. The methodology is well described in sections 1.4.2, 1.4.3 and A.3; but we resume the methodology setup.

Consider (2.16) with $d(t) \sim \mathcal{N}\left(0, \sigma^2\right)$, where σ^2 is the Gaussian noise variance, and the FIR model of order m

$$\hat{G}(z, \boldsymbol{\beta}) = \sum_{i=1}^{m} g_i z^{-i}, \quad \boldsymbol{\beta} = [g_1\, g_2\, \ldots,\, g_m]^\top \,. \tag{2.19}$$

Assume that a prior distribution $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K})$ is placed on $\boldsymbol{\beta} \in \mathbb{R}^{m\times 1}$, where $\boldsymbol{K} \in \mathbb{R}^{m\times m}$ is chosen as a tuned-correlated kernel [6], where the (i, o) element of $\boldsymbol{K}$ is defined as $K_{io} \coloneqq \delta \cdot \alpha^{\max(i,o)}$ with $\delta > 0, 0 \leq \alpha < 1$ are the kernel hyperparameters (for further info see Table 8). The posterior distribution $\boldsymbol{\beta}|\boldsymbol{y} \sim \mathcal{N}\left(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\Sigma}}\right)$ where $\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\Sigma}}$ are computed by the closed forms, respectively, (A.17b) and (A.17c). Where:

$$\boldsymbol{\Phi} \coloneqq [\boldsymbol{x}(1),\, \boldsymbol{x}(2),\, \ldots\, \boldsymbol{x}(n)]^\top \in \mathbb{R}^{m\times n}, \tag{2.20a}$$

$$\boldsymbol{x}(t) \coloneqq [u(t-1),\, \ldots\, u(t-m)]^\top \in \mathbb{R}^{m\times 1} \tag{2.20b}$$

$$\boldsymbol{y} \coloneqq [y(1),\, y(2),\, \ldots\, y(n)]^\top \in \mathbb{R}^{n\times 1}. \tag{2.20c}$$

The kernel's hyperparameters $\boldsymbol{\eta} = [\alpha, \delta]^\top$ (along with the noise variance σ_e^2) can be estimated by employing an Empirical Bayes scheme, by maximizing the log-marginal likelihood of the data, see (A.18).

The parameters $\hat{\boldsymbol{\beta}}$ are identified by (A.17b). The nominal model is denoted as $\hat{G}_0(z) \coloneqq G(z, \hat{\boldsymbol{\beta}})$. Relying on (A.17c) to estimate the modeling uncertainty set, the next section presents the proposed scheme for data-driven tuning of $W_T(z)$ and $W_S(z)$.

Remark 2.2

The result of the kernel-based identification is the posterior distribution $\boldsymbol{\beta}|\boldsymbol{y}$, defined by the mean $\hat{\boldsymbol{\beta}}$ and variance $\hat{\boldsymbol{\Sigma}}$. These will be used in Section 2.2. Specifically, $\hat{\boldsymbol{\beta}}$ defines the nominal model $\hat{G}_0(z)$, while $\hat{\boldsymbol{\Sigma}}$ allows to design $\hat{W}_T(z)$ by estimating the uncertainty bound through a random sampling of the posterior distribution $\boldsymbol{\beta}|\boldsymbol{y}$.

2.2.1 Design of the stability weight

Let $G_p(z) \coloneqq G(z, \boldsymbol{\beta}_p)$, with $\boldsymbol{\beta}_p$ a random sample drawn from the posterior distribution $\boldsymbol{\beta}|\boldsymbol{y} \sim \mathcal{N}\left(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\Sigma}}\right)$. Define

$$\Omega\left(e^{j\omega}\right) \coloneqq \max_p \left| \frac{G_p\left(e^{j\omega}\right)}{\hat{G}_0\left(e^{j\omega}\right)} - 1 \right| , \tag{2.21}$$

where $\hat{G}_0(z) \coloneqq G(z, \hat{\boldsymbol{\beta}})$ is the nominal identified model. Robust stability against the multiplicative uncertainty model (2.2) requires that [26, Chapter 9]

$$\Omega\left(e^{j\omega}\right) \leq \left|W_T\left(e^{j\omega}\right)\right|, \quad \forall \omega \in [0, \pi f_s] , \tag{2.22}$$

where ω is specific pulse in rad/s and f_s is the sampling frequency. The magnitude of the least conservative $W_T(z)$ can be estimated evaluating (2.21) in a discrete grid of $n_m \in \mathbb{N}^+$ frequencies $\mathcal{W} = \left\{\omega_1, \omega_2, \ldots \omega_{n_m}\right\} \subseteq [0, \pi f_s]$ for a set of $n_p \in \mathbb{N}^+$ samples. A nonparametric sampled estimate of the magnitude of $W_T(z)$ is thus

$$\left|\hat{W}_T\left(e^{j\omega_m}\right)\right| = \Omega\left(e^{j\omega_m}\right), \quad \forall \omega_m \in \mathcal{W} . \tag{2.23}$$

A stable and proper parametric model $\hat{W}_T(z)$ can then be obtained by fitting a model of adequate order to the magnitude frequency points (2.23), taking care that the magnitude of the fitted model lies above (or is equal to) (2.23), see Section 2.4 for practical details.

The following proposition suggests how to select n_p[2].

Proposition 2.2: Uncertainty bound reconstruction

Define a fixed confidence level $\zeta \in (0, 1)$ and accuracy level $\varepsilon \in (0, 1)$. Let

$$n_p \geq \frac{1}{2\varepsilon^2} \log\left(\frac{2}{\zeta}\right) . \tag{2.24}$$

[2]The choice of n_m is not critical. It suffices for the frequency grid to be resolute enough.

Then, with probability $\geq 1 - \zeta$, it holds that

$$\left|\hat{W}_T\left(e^{j\omega_m}\right) - W_T\left(e^{j\omega}\right)\right| < \varepsilon, \quad \forall \omega, \omega_m \in \mathcal{W}\,.$$

Figure 41 represents a snippet of the lower bound curve of n_p with $\zeta = \varepsilon = [0.001, 0.002, ..., 0.05]$. The red highlighted point represents the chosen values in Section 2.3. The proof of this proposition is reported in Proof 2.2.

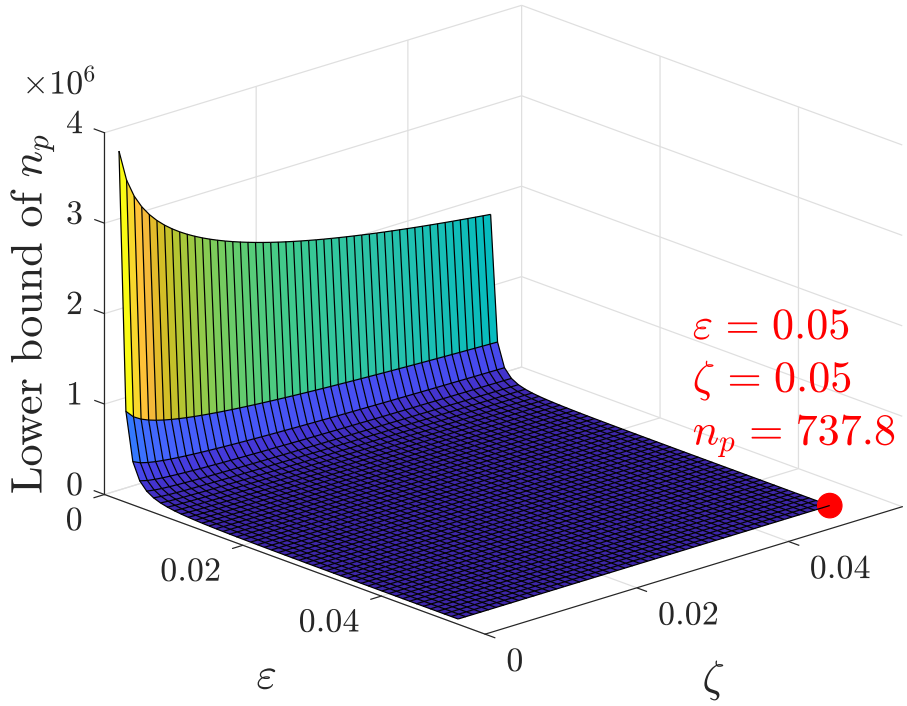


Figure 41: Lower bound curve of n_p with a subset of ε and ζ.

Proof 2.2: Chernoff bound

The proof of the Proposition 2.2 is a direct application of the *Hoeffding inequality* [32, Chapter 8]. Let n_p independent random variables $x_1, ..., x_{n_p}$, defined as:

$$x_i = \mathcal{I}_{\mathcal{B}_G}\left(\Delta^{(i)}(z)\right) \tag{2.25}$$

where:

- $\mathcal{I}_{\mathcal{B}_G}\left(\Delta^{(i)}(z)\right)$ is an indicator function associated with the good value set:

$$\mathcal{I}_{\mathcal{B}_G}\left(\Delta^{(i)}(z)\right) = \begin{cases} 1 & \text{if } \Delta^{(i)}(z) \in \mathcal{B}_G \\ 0 & \text{otherwise} \end{cases}; \tag{2.26}$$

- Good value set is $\mathcal{B}_G = \{\Delta(z) \in \mathcal{B}_\mathcal{A} : \|F_u(\boldsymbol{M}(z), \Delta(z))\|_\infty < \varpi\}^a$, where: $\mathcal{B}_\mathcal{A}$ is described in (1.31), $F_u(\boldsymbol{M}(z), \Delta(z))$ is the upper LFT defined in (1.8) and ϖ is a performance level;
- $\Delta^{(i)}(z)$ are the uncertainty samples;
- $i = 1, ..., n_p$.

Since $x_i \in [0, 1]$, letting $s_{n_p} = \sum_{i=1}^{n_p} x_i$, $\mathrm{E}[\cdot]$ the average operator and applying the two-sided Hoeffding inequality, we get the bound of the probability $\mathcal{P}\left\{\left|s_{n_p} - \mathrm{E}\left[s_{n_p}\right]\right)\right| \geq \varepsilon\right\}$ as:

$$\mathcal{P}\left\{\left|s_{n_p} - \mathrm{E}\left[s_{n_p}\right]\right)\right| \geq \varepsilon\right\} \leq 2e^{\frac{-2\varepsilon^2}{n_p}} . \tag{2.27}$$

where ε and n_p are defined in Proposition 2.2. Instead, putting $\hat{W}_T\left(e^{j\omega_m}\right) = \frac{s_{n_p}}{n_p}$ and $W_T\left(e^{j\omega}\right) = \mathrm{E}\left[\frac{s_{n_p}}{n_p}\right]$, the inequality (2.27) becomes:

$$\mathcal{P}\left\{\left|\hat{W}_T\left(e^{j\omega_m}\right) - W_T\left(e^{j\omega}\right)\right| \geq \varepsilon\right\} \leq 2e^{-2\varepsilon^2 n_p} . \tag{2.28}$$

Since $\zeta = \mathcal{P}\left\{\left|\hat{W}_T\left(e^{j\omega_m}\right) - W_T\left(e^{j\omega}\right)\right| \geq \varepsilon\right\}$ (as defined in Proposition 2.2), the inequality is:

$$\zeta \leq 2e^{-2\varepsilon^2 n_p} . \tag{2.29}$$

Doing so, the Chernoff bound follows straightforwardly.

[a] This set is valid for non-linear structured uncertainties, but in SISO systems with a single non-linear uncertainty it coincides to an unstructured uncertainty.

Algorithm 1 summarizes the steps for the design of the uncertainty weight $\hat{W}_T(z)$.

Algorithm 1: Design of $\hat{W}_T(z)$

Input: $\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\Sigma}}, \mathcal{W}, n_p$

1 $p = 0$
2 **while** $p < n_p$ **do**
3 Draw $\boldsymbol{\beta}_p \sim \mathcal{N}\left(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\Sigma}}\right)$ using $\boldsymbol{\beta}|\boldsymbol{y}$
4 Set $G_p = G(z, \boldsymbol{\beta}_p)$ as in (2.19)
5 **end while**
6 Compute $\left|\hat{W}_T\left(e^{j\omega_m}\right)\right|$ from (2.21) and (2.23) using $\mathcal{W}$
7 Fit a stable proper parametric model $\hat{W}_T(z)$ on frequency domain magnitude data in Step 7

Output: $\hat{W}_T(z)$

2.2.2 Design of the performance weight for nominal performance

As described in Section 2.2, the proposed method is applicable to a general class of systems. However, in practical applications, a second-order reference model is often sufficient to express the main dynamics of a desired closed-loop behaviour, in terms of step-response settling time and overshoot. For this reason, we consider a continuous-time second-order reference model $T_d(s)$ for the closed-loop system $T(s)$

$$T_d(s) \coloneqq \frac{\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2}. \tag{2.30}$$

where ω_n is the natural frequency of the poles and ξ is their damping factor, that can also be expressed as

$$\xi = \frac{|\ln(o)|}{\sqrt{\pi^2 + \ln^2(o)}}, \tag{2.31a}$$

$$\omega_n = \frac{\omega_c}{\sqrt{\sqrt{4\xi^4 + 1} - 2\xi^2}} = \frac{4}{\xi\ell}, \tag{2.31b}$$

with o, ℓ denoting the step response overshoot and settling time of $T(s)$, respectively, and ω_c denotes its critical frequency.

A simple approach to define the performance weight $W_S(s)$ in continuous time is to employ the inverse of the sensitivity function from the reference model (2.30), so that

$$W_S(s) \coloneqq (1 - T_d(s, o, \ell))^{-1} \ . \tag{2.32}$$

Then, $W_S(z)$ can be obtained by discretization of (2.32).

Remark 2.3

Doing so, we obtain a correspondence between the frequency domain and the time domain. Thus, the nominal performances of the resulting robust controller can be easily evaluated in both domains.

Since $\left|S\left(e^{j\omega}\right) + T\left(e^{j\omega}\right)\right| = 1 \ \forall\omega$, the robust performance condition under (2.2), see [26],

$$\max_{\omega} \ \left|W_S\left(e^{j\omega}\right) S\left(e^{j\omega}\right)\right| + \left|W_T\left(e^{j\omega}\right) T\left(e^{j\omega}\right)\right| < 1 \tag{2.33}$$

implies that [24]

$$\min\left(\left|W_S\left(e^{j\omega}\right)\right|, \left|W_T\left(e^{j\omega}\right)\right|\right) < 1 \quad \forall\omega \in [0, \pi f_s] \ . \tag{2.34}$$

Given $\omega_{c,\hat{W}_T} \in [0, \pi f_s]$ the critical frequency[3] of $\hat{W}_T(z)$ attained from Algorithm 1, relation (2.34) poses a performance limit on the bandwidth of $W_S(z)$, that cannot exceed of $\hat{W}_T(z)$. The automatic

[3]With the term "critical frequency" we denote the frequency where the magnitude of a transfer function crosses the 0 dB axis.

design of $W_S(z)$ aims to find the best values of (o, ℓ) so that $W_S(z)$ is stable and (2.3), (2.34) are satisfied.

Let $o_{\min} \neq 0$ be a user-defined minimum allowable overshoot level. The minimum allowable settling time $\ell_{\min}$ can be computed from (2.31) using $o_{\min}$ in (2.31a) and subsequently $\omega_{c,\hat{W}_T}$ in (2.31b). Then, the following multi-objective optimization problem is solved to design $\hat{W}_S(z)$:

$$(\mathcal{O}, \mathcal{L}) = \arg \min_{o,\ell} \ (o, \ell) \tag{2.35a}$$

$$\text{s.t.} \min_{\boldsymbol{\rho}} J\left(\boldsymbol{\rho}, \hat{G}_0, \hat{W}_T, W_S(o, \ell)\right) < 1 + \gamma \,, \tag{2.35b}$$

$$o_{\min} < o < 1 \,, \tag{2.35c}$$

$$\ell_{\min} < \ell < \ell_{\max} \,, \tag{2.35d}$$

where $(\mathcal{O}, \mathcal{L})$ are two sets of Pareto-optimal dominant solutions for the overshoot and settling time, respectively. Problem (2.35) is subject to the following constraints:

- (2.35b) requires the satisfaction of the robust stability and nominal performance condition (2.3a), based on the nominal model $\hat{G}_0(z)$ and the tuned stability weight $\hat{W}_T(z)$.

  Remark 2.4

  The sensitivity function has the property of: $S(e^{j\omega}) \to 1$ with $\omega \to \pi f_s$, therefore, $W_S(e^{j\omega})$ has the same behavior. Hence, the robust stability constraint is not feasible for frequencies near to πf_s. The constraint (2.35b) solves this problem by introducing a small positive slack quantity $\gamma \in \mathbb{R}^+$. This guarantees the feasibility of the optimization problem, also in spite of numerical inaccuracies.

- (2.35c) and (2.35d) bound the overshoot and settling time in feasible ranges. The value $\ell_{\max}$ can be defined by the user and it is not critical.

The estimates $(\hat{o}, \hat{\ell}) \in (\mathcal{O}, \mathcal{L})$ are chosen as the point closest (in Euclidean distance) to $(o_{\min}, \ell_{\min})$ in the space defined by overshoot and settling time, see Figure 42. Thus, the tuned performance weight $\hat{W}_S(s)$ is set as

$$\hat{W}_S(s) = \left(1 - T_d(s, \hat{o}, \hat{\ell})\right)^{-1} \tag{2.36}$$

and $\hat{W}_S(z)$ follows from discretization of (2.36). Algorithm 2 summarizes the steps for the tuning of the performance weight $\hat{W}_S(z)$. Finally, the designed controller $\hat{K}(z) \coloneqq K(z, \hat{\boldsymbol{\rho}})$ is found by solving (2.1) using $\hat{G}_0(z)$, $\hat{W}_T(z)$ and $\hat{W}_S(z)$.

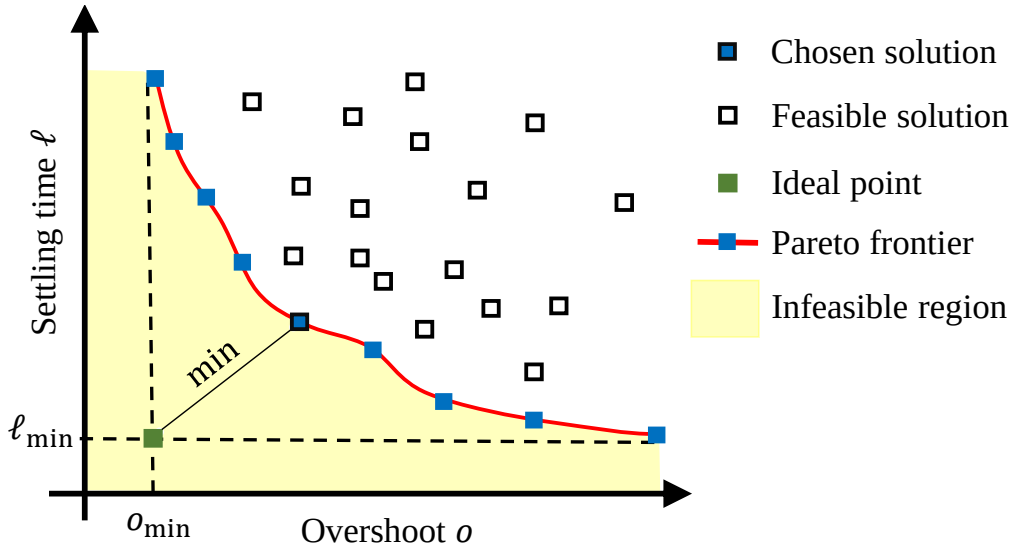


Figure 42: Multi-objective optimization with Pareto frontier and selection of the chosen solution in the set of dominant ones.

Remark 2.5: Performance weight stability

Each iteration of (2.35b), as well as discretization of (2.36), requires a stable and proper $W_S(z, o, \ell)$. To guarantee this, the following steps are followed:

- A continuous-time reference model $T_d(s)$ as in (2.30) is computed using the current values of (o, ℓ),
- The magnitude of (2.32) is evaluated into the frequencies grid in $\mathcal{W}$,
- A stable and proper parametric model is fit on frequency magnitude data, as in Step 8. of Algorithm 1.

See Section 2.4 for details.

2.2.3 Design of the performance weight for robust performance

The weight function $\hat{W}_S(z)$, designed with (2.36), cannot be used with robust performance aims, since the performance requirements will be too stringent. This is due to the feasibility problem explained in Remark 2.4. Thus, the performance requirements must be embedded in $W_S(s)$ in a different way. Another simple technique directly defines: low-frequency gain l_g, critical frequency ω_c and high-frequency gain h_g of a discrete filter with a monotonic gain profile. Specifically, we bound the domain

Algorithm 2: Estimation of $\hat{W}_S(z)$ for nominal performance and robust stability

Input: $\hat{W}_T(z)$, γ, $\mathcal{W}$, $o_{\min}$, $\ell_{\max}$

1 Set $\omega_{c,\hat{W}_T}$ as the critical frequency of $\hat{W}_T(z)$. If $\hat{W}_T(z)$ does not cross 0 dB, we set $\omega_{c,\hat{W}_T} = \pi f_s$

2 Set $\xi = \dfrac{|\ln(o_{\min})|}{\sqrt{\pi^2 + \ln^2(o_{\min})}}$, $\omega_n = \dfrac{\omega_{c,\hat{W}_T}}{\sqrt{\sqrt{4\xi^4+1} - 2\xi^2}}$, $\ell_{\min} = \dfrac{4}{\xi\omega_n}$ using (2.31)

3 Solve (2.35) to get $(\mathcal{O}, \mathcal{L})$, considering Remark 2.5

4 Select point estimates $(\hat{o}, \hat{\ell}) \in (\mathcal{O}, \mathcal{L})$ closest to $(o_{\min}, \ell_{\min})$

5 Compute $\hat{\xi} = \dfrac{-\ln(\hat{o})}{\sqrt{\pi^2 + \ln^2(\hat{o})}}$, $\hat{\omega}_n = \dfrac{4}{\hat{\xi}\hat{\ell}}$ using (2.31)

6 Set the reference model $T_d(s) = \dfrac{\hat{\omega}_n^2}{s^2 + 2\hat{\xi}\hat{\omega}_n s + \hat{\omega}_n^2}$

7 Set $\hat{W}_S(s) = (1 - T_d(s))^{-1}$ as in (2.36)

8 Evaluate $\left|\hat{W}_S(j\omega_m)\right|$ using $\mathcal{W}$

9 Fit a stable proper parametric model $\hat{W}_S(z)$ on frequency domain magnitude data in Step 8

Output: $\hat{W}_S(z)$

of the high frequency gain and critical frequency by: $h_g \in \left[h_{g\_\min}, h_{g\_\max}\right]$ and $\omega_c \in \left[\omega_{c\_\min}, \omega_{c,\hat{W}_T}\right]$, where: $\omega_{c\_\min}, l_g, h_{g\_\max}, h_{g\_\min}$ are user-defined[4].

Furthermore, a robust performance controller designed with S/T mixed-sensitivity loop-shaping usually guarantees $J(z, \boldsymbol{\rho}) < 1/\sqrt{2}$, but this is an approximation and it overconstraints the problem, as described in Proposition 2.1. Our proposed method instead employs directly the sufficient and necessary condition $\|\boldsymbol{H}(z, \boldsymbol{\rho})\|_1 < 1$.

Doing so, the optimization problem (2.35) becomes:

$$(\mathcal{Z}, \mathcal{Q}) = \arg \min_{\omega_c, h_g} (-\omega_c, -h_g) \tag{2.37a}$$

$$\text{s.t. } \|\boldsymbol{H}(z, \boldsymbol{\rho})\|_1 < 1, \tag{2.37b}$$

$$\omega_{c\_\min} < \omega_c < \omega_{c,\hat{W}_T}, \tag{2.37c}$$

$$h_{g\_\min} < h_g < h_{g\_\max}. \tag{2.37d}$$

Remark 2.6

For each iteration of (2.37), the controller parameters $\boldsymbol{\rho}$ are obtained by minimizing $J(\boldsymbol{\rho}, \hat{G}_0(z))$.

Where:

[4] l_g has a high value as desired. $h_{g\_\max}, h_{g\_\min} < 0dB$, where the max value is near to $0dB$. Instead, $h_{g\_\min}$ and $\omega_{c\_\min}$ are low as desired. Their choices are not critical if the two domains are big enough.

- $(\mathcal{Z}, \mathcal{Q})$ are respectively critical frequency and high-frequency gain sets of Pareto-optimal dominant solutions;

- The constraint (2.37b) guarantees robust performance feature;

- The cost functions aim to find the highest feasible values of high-frequency gain and critical frequency, therefore the objective functions become the minimization of $-\omega_c$ and $-h_g$. Remember that the inverse of the uncertainty weight shapes the band and gain of the sensitivity function since the constraint $|1/\hat{W}_S(e^{j\omega})| < \left|\hat{S}_0(e^{j\omega})\right|$ must be guaranteed for all frequencies. Therefore, if $1/\hat{W}_S(z)$ has an high gain profile, then the allowable gain profile of $\hat{S}_0(z)$ is high, otherwise with low values of $|1/\hat{W}_S(e^{j\omega})|$, the $\left|\hat{S}_0(e^{j\omega})\right|$ is low (under the gain profile of the inverse of the uncertainty weight). Since the ideal choice of a controller aims to shape the sensitivity function as $S(z) = 0$ [121], the goal is to obtain the highest values of h_g and ω_c.

Finally, the estimates $(\hat{\omega}_c, \hat{h}_g) \in (\mathcal{Z}, \mathcal{Q})$ are found trough the minimum Euclidean distance from $(\omega_{c,\hat{W}_T}, h_{g\_\max})$, then the performance weight is defined as a filter with a monotonic gain profile, defined by: l_g, $\hat{\omega}_c$ and $\hat{h}_g$. The practical aspects to design $W_S(z)$, from $\hat{\omega}_c$ l_g and $\hat{h}_g$, will be shown in 2.4.

Algorithm 3: Estimation of $\hat{W}_S(z)$ for robust performance

Input: $\hat{W}_T(z)$, γ, $\mathcal{W}$, $h_{g\_\max}$, $h_{g\_\min}$, $\omega_{c\_\min}$, l_g

1 Compute $\omega_{c,\hat{W}_T}$, the critical frequency of $\hat{W}_T(z)$. If $\hat{W}_T(z)$ does not cross 0 dB, we set $\omega_{c,\hat{W}_T} = \pi f_s$

2 Solve (2.37) to get $(\mathcal{Z}, \mathcal{Q})$, considering Remark 2.5 and Remark 2.6

3 Select the couple $(\hat{\omega}_c, \hat{h}_g) \in (\mathcal{Z}, \mathcal{Q})$ closest to $(\omega_{c,\hat{W}_T}, h_{g\_\max})$

4 Set $\hat{W}_S(z)$ as a filter with monotonic gain profile, defined by: l_g, $\hat{\omega}_c$ and $\hat{h}_g$

Output: $\hat{W}_S(z)$

Remark 2.7

Both multi-objective problems (2.35) and (2.37) can be cast to a single-objective problem by multiplying the less important cost function by zero. This leads to high customization according to user needs. For instance, if the user would like to obtain a controller that guarantees robust stability and nominal performance and minimize the settling time without caring about overshoot the cost functions of (2.35) becomes $\arg\min_{o,\ell} (o, 0 \cdot \ell)$. Doing so, the Pareto-front will be composed of a single point and so step 4 of Algorithm 2 or step 3 of Algorithm 3 are negligible.

2.3 Numerical example

2.3.1 Experimental setup

Consider the following benchmark system [122]

$$G_T(z) = \frac{0.28261z + 0.50666}{D(z)}, \tag{2.38a}$$

$$y(t) = G_T(z)u(t) + e(t) = y_T(t) + e(t), \tag{2.38b}$$

$D(z) = z^4 - 1.41833z^3 + 1.58939z^2 - 1.31608z + 0.88642$, sampled at $T_s = 1/f_s = 0.01\,\text{s}$. We simulated $n = 5000$ data from (2.38) using a zero mean white noise input and a bounded zero mean white noise disturbance $e(t)$ with SNR $= \text{var}\left[y_T(t)\right]/\text{var}\left[e(t)\right] = 25$. The boundaries were set so that no saturation in $e(t)$ was present. The first 1000 data are discarded to remove the transient effects from the data. The FIR order for kernel identification is set as $m = 100$.
We compared kernel identification with PEM using an Output Error (OE) model set $\mathcal{G}$, considering two cases:

- $G_T \notin \mathcal{G}$ (*PEM undermodeling*): the OE orders are chosen from 1 to 3 selected by Akaike Information Criterion (AIC);
- $G_T \in \mathcal{G}$ (*PEM full*): the OE orders are chosen from 1 to 10 selected by AIC;

In the PEM cases, we follow the same procedure devised in Section 2.2 for estimating the uncertainty weight $W_T(z)$. In both PEM and kernel cases, we assume to know the true value of the noise variance[5]. By selecting $\zeta = 0.05$ and $\varepsilon = 0.05$ the number of sampled system is set as $n_p = 738$ following (2.24), and the number of sampled frequencies is set to $n_m = 600$, logarithmically spaced in the range $\left[10^{-3}, \pi f_s\right]$ rad/s.
For all the three identification settings, we test the following weights design strategies:

- **DS1).** Manual design of $W_S(z)$ and $W_T(z)$;
- **DS2).** Manual design of $W_S(z)$ and automatic design of $W_T(z)$ using the approach of Section 2.2.1;
- **DS3 - Proposed).** Automatic design of $W_S(z)$ and $W_T(z)$.

For the design strategy DS1), we use the following settings: the $W_T(z)$ shape is set as a discrete filter with a gain monotonic profile, made by low-frequency gain of 10 dB and a high-frequency gain of

[5]This assumption is not critical since the noise variance can be estimated with good accuracy from data.

40 dB. The cut off frequency is fixed to a bit less than the first resonant peak of $G_T(z)$, i.e. 40 rad/s. The manual setting of $W_S(z)$ is established by imposing a settling time of $\ell = 1$ s and an overshoot of $o = 0.01$.

The chosen controller structure is :

$$K(z, \boldsymbol{\rho}) = k_p + k_i \frac{T_s}{z-1} + k_d \frac{1}{N_d + \frac{T_s}{z-1}}, \tag{2.39}$$

where $\boldsymbol{\rho} = \left[k_p, k_i, k_d, N_d\right]^\top \in \mathbb{R}^{4\times 1}$ and T_s is the sampling time. The performance of the controllers $\hat{K}(z)$, designed by solving problem (2.35) with $\gamma = 0.1, o_{\min} = 0.01, \ell_{\max} = 5$ s in all the aforementioned conditions, are evaluated by drawing $n_v = 200$ random open-loop systems from the Gaussian distributions $\mathcal{N}(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\Sigma}})$ of PEM and kernel parameters estimates, centered at the respective estimate $\hat{G}_0(z)$[6].

2.3.2 Robust stability and nominal performance results and discussion

Figure 43 shows the estimate $\hat{G}_0(z)$ of the open-loop transfer function $G_T(z)$. The PEM-undermodeling (denoted as PEM u.m.) case is able to fit only the first resonance peak, while the full PEM and kernel identifications attain almost perfect results.

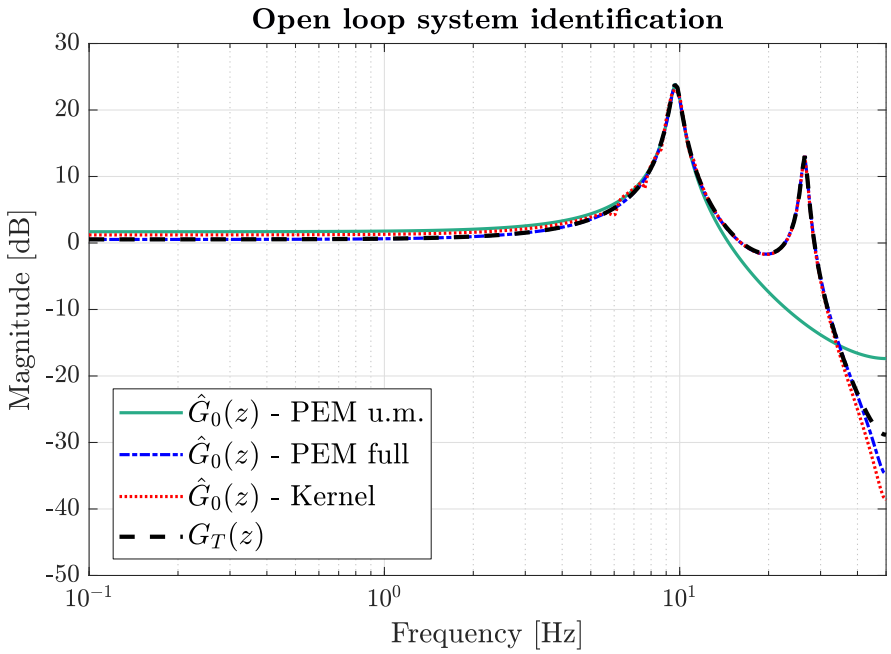
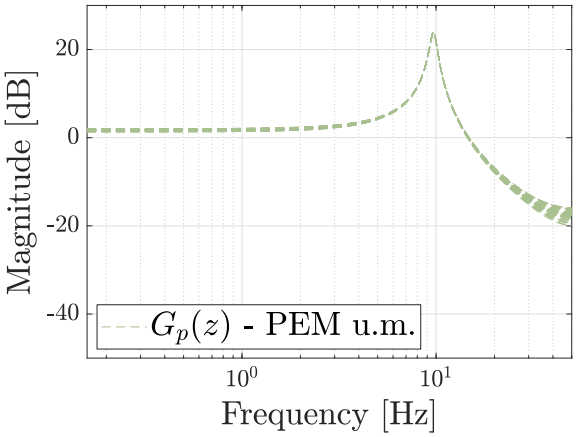


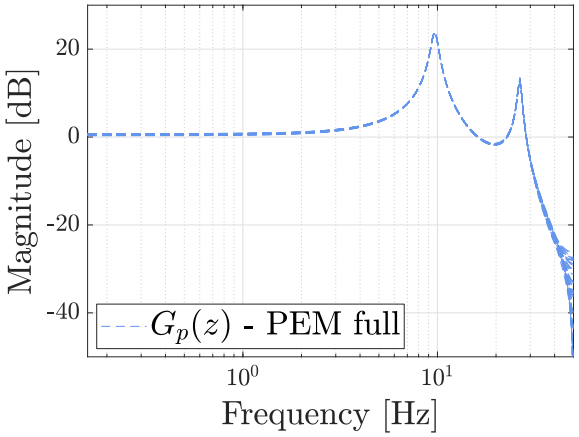
Figure 43: Identification of the transfer function $G_T(z)$. The PEM undermodeling approach cannot model the second resonance peak. The PEM full approach perfectly represents the true system due to its exact structure selection. The kernel approach attains almost perfect results, apart of a slight bias at low frequencies due to its regularized nature.

[6]For the PEM case, this assumes that $G_T \in \mathcal{G}$.

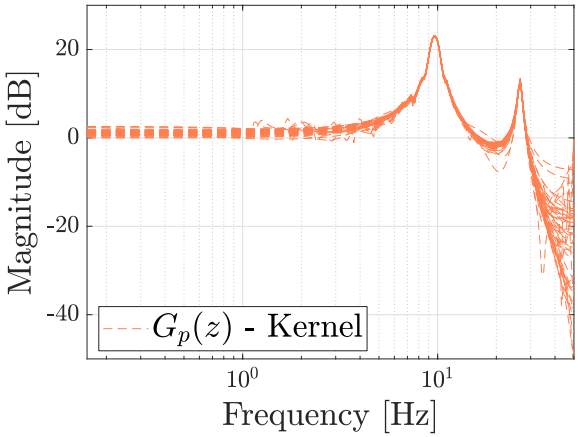
The images of Figure 44 depict frequency responses of some $G_p(z)$ extracted from the estimated distributions of all three identified systems.



(a) $G_p(z)$ extracted from PEM with undermodeling identification.



(b) $G_p(z)$ extracted from PEM-full identification.



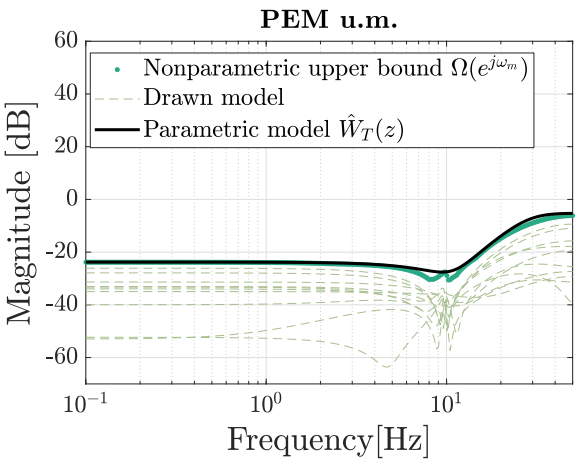
(c) $G_p(z)$ extracted from kernel identification.

Figure 44: Frequency response of 50 $G_p(z)$**.**

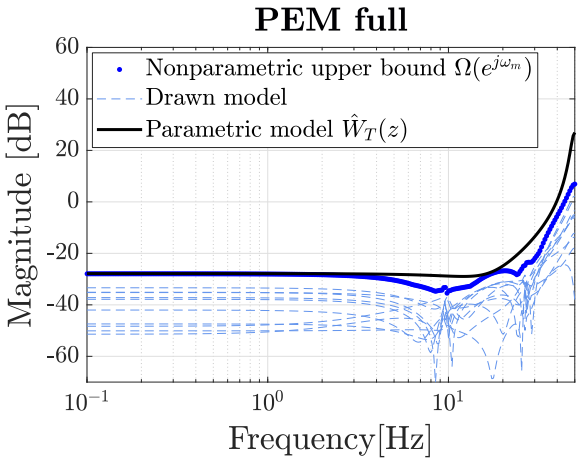
The images of Figure 45 show the estimates of the uncertainty weight $\hat{W}_T(z)$, using a second order proper transfer function model. In all the cases, the estimate lies above (or it is equal to) the upper bound $\Omega(e^{j\omega_m})$ in (2.23), which is computed considering n_p extractions of systems from the respective parameters distribution of each identification method. We notice how PEM-undermodeling attains a similar uncertainty level as PEM-full; however, in the former case also a bias contribution is present in the identified model. The bias is negligible in the PEM-full and kernel cases, the latter of whose attains the highest modeling uncertainty[7].

The images of Figure 46 show:

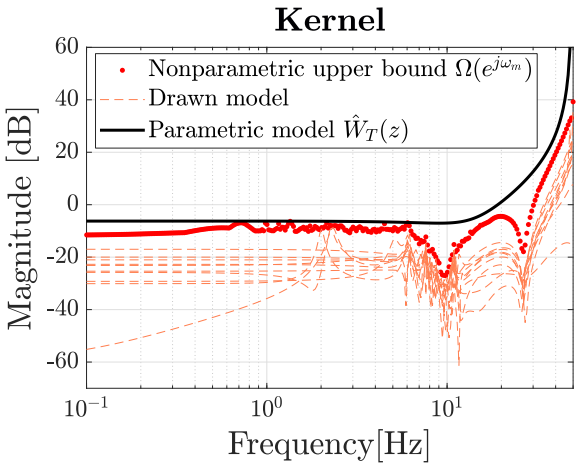
[7]We remark here that the PEM-full case is able to perfectly represent the true system $G_T(z)$.



(a) $\hat{W}_T(z)$ obtained with PEM with undermodeling identification.



(b) $\hat{W}_T(z)$ obtained with PEM-full identification.



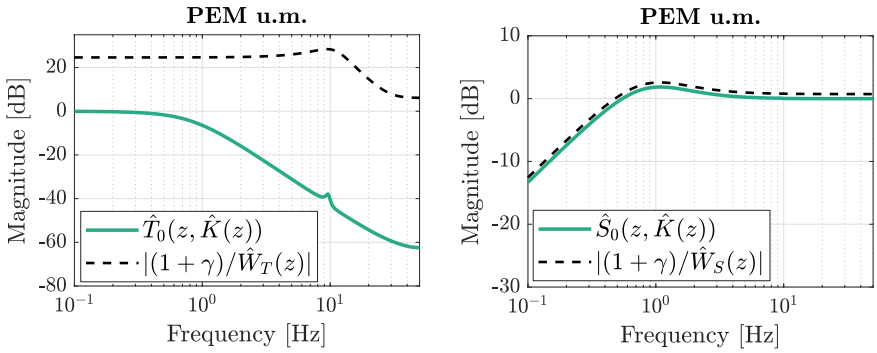
(c) $\hat{W}_T(z)$ obtained with kernel identification.

Figure 45: Uncertainty weight $W_T(z)$ estimation. (Continuous line) Magnitude of the parametric model $\hat{W}_T(z)$. (Dots) Nonparametric estimate $\Omega(e^{j\omega_m})$. (Dashed line) The gain of the frequency response of $\frac{G_p(z)}{\hat{G}_0(z)} - 1$, where the systems $G_p(z)$ are drawn from the sampling distribution of the parameters estimates. The dashed lines are a subset of the n_p systems for graphical purposes.

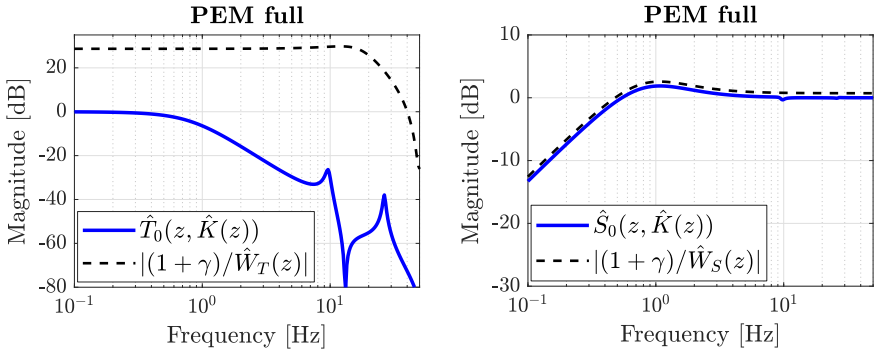
- Left: a comparison of the complementary sensitivity function $\hat{T}_0(z, \hat{K}(z))$ obtained with the designed controller $\hat{K}(z)$ against its bound $1/\hat{W}_T(z)$ with margin γ defined in problem (2.35);

- Right: a comparison of the sensitivity function $\hat{S}_0(z, \hat{K}(z))$ against its bound $1/\hat{W}_S(z)$ with margin γ.

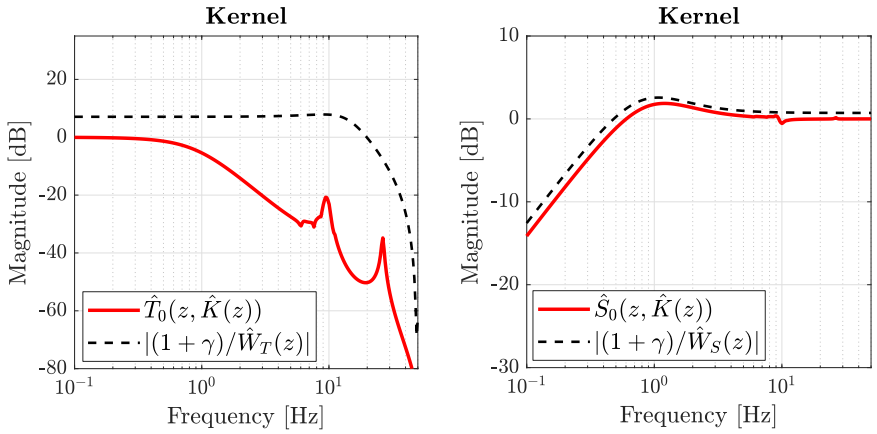
First, we note how all the controllers provide robust stability and nominal performance. Second, the controller designed in the PEM-undermodeling case lead to a closed-loop system with higher bandwidth with respect to the PEM-full and kernel cases. This derives from constraint (2.35d), where the lack of the second resonance peak in the PEM-undermodeling nominal model makes possible to the controller to attain an higher control bandwidth.



(a) Constraints analisys of PEM undermodeling.



(b) Constraints analisys of PEM-full.



(c) Constraints analisys of kernel identification.

Figure 46: (Left) (continuous line) Closed-loop complementary sensitivity function $\hat{T}_0(z)$ using the estimated controller $\hat{K}(z)$. (Dashed line) Inverse of the uncertainty weight $\hat{W}_T(z)$, that should lie above $\hat{T}_0(z)$ for robust stability with margin γ. (Right) (continuous line) Closed-loop sensitivity function $\hat{S}_0(z)$ using the estimated controller $\hat{K}(z)$. (Dashed line) Inverse of the performance weight $\hat{W}_S(z)$, that should lie above $\hat{S}_0(z)$ for nominal performance with margin γ.

Figure 47 evaluates n_v closed-loop unit step responses $\hat{y}_{\text{step}}(t)$ in terms of the Integral Absolute Error (IAE) over a period of 4 s:

$$\text{IAE} = \sum_{t=1}^{4\cdot T_s} |1 - \hat{y}_{\text{step}}(t)| \,. \tag{2.40}$$

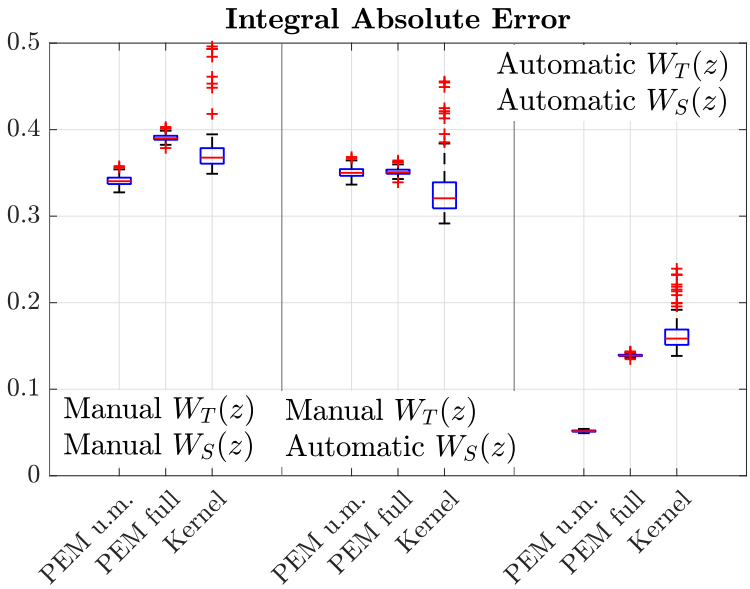
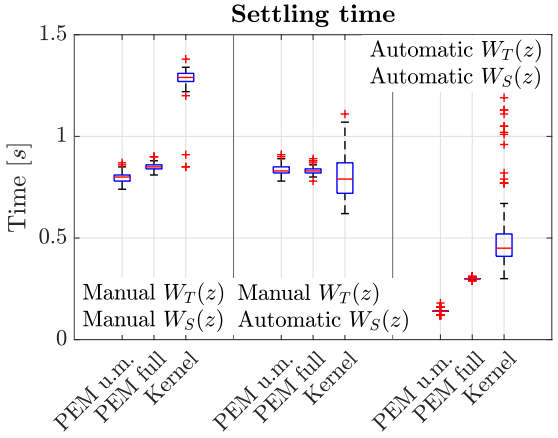
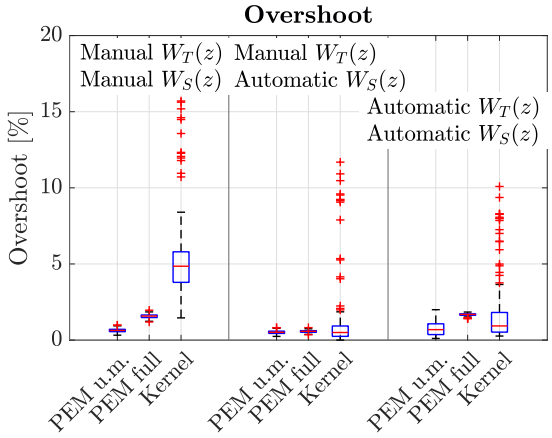


Figure 47: Integral Absolute Error of the closed-loop unit step response, over $n_v = 200$ randomly sampled systems from the estimated parameters distribution. PEM u.m. stands for PEM undermodeling. (Left) Design strategy DS1): manual design of $W_S(z)$ and $W_T(z)$. (Center) Design strategy DS2): manual design of $W_S(z)$ and automatic design of $W_T(z)$. (Right). Design strategy DS3): automatic design of $W_S(z)$ and $W_T(z)$.



(a) Settling time index analisys.



(b) Overshoot index analisys.

Figure 48: Performance indices analysis of the closed-loop unit step response, over $n_v = 200$ randomly sampled systems from the estimated parameters distribution. PEM u.m. stands for PEM undermodeling. (Left) Design strategy DS1): manual design of $W_S(z)$ and $W_T(z)$. (Center) Design strategy DS2): manual design of $W_S(z)$ and automatic design of $W_T(z)$. (Right). Design strategy DS3): automatic design of $W_S(z)$ and $W_T(z)$.

No relevant difference in performance can be observed in the design conditions DS1) and DS2). In the DS3) design, all the identification approaches lead to a significantly better closed-loop performance. As already noticed, the PEM undermodeling case is the best one, due to its higher control bandwidth. The PEM-full design performs slightly better than the kernel method, but it has access to the true system parametrization.

Remark 2.8

The automatic $\hat{W}_T(z)$ design leads to a high performance $\hat{W}_S(z)$, since the manual choice of $W_T(z)$ usually is more conservative than the real system needs.

Results in Figure 47 might suggest that it is better to have a model with modeling bias for robust control design. However, these performance are evaluated to systems sampled from the parameters distribution centered on the $\hat{G}_0(z)$ found by each respective identification method. Instead good, if a modeling bias is present, the true system $G_T(z)$ might not be included in the uncertainty set considered for robust control. Figure 49 compares the closed-loop unit step responses of the design strategy DS3) using each identification scheme. The PEM-undermodeling scheme performs fine on the nominal model $\hat{G}_0(z)$, but it is unstable on $G_T(z)$. The PEM full and kernel methods, instead, performs equally good on the respective $\hat{G}_0(z)$ and $G_T(z)$. Again, the PEM-full approach has an advantage in having access to the true system parameterization.

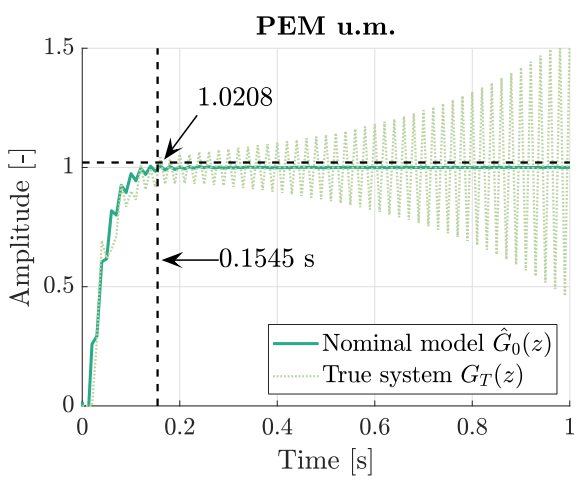
Remark 2.9: Bias and model error modeling

The presence of a large bias in the PEM-undermodeling and of a low bias in the kernel cases should bring to the employment of a model error model approach to describe the modeling uncertainty set (see Section 1.4.3). However, the main use of kernel methods is exactly to avoid this bias modeling due to its negligible nature with respect to the modeling variance. In the PEM-undermodeling case, an uncertainty region that considers also a model error model would bring $\hat{W}_T(z)$ to increase considerably, thus attaining a lower critical frequency $\omega_{c,\hat{W}_T}$ that would have impacted negatively the design of the performance weight $\hat{W}_S(z)$, leading to a not adequate controller tuning.

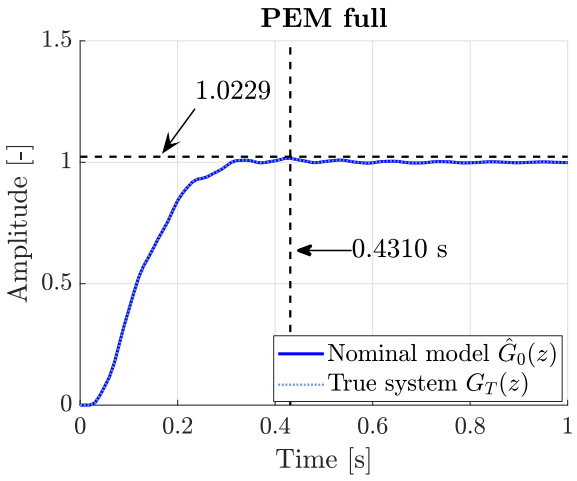
2.3.3 Robust performance results and discussion

The robust performance simulation is done with the same setup of the robust stability and nominal performance. Furthermore, the parameters of the robust performance multi-objective problem is set as: $l_g = 120dB$, $h_{g\_min} = -50dB$ and $\omega_{c\_min} = 1^{rad}/_s$. We simulated only our proposed method with the three identification setup, since the disadvantages of manual design are already explained in the robust stability and nominal performance simulation.

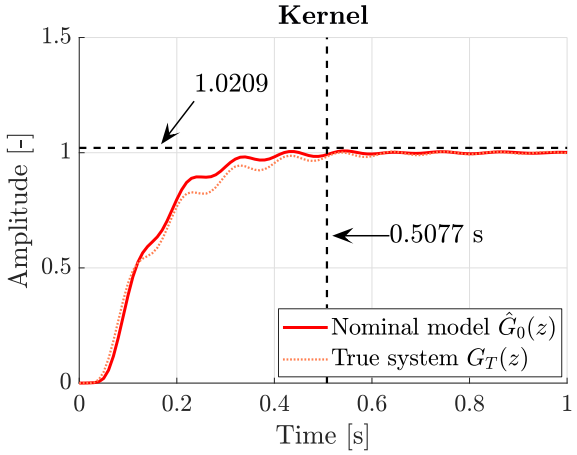
The images of Figure 50 show the $\|\boldsymbol{H}(z, \boldsymbol{\rho})\|_1$ (straight line) resulting from the robust performance design with data-driven S/T mixed-sensitivity loop-shaping. All colored curves are below to the constraint (dashed line), this means that the three controllers (one of each identification setup) guarantee that all systems, that belong to the uncertainty model, agree with the performance requirements. For a



(a) Closed-loop step responses of PEM with undermodeling.



(b) Closed-loop step responses of PEM-full.



(c) Closed-loop step responses of kernel.

Figure 49: (Continuous line) Closed-loop unit step responses on nominal model $\hat{G}_0(z)$. (Dotted line) Closed-loop unit step responses on the true system $G_T(z)$. (Dashed lines) Performance requirements $(\hat{o}, \hat{\ell})$ estimated from problem (2.35). In the PEM-undermodeling case the designed controller makes the true system unstable. In the PEM-full case, the response on the nominal model and true system overlap.

further analysis, the images of Figure 54 illustrate the robust performance tests (2.8) for all three cases by using $\hat{W}_T(z)$. As we expect, all solutions meet the two robust performance constraints. Another graphical representation of the robust performance condition is depicted in the images of Figure 51: these show the sensitivity function of all dynamic systems of the validation set. Since the controller has the robust performance aim, all sampled curves are less than $\hat{W}_S(z)$. Nevertheless, due to the bias in the PEM undermodeling identification, this is misleading, since the performance of $G_T(z)$ in the time domain are different respect to the identified uncertain model.

Figure 52 reports the IAE (see equation (2.40)) distribution for the PEM undermodeling (left), PEM full (center) and kernel (right) of the n_v closed-loop unit step responses $\hat{y}_{step}(t)$. The black and green stars represent the IAE of the closed-loop unit step responses of the real system coupled with the

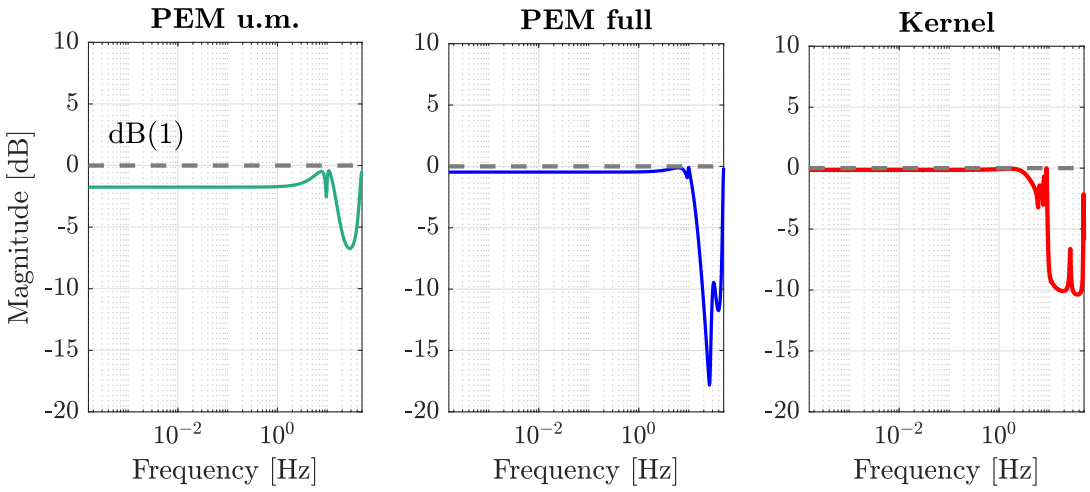


Figure 50: Analysis of the robust performance test $\left|\hat{W}_S(z)\hat{S}_0(z,\hat{K}(z))\right| + \left|\hat{W}_T(z)\hat{T}_0(z,\hat{K}(z))\right| < 1$ with all identification methods. Specifically, the images represent the results obtained with the three identification methods: (Left) PEM undermodeling (Center) PEM-full and (Right) kernel-based.

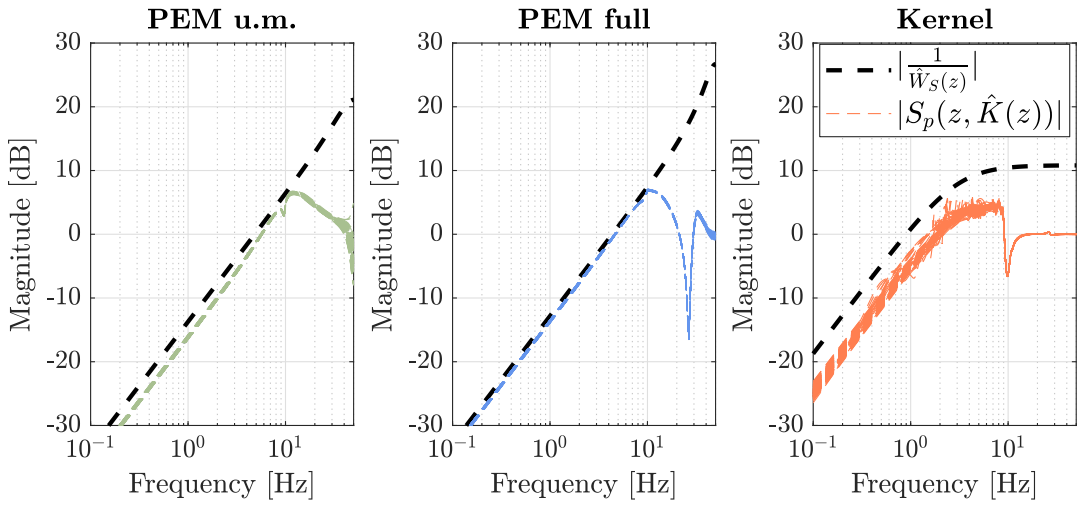


Figure 51: Performance analysis of the sensitivity functions computed by the sampled system from the validation set, denoted as $S_p(z,\hat{K}(z)) = \frac{1}{1+G_p(z)\hat{K}(z)}$. All sampled systems are compared to the inverse of the performance weight. Specifically, the images represent the results obtained with the three identification methods: (Left) PEM undermodeling (Center) PEM-full and (Right) kernel-based.

controllers resulted from the three mixed sensitivity loop-shaping, one for each identification setting. Using the kernel and PEM full identification methods, the star belongs to the validation distribution of the n_v closed-loop system. *Instead, with the PEM undermodeling the star does not belongs to the distribution, since the bias cause the instability of the true closed-loop system and so the overshoot and settling time of the true step response is out of scale.* The images of Figure 53 depict the same representation of Figure 52, but considering the settling time (Figure 53a) and overshoot (Figure 48b) indices.

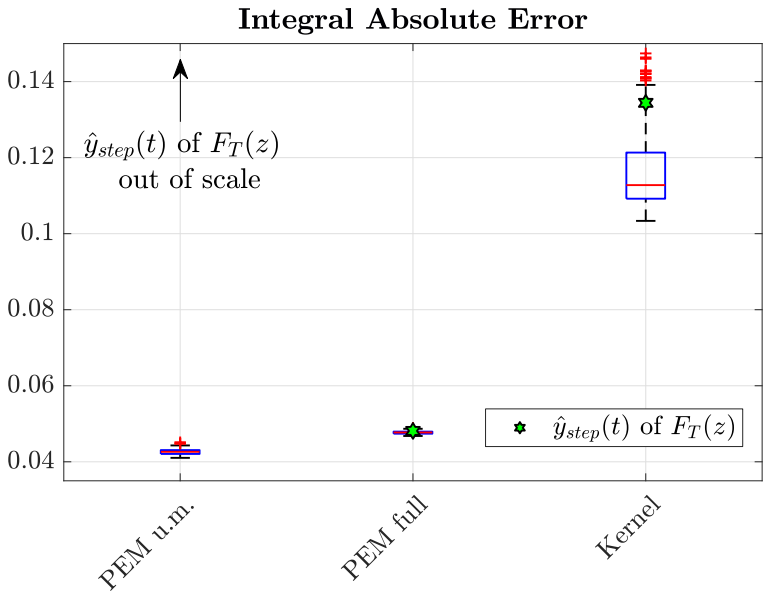
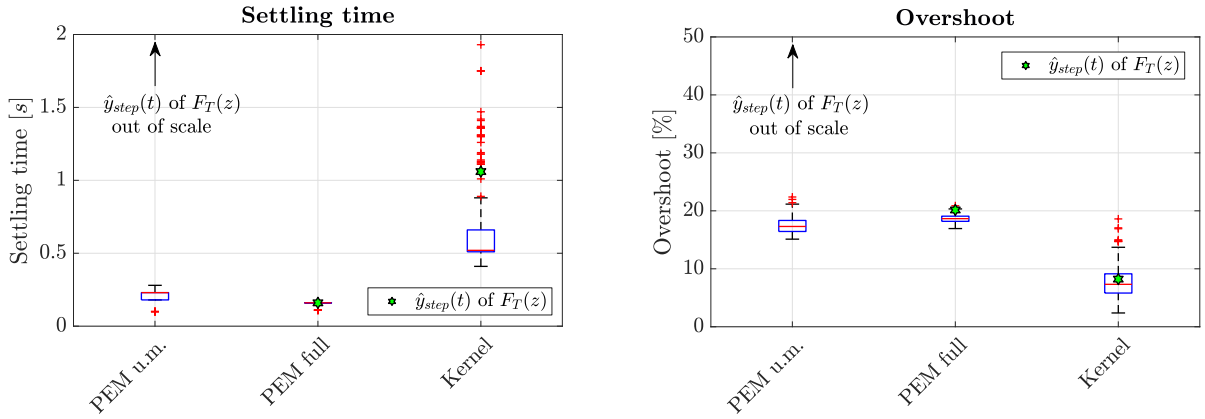


Figure 52: Integral Absolute Error of the closed-loop unit step response, over $n_v = 200$ sampled systems from the estimated parameters distributions. PEM u.m. stands for PEM undermodeling. The black and green stars correspond to the performance indices obtained with $G_T(z)$. The PEM u.m. star is out of scale.



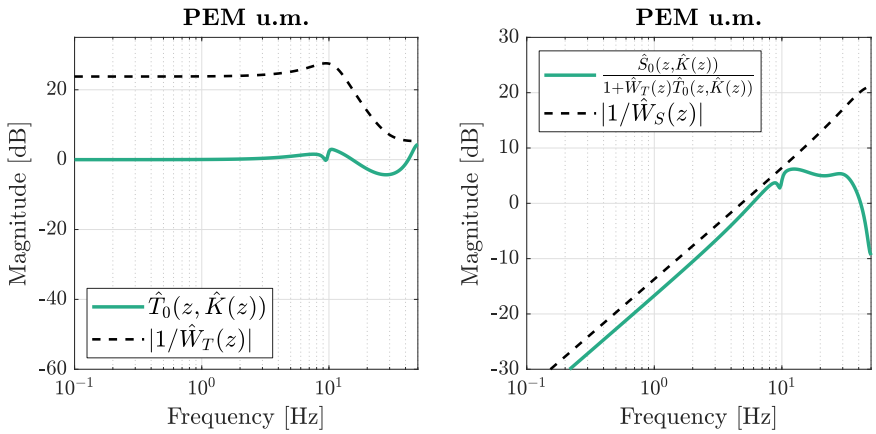
(a) Settling time index analisys.

(b) Overshoot index analisys.

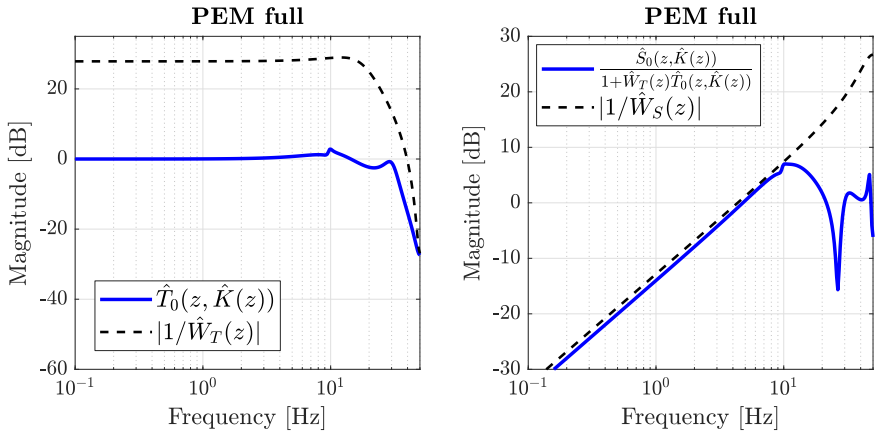
Figure 53: Performance indices analysis of the closed-loop unit step response, over $n_v = 200$ randomly sampled systems from the estimated parameters distribution. PEM u.m. stands for PEM undermodeling. The black and green stars correspond to the performance indices obtained with $G_T(z)$. The PEM u.m. stars are out of scale.

Remark 2.10

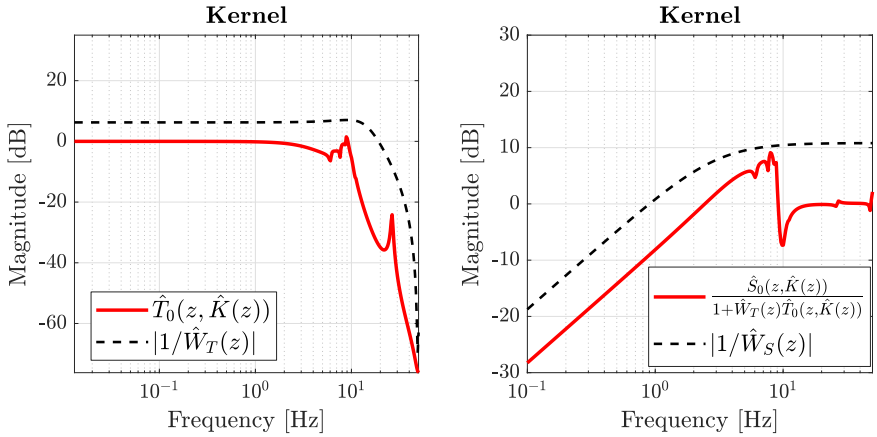
The proposed method for the robust performance problem gives an automatic design of the performance requirements, found by the optimization problem (2.37). These are not so interpretable as in the nominal performance and robust stability setting. Indeed, we cannot check if whole distribution complies with some time domain constraints.



(a) Constraints analisys of PEM undermodeling.



(b) Constraints analisys of PEM-full.



(c) Constraints analisys of kernel identification.

Figure 54: (Left) (continuous line) Closed-loop complementary sensitivity function $\hat{T}_0(z)$ using the estimated controller $\hat{K}(z)$. (Dashed line) Inverse of the uncertainty weight $\hat{W}_T(z)$, that should lie above $\hat{T}_0(z)$. (Right) (continuous line) Uncertain closed-loop sensitivity function $\frac{\hat{S}_0(z)}{1+\hat{W}_T(z)\hat{T}_0(z,\hat{K}(z))}$. (Dashed line) Inverse of the performance weight $\hat{W}_S(z)$, that should lie above the uncertain closed-loop sensitivity function.

2.4 Computational aspects

This section describes practical aspects that might be of interest to the practitioner. The results were obtained using Matlab software packages.

2.4.1 Computation of $\hat{G}_0(z)$

In the PEM cases, we fixed an OE(n_b, n_f) model structure with n_b, n_f the numerator and denominator orders, respectively, with 0 pure input/output delay. The search for the optimal model is performed by fixing the same $n_{\text{oe}} = n_b = n_f$ value for both the model orders.

In the kernel identification, first the kernel hyperparameters are estimated by optimizing the marginal likelihood developed with the Cholesky decomposition (A.19), assuming a known noise variance. The optimization is performed by `fmincon` where positiveness bound are placed on the hyperparameters. Then, a FIR model of order $m = 100$ is estimated.

2.4.2 Computation of $\hat{W}_T(z)$

The uncertainty weight estimation is based on fitting the frequency domain data $\Omega(e^{j\omega_m})$ in (2.21) with the function `fitmagfrd`, that estimates a stable proper continuous time transfer function model, solving a log-Chebyshev magnitude filter design [123, Chapter 6]. The function allows constraints on the filter magnitude, so we imposed that $|\hat{W}_T(s)| > \Omega(e^{j\omega_m}) + \epsilon$, with $\epsilon = 2.2 \cdot 10^{-16}$. We also weighted 100 times more the fit to the frequencies in the range $\left[10^{-3}, 10^{0}\right]$ rad/s. The order of $\hat{W}_T(s)$ is fixed to 2. Then, $\hat{W}_T(s)$ is converted to discrete time using the `'matched'` option of the `c2d` command.

2.4.3 Computation of $\hat{W}_S(z)$

The estimation of the performance weight is the most computational cumbersome part of the proposed algorithm, The resolution of robust stability and nominal performance problem (2.35) is implemented using the function `gamultiobj` with a limit of 50 iterations. The nonlinear constraint (2.35b) requires to solve an $\mathcal{H}_\infty$ control design problem with $\hat{W}_T(z)$ the actual value of $\hat{W}_S(z)$. This is solved by the `hinfstruct` function, considering 10 different initializations of the optimization procedure. This command solves the $\mathcal{H}_\infty$ norm with a fixed controller structure.

The design of $\hat{W}_S(z)$ follows this rationale. First, a continuous time reference model $T_d(s)$ (2.30) is defined using the actual values of $(\hat{o}, \hat{\ell})$. The magnitude of $W_S(s) = 1 - T_d(s)$ is sampled in the frequency grid $\mathcal{W}$. Then, a continuous-time transfer function of order 20 is fit with the `fitmagfrd` command (with default options). This guarantees that the resulting transfer function is stable and proper. Note that the samples are injected to the function as lower bound data. The function is then converted to discrete time obtaining $\hat{W}_S(z)$, using the `'matched'` option of the `c2d` command.

The robust performance problem (2.37) is solved with the computational aspects as the nominal performance and robust stability, while the design of $\hat{W}_S(z)$ from l_g, h_g, ω_c is done with the `makeweight` command specifying the sampling time and default options.
For both problems the final designed controller is again fit with the `hinfstruct` command, with $\hat{G}(z)$, $\hat{W}_T(z)$ and $\hat{W}_S(z)$.

2.5 Conclusion

We presented a data-driven mixed-sensitivity control design approach for SISO LTI systems embedded with automated tuning of the weighting functions. The approach leverages concepts from Robust identification and methods for assessing the bias and variance of the estimated model. We show how kernel methods can cope well with the bias problem, allowing an estimation of the model uncertainty region without the explicit need for building a model error model as in standard literature. We propose two multi-objective optimization problems for tuning the performance weight: one that guarantees the nominal performance and robust stability and another that designs a robust performance controller. The former optimization problem allows to design the performance weight whose requirements are interpretable for the designer and given in terms of closed-loop unit step response overshoot and settling time. The performance weight resulting from the latter is less interpretable since the requirements are given in terms frequency parameters of a monotonic gain profile filter.

Chapter 3. Data-driven mixed-sensitivity loop-shaping for multi-model systems

This chapter presents an extension of the proposed method explained in Chapter 2. In particular, the data-driven S/T mixed-sensitivity loop-shaping deals only with the uncertainty introduced by the measurement noise. In some practical applications, the proposed method is not enough flexible. This extension aims to design a robust controller which guarantees robust stability to deal with multiple sources of uncertainties.

Remark 3.1

This chapter describes the theoretical contributions of the proposed method. Instead, the practical application will be described in Section 5.

3.1 Motivation

Often, some complex systems are afflicted by multiple sources of uncertainty, in addition to the plant model uncertainty. These translate into a large variation of the model parameters or of the model structure. Usually, in the control theory literature, this typology of models is designed as a *multi-model dynamic system*. Specifically, this is defined as the combination of a finite number of simple local models endowed by an uncertainty region [124, Chaper 9]. This region is called *validity region* and it is defined by choosing an upper bound that constraints the euclidean norm of the model error. To assess which validity region contains the local model, a *validity function* is defined. This can be modeled as a probability function or as a fuzzy function. The latter is applicable if the validity regions are seen as fuzzy sets [125]. Note that, the fuzzy logic theory is useful with a priori qualitative knowledge.

The multi-model can be casted to a single global model by using an interpolation technique. This procedure is complex and it needs of local models with same model structure. However, it allows to simplify the control design, since the global model is more easy to handle than the multi-model.

In literature, the multi-model design is also used to represent the nonlinear systems with multiple linear local models that approximates the true systems in different operational points [124, Chaper 9]. Note that each local model, endowed by its validity region, can be seen as an uncertain system (Section 1.1), so the multi-model dynamic system can be represented also as an $\Delta - M$ model by using the Linear Fractional Transformation. The elements of the diagonal of the matrix Δ contain the information of the validity region. Instead, $\boldsymbol{M}(s)$ is a matrix in which each element of the diagonal is the nominal

transfer function of a local model. In doing so, the robust control techniques can be used. An example of this methodology applied to a flight control system is described in [126]. The main disadvantage of this method is to select the weighting filters for each local model. An alternative technique is to design a matrix of controllers, i.e. one for each local model. This is called *nonlinear structured controller*. The so called *two-stage interpolation* algorithm is employed to limit the usage of the active controllers. It selects which controllers are active in accordance with which local models are currently used [124, Chaper 9]. The latter control design technique is closely related to the gain-scheduling controller design. Typically, these methods employ a fixed single or multi-loop control structures. The gain values of the controllers are embedded in lookup tables and the functions that selects the gain values are called *scheduling variables*. A withdraw of the gain scheduling control is that it does not guarantee any stability or performance during a rapid change of the variables [127].

Our proposed method identifies a single system endowed with a global output multiplicative unstructured uncertainty, which assesses the variability region of all local models, through multiple kernel-based identifications. In particular, it avoids the design on multiple weighting filters and it simplifies also the interpolation technique of local models. The resulting single robust controllers avoids the usage of the scheduling variables.

3.2 Data-driven weights design for S/T mixed-sensitivity loop-shaping for multi-model systems

Consider a multi-model system, the goal is to design a fixed-order controller $K(z, \boldsymbol{\rho})$, parametrized by the parameters vector $\boldsymbol{\rho} \in \mathbb{R}^{n_\rho \times 1}$, by minimizing the S/T mixed-sensitivity cost function (2.1), as described in Section 2.1.1.

With multi-model systems, the problem shifts from unstructured uncertainty modeling to structured uncertainty modeling with linear and nonlinear uncertainties. As suggested by [16, Chaper 7], this general structured uncertainty can be also represented as a general output multiplicative[1] unstructured uncertainty, provided that the uncertainty poles (if exist), which belong to the uncertainty set, do not change the half-planes (from right half plane to left half plane and vice-versa)[2]. This consideration plays a key role to cast the multi-model dynamic system into a single general system endowed

[1]The multiplicative uncertainty is preferred to the additive uncertainty model because their numerical values are more informative.

[2]If the system has this characteristic, the unstructured uncertainty should be designed as inverse multiplicative output model.

with unstructured uncertainty that models both parameter variations and model variance. Thus, the innovation aims to identify this general model by exploiting the low-bias kernel-based identification. Note that, the experimental design is fundamental. Thus, we assume that the user only knows the *boundary configurations*, i.e. the configurations which are considered as the limit operational point. This assumption is not essential but allows us to handle the problem in a simpler way.

Remark 3.2

From a practical point of view, the boundary configurations knowledge is more common to recognize with respect to the traditional approach, where the parametric model is evaluated by its parameters range.

Figure 55 shows the frequency response of the boundary configurations. The boundary configurations are denoted with $G_{BU}(z)$ the highest condition, represented in the red line, and $G_{BL}(z)$ for the lowest condition, depicted in the blue line. The highlighted yellow region contains the other configurations.

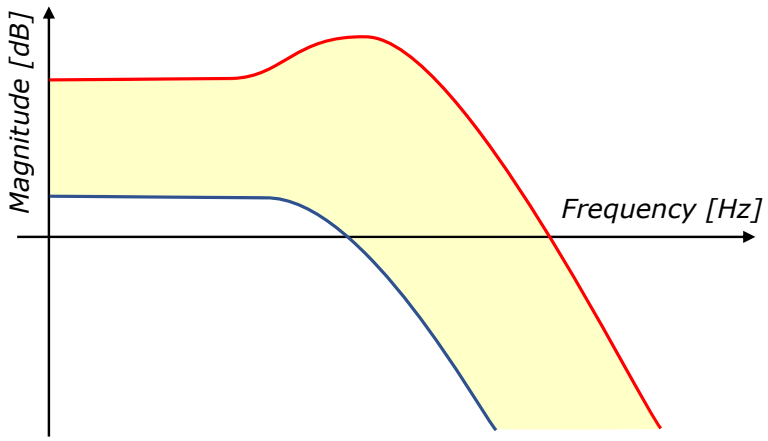


Figure 55: Example of the boundary conditions frequency response.

The experimental design involves two experiments, one for each condition, linked by a transient sector. The overall dataset is chosen as:

$$\mathcal{D}_c = \left\{ u(1), ..., u(n), 0, ..., 0, u(1), ..., u(n), y_{BU}(1), ..., y_{BU}\left(n+\frac{b}{2}\right), y_{BL}(1), ..., y_{BL}\left(n+\frac{b}{2}\right) \right\} , \quad (3.1)$$

where b is the number of samples that belong to the null sector. The transient sector is set up as a null sector, which avoids possible discontinuities between the two configurations. The input given to the two conditions is quite similar since the only variation is that: the upper boundary condition input ends with half of the null sector, instead the lower boundary condition begins with the half of the null sector.

Thus, using the kernel-based identification, the following dynamic systems are identified:

- $\hat{G}_{BU}(z)$ by employing the sub-dataset $\mathcal{D}_u = \{u(1), ..., u(n), y_{BU}(1), ..., y_{BU}(n)\}$;
- $\hat{G}_{BL}(z)$ by employing the sub-dataset $\mathcal{D}_l = \{u(1), ..., u(n), y_{BL}(1), ..., y_{BL}(n)\}$;
- $\hat{G}_{0g}(z)$ by employing the dataset $\mathcal{D}_c$. This represents the average dynamic system between the boundary conditions[3,4].

Remark 3.3

If the boundary conditions are not known, this procedure is easily scalable to design a single huge experiment which contains all configurations. Then, the above multiple identification procedures can be adapted.

A nonparametric sampled estimation of the magnitude of the overall uncertainty is computed by exploiting the equations (2.21) (2.23), where:

- The parameters vector of the high-order FIR $G_p(z)$ is sampled from the two posterior distributions: $\boldsymbol{\beta}_{BU}|\boldsymbol{y}_{BU} \sim \mathcal{N}\left(\hat{\boldsymbol{\beta}}_{BU}, \hat{\boldsymbol{\Sigma}}_{BU}\right)$ and $\boldsymbol{\beta}_{BL}|\boldsymbol{y}_{BL} \sim \mathcal{N}\left(\hat{\boldsymbol{\beta}}_{BL}, \hat{\boldsymbol{\Sigma}}_{BL}\right)$;
- The number of extraction n_p (see the proposition 2.2) are equally sampled between the two distributions;
- The nominal dynamic system used in (2.21) is replaced by the mean dynamic system $\hat{G}_{0g}(z)$.

Therefore, the computation of the overall nonparametric sampled estimate of the magnitude of $W_T(z)$ becomes:

$$\left|\hat{W}_T(e^{j\omega_m})\right| = \max_p \left|\frac{G_p(e^{j\omega_m})}{\hat{G}_{0g}(e^{j\omega_m})} - 1\right|, \quad \forall \omega_m \in \mathcal{W}\,. \tag{3.2}$$

After, as for SISO systems, a stable and proper parametric model is obtained by fitting a model to the resulting magnitude, as in Section 2.2.1.

Remark 3.4

The proposed procedure is convenient due to the low-bias nature of the kernel-based identification, otherwise, in presence of bias, the construction of the general distribution is not reliable and therefore, the resulting robust controller is useless.

Finally, the automatic design of the performance weight for nominal performance (described in Section 2.2.2) or robust performance (described in Section 2.2.3) is applied using the average dynamic system $\hat{G}_{0g}(z)$ and the overall uncertainty $\hat{W}_T(z)$.

[3]The kernel-based identification applied to a dataset composed of two experiments performed on two different systems returns a model that is the average of the two systems.

[4]Note that the average model shouldn't represent real systems, but this is used only as a mathematical tool.

Algorithm 4: Design of $\hat{W}_T(z)$ and $\hat{W}_S(z)$ for multi-model systems

Input: $\hat{G}_{0g}(z)$, $\hat{\boldsymbol{\beta}}_{BU}$, $\hat{\boldsymbol{\Sigma}}_{BU}$, $\hat{\boldsymbol{\beta}}_{BL}$, $\hat{\boldsymbol{\Sigma}}_{BL}$ $\mathcal{W}$, n_p

1 $p = 0$
2 **while** $p < n_p/2$ **do**
3 Draw $\boldsymbol{\beta}_p \sim \mathcal{N}\left(\hat{\boldsymbol{\beta}}_{BU}, \hat{\boldsymbol{\Sigma}}_{BU}\right)$ using $\boldsymbol{\beta}_{BU}|\boldsymbol{y}_{BU}$
4 Set $G_p = G(z, \boldsymbol{\beta}_p)$ as in (2.19)
5 **end while**
6 **while** $p < n_p/2$ **do**
7 Draw $\boldsymbol{\beta}_p \sim \mathcal{N}\left(\hat{\boldsymbol{\beta}}_{BL}, \hat{\boldsymbol{\Sigma}}_{BL}\right)$ using $\boldsymbol{\beta}_{BL}|\boldsymbol{y}_{BL}$
8 Set $G_p = G(z, \boldsymbol{\beta}_p)$ as in (2.19)
9 **end while**
10 Compute $\left\|\hat{W}_T\left(e^{j\omega_m}\right)\right\|$ from (3.2) using $\mathcal{W}$
11 Fit a stable proper parametric model $\hat{W}_T(z)$ on frequency domain magnitude data obtained in Step 12
12 Perform Algorithm 2 or Algorithm 3 to compute $\hat{W}_S(z)$ for robust stability and nominal performance or robust performance

Output: $\hat{W}_T(z)$, $\hat{W}_S(z)$

Remark 3.5

This methodology is also applicable to designing a fault-tolerant controller. The two boundary configurations represent the healthy and faulty states. It is possible since the fault condition is a priori known. Therefore, the robust controller works in both states, without any unexpected behaviour. The main advantage of a robust methodology employed to design a fault-tolerant controller is that the resulting robust controller does not change according to plant conditions [128, 48].

3.3 Conclusion and final remarks

We presented an extension of the methodology described in the previous chapter. The robust controller deals with the control of multi-model systems. The rationale consists of designing a general experiment, which contains information about the boundary conditions (given by the knowledge of the system). The boundary conditions allow to simplify the problem, but their knowledge is not mandatory. Then, the automatic uncertainty weight is given by multiple kernel-based system identification. Doing so, the multi-model system can be represented as an uncertain LTI SISO system, where the uncertainty contains both variance uncertainty and parameters uncertainty. This general uncertain system is employed to automatically design the weight functions employed in the S/T

mixed-sensitivity loop-shaping since the overall uncertainty is converted into an unstructured output multiplicative uncertainty. In Chapter 5, our proposed method is tested in a real multi-model system.

Chapter 4. Data-driven robust residual generator

This chapter presents another theoretical contribution. The idea is to design a robust residual generator based on the model uncertainty information, designed by the automatic uncertainty weight design, proposed in Chapter 2. The robust residual generator is modeled by the stable coprime factorization methodology. Specifically, we employ the Approximate Fault Detection Problem (AFDP) [27]. Finally, we test the proposed method on a benchmark problem to evaluate the diagnosis performances.

4.1 Motivation

Model-based fault diagnosis employs a model that acts as a digital twin of the fault-free plant. The model is developed by a model identification procedure. In this way, the model uncertainties are implicitly accepted, as described in Section 1.1. Usually, in literature, the uncertainties can be seen as an additive fictitious noise[1]. With this insight, the robust counterparts of the three fault diagnosis methods, described in Section 1.3, can be employed to generate a robust residual generator which decouples the noise with respect to the residual signal. Note that, the robustness of a residual generator is important because allows for reducing false alarms.

In fault diagnosis literature, the uncertainty information is considered known and modeled in $G_v(z)$ (the transfer function from noise $v(t)$ to the output $y(t)$). In this setting, the so-called Approximate Fault Detection Problem (AFDP) is applicable, since $v(t) \neq 0$. This method guarantees that the generated residual is insensitive to disturbance and low sensible to noise. As reviewed in Section 1.3.1, AFDP designs a filter by minimizing a cost function which is proportional to the sensitivity of the residual to a fault and inversely proportional to the uncertainties, described as unknown input noise. In [74], the authors solved AFDP by employing the robust synthesis method. Instead, in [76] and [75], the robust residual generator is coupled with a robust controller both designed with robust control synthesis.

By modeling the uncertainties as unknown but bounded noise, the Set membership identification can be used to develop a robust fault diagnosis technique, as proposed in [129, 130]. Specifically, these solve the problem of robustness by the passive technique (see Section 1.3.4). The paper [131] proposes a method that reconstructs the input and output signals of $\Delta(z)$ block in an LFT representation. In

[1]The authors of [28] models the uncertainty as fictitious disturbance, but the meaning does not change, since the only difference is that the disturbance can be fully decoupled from the residuals, instead the input noise cannot be fully decoupled, but only reduced.

particular, by assuming that the uncertainty block is bounded by 1, the output signal must be less than the input in a fault-free case, otherwise, the system has a fault.
We propose a method which exploits the kernel-based identification to obtain a nominal model endowed by its uncertainty quantity. Therefore, using the LFT representation, i.e. modeling the uncertainty as an unstructured output multiplicative uncertainty, the Approximate Fault Detection Problem is solved by employing the identified uncertainty. The resulting robust residual generator is obtained without exploiting any user's knowledge. Furthermore, we propose an automatic threshold design to compensate for the remaining noise components in the residual signal.

4.2 Data-driven robust residual design

Let a SISO LTI dynamic system by considering an additive output noise term and the fault:

$$y(t) = G_T(z)u(t) + G_f(z)f(t) + e(t)\,, \tag{4.1}$$

with:

- $f(t)$ represents the unknown fault signal;
- $G_f(z)$ is a stable transfer function which describes the relation between the fault and the output $y(t)$ signals. Usually, with a sensor fault $G_f(z) = 1$, instead with an actuator fault $G_f(z) = G_T(z)$. Since $G_T(z)$ is not available $G_f(z)$ is chosen equal to the identified nominal model $\hat{G}_0(z)$;
- $e(t)$ is the bounded additive noise signal (see Remark 2.1).

If we consider an uncertain system modeled by an output multiplicative uncertainty, the equation (4.1) becomes:

$$y(t) = \hat{G}_0(z)\Big(1 + \Delta(z)W_T(z)\Big)u(t) + G_f(z)f(t) + e(t)\,, \tag{4.2}$$

where: $\|\Delta\|(z) \leq 1$ represents the uncertainty (unknown and bounded), $W_T(z)$ is a proper and stable transfer function which describes the uncertainty weight function and $\hat{G}_0(z)$ is the nominal transfer function.
As described in [27, Chapter 2], the uncertainties are modeled as a fictitious noise $v(t)$, so the output signal can be written as:

$$y(t) = \hat{G}_0(z)u(t) + G_v(z)v(t) + G_f(z)f(t)\,, \tag{4.3}$$

where $G_v(z)$ is a transfer function between the unknown noise $v(t)$ and $y(t)$.

By matching the two output representations (4.2) and (4.3), we implicitly impose the following equality:

$$G_v(z)v(t) = \hat{G}_0(z)\Delta(z)W_T(z)u(t) + e(t)\,, \tag{4.4}$$

where the noise term contains both output noise and fictitious noise. Our proposed method employs the kernel-based identification to estimate $\hat{G}_0(z)$ and $\hat{W}_T(z)$, through the algorithm 1 from the dataset $\mathcal{D} = \{u(1), ..., u(n), y(1), ..., y(n)\}$ modeled by (4.1), where: n number of measured sampled, $y(t)$ noisy measurements in healthy state and $u(t)$ is the input signal.

We estimate the uncertainty weight $\hat{W}_T(z)$ as in 2.2.1, thus the equality (4.4) becomes:

$$G_v(z)v(t) = \hat{G}_0(z)\hat{W}_T(z)u(t) + e(t)\,. \tag{4.5}$$

Thanks to this equality, we set $G_v(z) = \begin{bmatrix}\hat{G}_0(z)\hat{W}_T(z) & 1\end{bmatrix}$ and $v(t) = \begin{bmatrix}u(t) & e(t)\end{bmatrix}^\top$. In light of this, the robust residual generator can be designed by rewriting the cost function of AFDP (1.55), as:

$$\hat{Q}(z) = \max_{Q(z)} \frac{\left\|Q(z)\hat{M}(z)G_f(z)\right\|_\infty}{\left\|Q(z)\hat{M}(z)\begin{bmatrix}\hat{G}_0(z)\hat{W}_T(z) & 1\end{bmatrix}\right\|_\infty}\,, \tag{4.6}$$

where $\hat{M}(z)$ is derived from the left coprime factorization of $\hat{G}_0(z)$ and $Q(z) \in \mathcal{RH}_\infty$ is the post filter. The optimization problem (4.6) can be solved as proposed in [27, Chapter 5.3]. The authors assume that the denominator is infinity norm bounded, this meets our Assumption 2.1 on the additive noise applied to $y(t)$. Specifically, the filter synthesis goal is to find the optimal post filter that maximizes the fault sensitivity, given:

$$\left\|Q(z)\hat{M}(z)\begin{bmatrix}\hat{G}_0(z)\hat{W}_T(z) & 1\end{bmatrix}\right\|_\infty \leq \gamma\,, \tag{4.7}$$

with a priori known $\gamma \geq 0$. Thus, the optimal filter produces $\beta > 0$, such that:

$$\beta = \max_{Q(z)} \left\{ \left\|Q(z)\hat{M}(z)G_f(z)\right\|_\infty \,\middle|\, \left\|Q(z)\hat{M}(z)\begin{bmatrix}\hat{G}_0(z)\hat{W}_T(z) & 1\end{bmatrix}\right\|_\infty \leq \gamma \right\}\,. \tag{4.8}$$

The value β/γ is a decoupling performance index, with $\gamma = 0$ the AFDP corresponds to EFDP. This means that if β/γ is high, the decoupling performance will be better. The solution of the optimization problem (4.6),solve by (4.8), returns the least order robust residual generator filter $\hat{Q}(z)$. Specifically, the resulting residual generator filter is composed as $\tilde{\boldsymbol{Q}}(z) = [\hat{Q}(z)\hat{M}(z), -\hat{Q}(z)\hat{N}(z)]$.

Remark 4.1

Due to the low bias of the kernel-based system identification the model uncertainty corresponds to the noise-induced uncertainty, therefore the generated robust residual generator works also with the true plant.

Algorithm 5: Synthesis of $\tilde{\boldsymbol{Q}}(z)$ with automatic design of $\hat{W}_T(z)$

Input: $\hat{\beta}, \hat{\boldsymbol{\Sigma}}, \mathcal{W}, n_p, \gamma$

1 Compute $\hat{W}_T(z)$ by using the Algorithm 1

2 Synthesis of the least-order robust residual generator filter $\tilde{\boldsymbol{Q}}(z)$ by optimizing $\hat{Q}(z) = \max_{Q(z)} \frac{\left\|Q(z)\hat{M}(z)G_f(z)\right\|_\infty}{\left\|Q(z)\hat{M}(z)\left[\hat{G}_0(z)\hat{W}_T(z) \;\; 1\right]\right\|_\infty}$. This is done through the algorithm proposed in [27, Chapter 5.3], by setting γ. This algorithm returns also the performance indicator β/γ.

Output: $\tilde{\boldsymbol{Q}}(z)$,$\beta/\gamma$

Since the sources of the fictitious noise in our case is known, we propose also an automatic threshold selection method, in line with the advice of [27, Chapter 5]:

$$\tau = \gamma \cdot \max\left\{\|u(t)\|_\infty, \|e(t)\|_\infty\right\} . \tag{4.9}$$

This is useful since the optimization of (4.6) reduces the impact of the fictitious noise, but does not avoid it entirely. So, the residual signal can be different from zero when the system is healthy. As described in Section 1.3, there is a necessity to implement a threshold that further reduces false alarms. Our choice is devoted to detecting the worst case of the residual signal in healthy conditions. Therefore, by complying with the Assumption 2.1 and by assuming that the bound value is known, we can define de threshold by multiplying γ with respect to maximum between of the infinity norm of input of of the noise. To do so we need to assume that the noise bound is known. Finally, if the residual signal overtakes the selected threshold τ, the system is considered faulty.

Remark 4.2

Note that the knowledge of the fictitious noise is not employed during the post filter estimation, but in the threshold selection, therefore we exploit all the information gathered during the system identification procedure.

Algorithm 6: Fault detection algorithm

Input: $\tilde{\boldsymbol{Q}}(z), \hat{G}_0(z), \gamma, u(t), y(t)$

1 Compute the threshold τ using (4.9)

2 Compute the residual signal $r(t) = \hat{Q}(z)\hat{M}(z)y(t) - \hat{Q}(z)\hat{N}(z)u(t)$

3 Compare τ with respect to $|r(t)|$:

- If $|r(t)| > \tau$ then $f(t) = 1$;
- If $|r(t)| < \tau$ then $f(t) = 0$.

Output: $f(t)$

4.3 Experimental results

The proposed method is tested on a benchmark plant, proposed in [28, Chapter 3]:

$$G_T(s) = \frac{47619}{s^3 + 234.0136s^2 + 6857.1s + 5442.2}. \tag{4.10}$$

The identification is done with a white noise input signal with mean 0, variance 1, sample time $T_s = 0.1s$ and $n = 5000$ samples. The output signal is corrupted with additive bounded noise $e(t)$. The signal to noise ratio of the output signal is $SNR = \text{var}\left[y_T(t)\right]/\text{var}\left[e(t)\right] = 9$. The first 1000 samples are not considered to avoid the transitory effect. The FIR order for the kernel-based identification is $m = 100$. The automatic design of $\hat{W}_T(z)$ is done by choosing $\zeta = 0.05$ and $\varepsilon = 0.01$, the number of sampled system is set as $n_p = 18445$ by following (2.24). Instead, the number of sampled frequencies is set to $n_m = 600$, logarithmically spaced in the range $\left[10^{-3}, \pi f_s\right]$ rad/s.

Figure 56 shows on the left image the identification results and on the right image the uncertainty weight resulting from the automatic design.

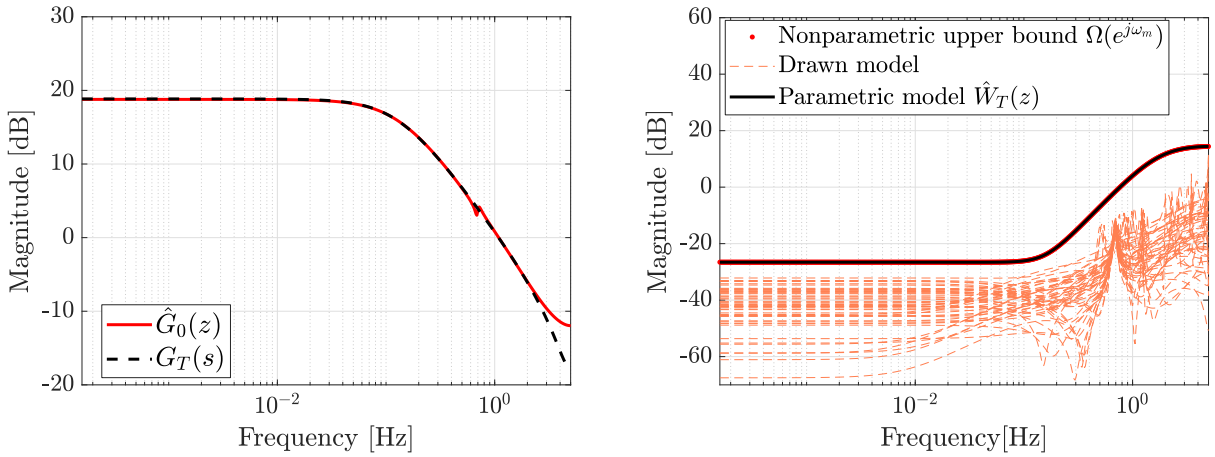


Figure 56: (Left) Identification results, where: the dashed line is the true system instead the straight line represents the identified discrete model by using the kernel based identification (Right) Uncertainty weight $W_T(z)$ estimation, where: the continuous line is the magnitude of the parametric model $\hat{W}_T(z)$, the dots line is the nonparametric estimate $\Omega(e^{j\omega_m})$ and the dashed lines represent The gain of the frequency response of $\frac{G_p(z)}{\hat{G}_0(z)} - 1$, where the systems $G_p(z)$ are drawn from the sampling distribution of the parameters estimates. The dashed lines are a subset of the n_p systems for graphical purposes.

Figure 57 depicts the resulting filter $\hat{Q}(z)$ generated from the Algorithm 5 with $\gamma = 0.1$. The filter, as we expect, has the inverse frequency response of the uncertainty weight $\hat{W}_T(z)$. This means that $\hat{Q}(z)$ *filters out the components of the residual signal that belong to the most relevant frequencies in the frequency response of the uncertainty weight.*

The threshold is chosen as the proposed automatic method (4.9). Since all required information to produce the threshold should be a priori known, the computation is very trivial.

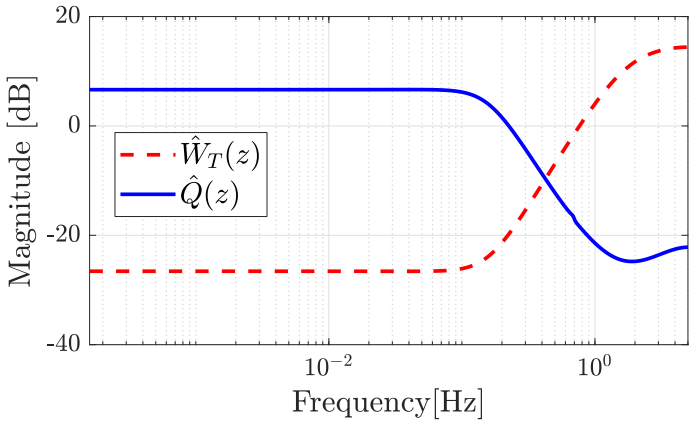


Figure 57: Comparison between $\hat{W}_T(z)$ and the resulting $\hat{Q}(z)$.

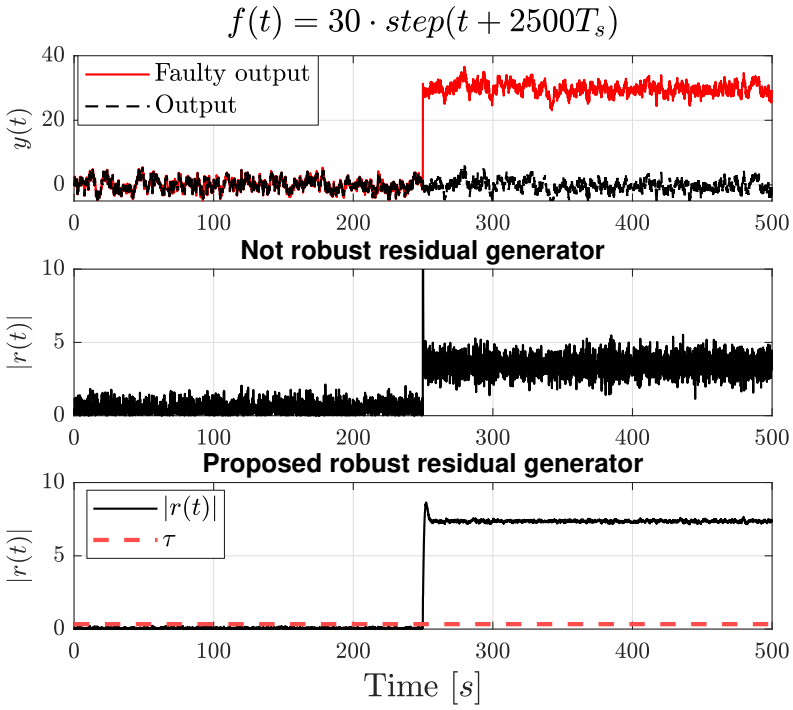
The resulting residual generator is compared with the non-robust stable coprime factorization residual generator, modeled as: $r(t) = \hat{M}(z)u(t) - \hat{N}(z)y(t)$. To proof of the robustness of the proposed methodology, we have applied a sensor fault modeled as an additive step fault signal that arises at time 250 s with two different amplitudes: 30 and 3. The images of Figure 58 depict:

- On the top plot the output healthy signal (black dashed line) compared with the faulty output signal (red line);
- On the middle plot the absolute value of the residual generated by the non-robust residual generator;
- On the bottom the absolute value generated by the robust residual generator, resulting by our proposed method, compared to the threshold, selected as in (4.9).

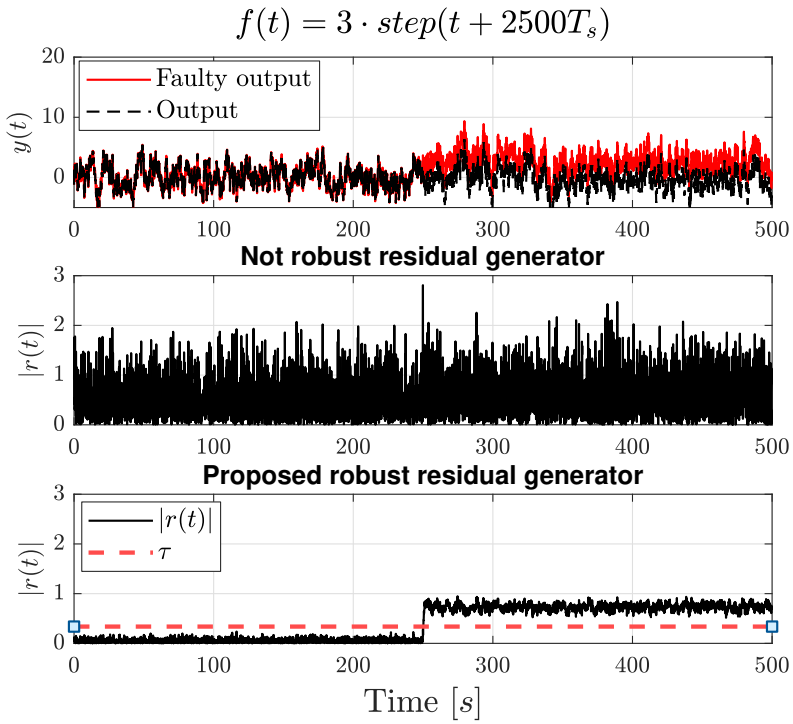
The results show the robustness of our proposed methodology with respect to the non-robust method. Specifically, both robust and not robust residual signals can detect $f(t) = 30 \cdot \text{step}(t + 2500T_s)$. However, the robust residual generator decouples better the noise w.r.t $|r(t)|$ since the sector of signal $|r(t_h)|$ (with $t_h < 250s$) differs little with respect to the sector $\left|r(t_f)\right|$ (with $t_f > 250s$) computed with the not robust residual generator. Instead, the second simulation shows that the not robust residual generator cannot detect the fault, while the robust proposed technique detects correctly the fault.

4.4 Conclusions

The proposed method aims to design a data-driven robust residual generator using the system identification results. The algorithm exploits a kernel-based system identification to model the uncertainty information as an unstructured multiplicative uncertainty. Furthermore, the uncertainty information is used to design the least order filter that decouples the uncertainty from the residual signal. The filter



(a) Analisys of residual signal with fault step amplitude as 30



(b) Analisys of residual signal with fault step amplitude as 3

Figure 58: Comparison between robust and not robust stable coprime factorization varying the amplitude of the fault signal. The top image depicts healthy (black line) and faulty (red line) sensed output signals. The bottom image shows the module of the residual signal computed with the not robust residual generator. The bottom image illustrates the module of the residual signal computed with the proposed robust residual generator (black line) and the proposed threshold (red dashed line).

design is done by solving the Approximate Fault Detection Problem (AFDP), since the uncertainty model is translated as an additive fictitious noise. Furthermore, we propose also an automatic threshold design. The effectiveness of the proposed method is shown on a benchmark dynamic system. The results prove that the data-driven robust residual generation detects a sensor fault with the both proposed fault signals, modeled as an additive step signal. Instead, with low amplitude fault, the not robust technique fails. Furthermore, the proposed threshold detects correctly the faults.

Part II

Applicative contributions

Chapter 5. Robust control design for a reconfigurable industrial oven

This chapter describes a real application that exploits the methodology proposed in Chapter 3. The rationale is to design a robust controller through the *data-driven S/T mixed-sensitivity loop-shaping for multi-model systems*. The plant under analysis is an oven for heat shrinking. This system is complex and configurable. The aim is to design a controller that guarantees robust stability for all configurations and nominal performance of some requirements automatically designed. The results are evaluated by comparing the step response of the closed-loop system obtained with the proposed method and with a manual choice of the uncertainty weight.

5.1 Motivation

The industrial world is a competitive field. This industrial "race" leads to building the cutting-edge machinery, which often translates into the development of customizable machines. This phenomenon is not trivial for the control community since these configurations require a complex controller to guarantee that the plant works with all configurations.

Often, these configurable systems are translated into multi-model systems by the controls engineers. These multi-model systems can be controlled in different ways, such as gain scheduling control, Model Predictive Control (MPC) of even by robust control. As already said in Chapter 3, the former does not guarantee any stability or performance during a rapid change of the variables [127]. Instead, MPC controls the plant by solving an online optimization problem in real-time [132]. Always, in the industrial context, the hardware is not sufficient to solve real-time problems. The robust control literature helps us to avoid these problems by guaranteeing stability and some performance requirements. We develop a controller which stabilizes the plant under all configurations by solving an offline optimization problem.

In this work, we apply the methodology presented in Chapter 3 to an industrial oven for shrinking plastique films.

5.2 Application context

The industrial oven context is considered part of the thermal engineering literature. The thermal systems taxonomy is mainly: Heat, ventilation and air conditioning system (HVAC), industrial furnace

and industrial oven. HVAC controls the temperature, humidity, and purity of the air in an enclosed space. The authors of [133] and [134] have developed a mixed-sensitivity loop-shaping for an HVAC. In both works, the weight functions are designed manually. Specifically, the first work models the HVAC with a MIMO dynamic system. These systems are complex to control with mixed-sensitivity loop-shaping because it leads to a high number of manually tuned weight functions. The second work exploits a robust control design for an HVAC system applied to an automotive air conditioning system. The same robust control loop-shaping approach is developed for a thermoforming oven field [135]. Notwithstanding the cons of the gain scheduling and MPC, in [136] and [137], the authors have developed an MPC controller to regulate the temperature for a different types of furnace. In [138], a Gain scheduling control is applied to control an HVAC.
Before evaluating the proposed method's performance, in the next subsection, we describe the properties of the oven under analysis.

5.3 Oven for heat shrinking

The plant under analysis is an industrial oven. This item is developed for heat shrinking of plastique film. The plastique film cover different typologies of products, for instance: plastique bottles, glass bottles or tin cans, etc.

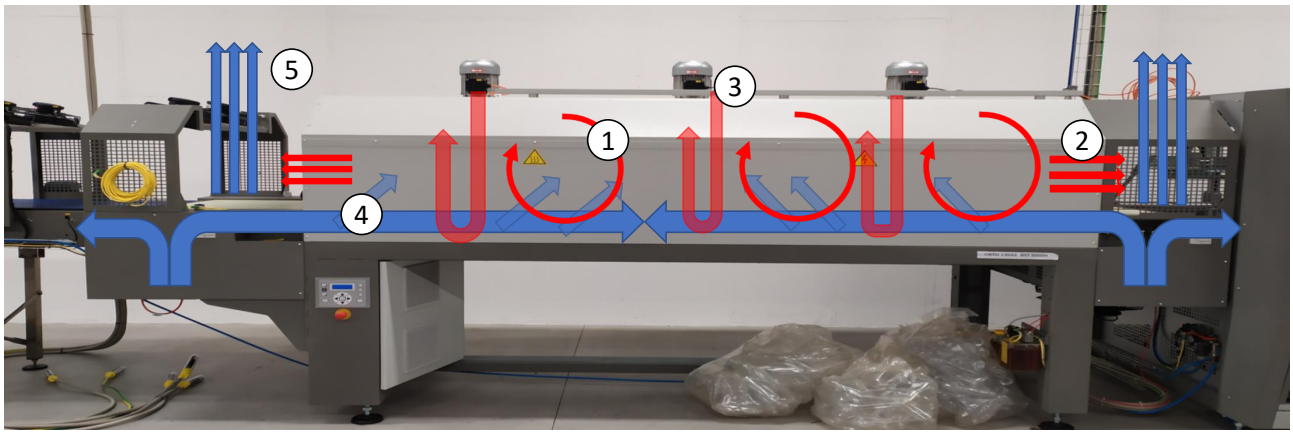


Figure 59: The industrial oven for heat shrinking with the highlighted hot and cold air flows.

The oven is composed of:

- *Main hole*: which is the biggest part of the system. It covers all the length of the oven. The products, wrapped with the film, pass through it and thanks to the high temperature the plastique film shrinks;
- *Three internal fans*: which are positioned on the top and equidistantly spaced. Two of these are powered by a fixed signal, instead the third is controllable;

- *Conveyor belt*: which feeds the oven with the products wrapped by the plastique film. Mainly, it is inside the main hole[1]. The height position is about at 1/4 (from the floor of the main hole) of the total main hole height;

- *Four external fans*: two out of four cool the conveyor belt[2], instead, the other two are positioned over the oven entrance and exit, which cool the products;

- The oven temperature is measured by a *single thermocouple* positioned at the top center of the main hole;

- The main hole is flanked by *two cavities*. They are two parallelepipeds which have the same height and length as the main hole. These cavities are connected to the main hole with some lateral slots positioned at 3/4 (from the floor of the main hole) of the total main hole height. They share the area between the floor of the main hole and the conveyor belt. The slots of the cavities can be totally closed or totally open;

- *Some air deflectors*: which are positioned inside the area shared between the cavities and the main hole shared (under the conveyor belt and over the floor oven). These direct the airflow without changing the amount of flow;

- *Six thermoresistances*: which heat the oven. They are powered by the same Pulse-Width Modulation (PWM) signal. Their positions are into the concavities, three for each. Their lengths are long as the main hole size;

- *Input and output PVC strip curtains*: which allow isolating the hot air inside the oven with respect to the colder ambient air. If the industrial oven is in manufacturing, then the PVC strip curtains remain open almost all the time (since the wrapped products enter and exit), otherwise, in the "out of the production" condition, the curtains are closed. They are positioned on the oven exit and oven entrance.

Figure 59 illustrates the industrial oven. The hot air flows are represented in red arrows and the cold air flows in blue arrows. The following list explains the meaning of each arrow. Specifically, each list item number corresponds to the number depicted in Figure 59. The main air flows are:

[1] A small part of the belt is outside the oven, specifically at the oven entrance and exit.

[2] They are positioned under the external part of the conveyor belt.

1. The air inside the oven heats up in the concavities and it flows into the main hole through the lower shared area. If the lateral slots are open the air recycling is higher since the internal fans feed the concavities with the main hole air through the slots;

2. The hot air goes out of the oven through the input and output PVC strip curtains. Obviously, if the oven is in production, then the quantity of this flow is higher than the "out of the production" condition because the position of the curtains changes;

3. The three internal fans move down the hot air;

4. The movement of the conveyor belt brings the outside air into the oven;

5. The external fans move up the clod air next to the PVC strip curtains.

A robust controller is needed to control all the oven configurations. A non-robust controller can be ineffective, because the huge configuration variability can lead to undesired oven behaviors.
The main assumptions of the proposed methodology are valid since the boundary configurations are known and the input and output dataset can be acquired in both conditions. Table 5 reports the best and worst configurations.

	Worst	***Best***
Conveyor belt	ON	ON
Conveyor belt fans	ON	OFF
Power frequency internal fans	$50Hz$	$50Hz$
Lateral slots	Open	Closed
Input and output PVC strip curtains	Open	Closed

Table 5: Boundary conditions features.

Remark 5.1

The air deflector configurations are not considered in the analysis of the boundary conditions, because their impact is much less than the effect of the other features since they direct the flows without changing the quantity.

5.4 Experimental setup

The input and output signals are acquired through a simulator that simulates both conditions, where the input is the duty cycle of the PWM of the six thermoresistances and the output is the sensed temperature. The input signal can vary between 0% to 100%. The simulator is developed to represents all possible configurations and it faithfully represents the real plant.
According to the experimental design, described in Section 3.2, Figure 60 shows the measured output data, sampled with sampling time $T_s = 60\,s$. Specifically:

- We simulated $n = 4800$ samples for each configuration;
- The first and the last 400 samples for both configurations are acquired with null input;
- The first and last 1000 data of the general dataset are discarded to remove initial and final transitory effects.

Note that the first sector corresponds the output of the industrial oven in the worst condition, instead, the second represents the output of the best condition. Since the most used duty cycle values are between the 30% and 90%, the chosen input is an amplitude bounded noise with mean 60 and standard deviation 30 for both segments.

Remark 5.2

It is important to highlight that the output signal does not contain some discontinuities. This is fundamental to identifying the mean dynamic system with the kernel-based identification.

The FIR order is set as $m = 100$.
The chosen controller structure is :

$$K(z, \boldsymbol{\rho}) = k_p + k_i \frac{T_s}{z-1} + k_d \frac{1}{N_d + \frac{T_s}{z-1}}\,, \tag{5.1}$$

where $\boldsymbol{\rho} = \left[k_p, k_i, k_d, N_d\right]^\top \in \mathbb{R}^{4\times 1}$. The multi-objective problem (2.35) is solved by setting $\gamma = 0.15, o_{\min} = 0.01, \ell_{\max} = 10000\,\text{s}$.

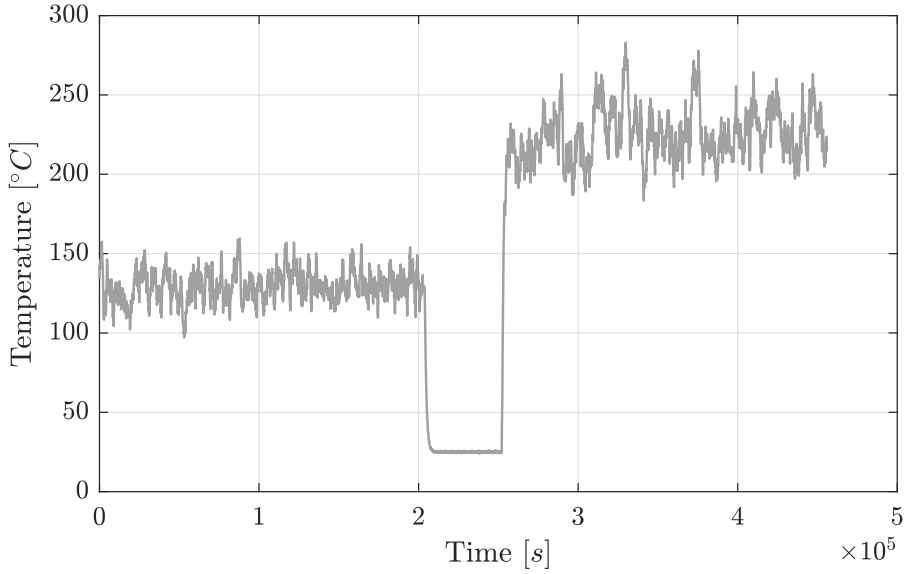


Figure 60: The measured output signal of the general experiment.

5.5 Experimental results

The design of $\hat{W}_S(z)$ is designed for robust stability and nominal performance, however the entire procedure is also applicable for the robust performance aim.

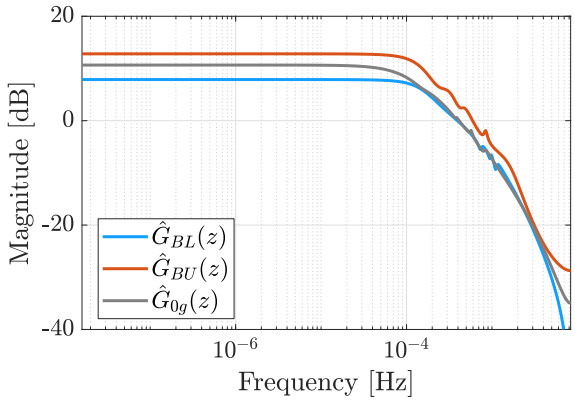
Figure 61a depicts the results of the identification procedure: the best condition is highlighted in orange, the worst condition in light blue and the average system in grey.

Figure 61b depicts the overall uncertainty $\hat{W}_T(z)$ estimated by the Algorithm 4.

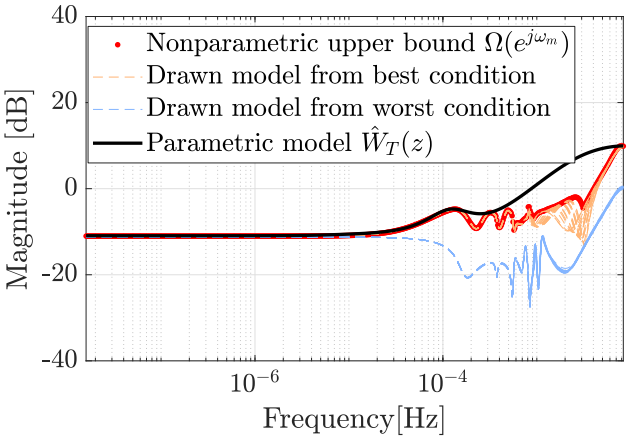
The controller parameters are estimated by exploiting the Algorithm 2. The images of Figure 62 show the graphical representation of robust stability and nominal performance constraints. The resulting controller guarantees the nominal performance and robust stability since the inverse of the weight functions plus γ stay-over, respectively, the complementary sensitivity function and the sensitivity function computed using the average model $\hat{G}_{0g}(z)$.

The performance evaluation are done by random sampling the open-loop systems from the Gaussian distributions $\mathcal{N}(\hat{\beta}_{BU}, \hat{\Sigma}_{BU})$ and $\mathcal{N}(\hat{\beta}_{BL}, \hat{\Sigma}_{BL})$, specifically $n_v = 200$ for both. The step responses of the resulting $2 \cdot n_v$ closed loop systems are depicted in Figure 63. Observe that the performance constraints (black dashed lines) are valid only for the closed-loop of the system $\hat{G}_{0g}(z)$ denoted as $\hat{T}_{0g}(z)$ (grey dashed line) and, as we expect, both constraints are respected.

The performances of the resulting controller obtained with the data-driven mixed-sensitivity loop-shaping are compared to the performances of a controller synthesized with a manual choice of $W_T(z)$. Usually, the manual weight function is defined by three parameters: critical frequency ω_c, high-frequency gain h_g and low-frequency gain l_g. In this application example, we consider that:



(a) Frequency response of the boundary conditions $\hat{G}_{BU}(z)$ (in orange), $\hat{G}_{BL}(z)$ (in light blue) and the average system $\hat{G}_{0g}(z)$ (in grey).



(b) Overall uncertainty weight $W_T(z)$ estimation. (Continuous line) Magnitude of the parametric model $\hat{W}_T(z)$. (Dots) Nonparametric estimate $\Omega(e^{j\omega_m})$. (Dashed line) The gain of the frequency response of $\frac{G_p(z)}{\hat{G}_{0g}(z)} - 1$, where the systems $G_p(z)$ are drawn from the sampling distribution of the parameters estimates, in particular, the orange lines depict the frequency responses with $G_p(z)$ drawn from the best condition distribution, instead, the light blue lines illustrate the frequency responses with $G_p(z)$ drawn from the worst condition distribution. The dashed lines are a subset of the n_p systems for graphical purposes.

Figure 61: Kernel-based identification and estimation of $\hat{W}_T(z)$ results.

- $W_S(z)$ is equal to the $\hat{W}_S(z)$, i.e. the performance weight resulting from the automatic weight function design;
- The critical frequency of the manual uncertainty weight is assumed equal to the critical frequency of $\hat{W}_T(z)$.

In this way, we greatly simplify the real problem, since we reduce the tuning parameters from 6 (three for each weight) to 2. Furthermore, we reproduce the trial and error design for the last two tunable parameters by testing the S/T mixed-sensitivity loop-shaping on all pairs of a test grid, structured as h_g chosen from the vector $[10, 20, 30, ..., 100]$ and l_g chosen from the vector $[-50, -45, -40, ..., -5]$. Figures 64 represents the cost function $J(\hat{\rho}, \hat{G}_{0g}(z))$ resulting from the manual design for all combinations of h_g and l_g. The red curve represents $1+\gamma$: the constraint of the data-driven mixed-sensitivity loop-shaping. As depicted, only a small subset of the considered manual choices of $W_T(z)$ agrees with the constraint.

The images of Figure 65 represent respectively the curves of the indices: IAE, settling time and overshoot. In all images, the green curve illustrates the performance of the proposed design, instead

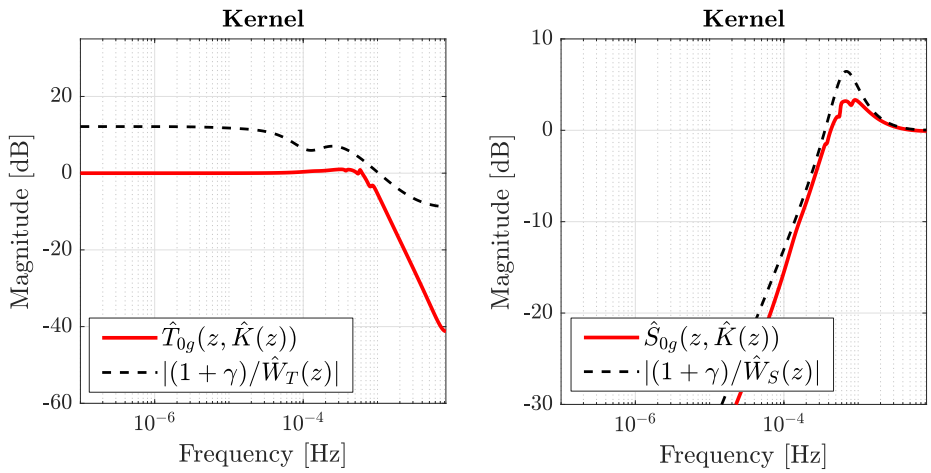


Figure 62: (Left) (Continuous line) Closed-loop complementary sensitivity function $\hat{T}_{0g}(z)$ using the estimated controller $\hat{K}(z)$ and the average system. (Dashed line) $1+\gamma/\hat{W}_T(z)$, that should lie above $\hat{T}_{0g}(z)$. (Right) (Continuous line) Uncertain closed-loop sensitivity function $\hat{S}_{0g}(z)$, computed with the estimated controller $\hat{K}(z)$ and the average system. (Dashed line) $1+\gamma/\hat{W}_S(z)$, that should lie above the uncertain closed-loop sensitivity function.

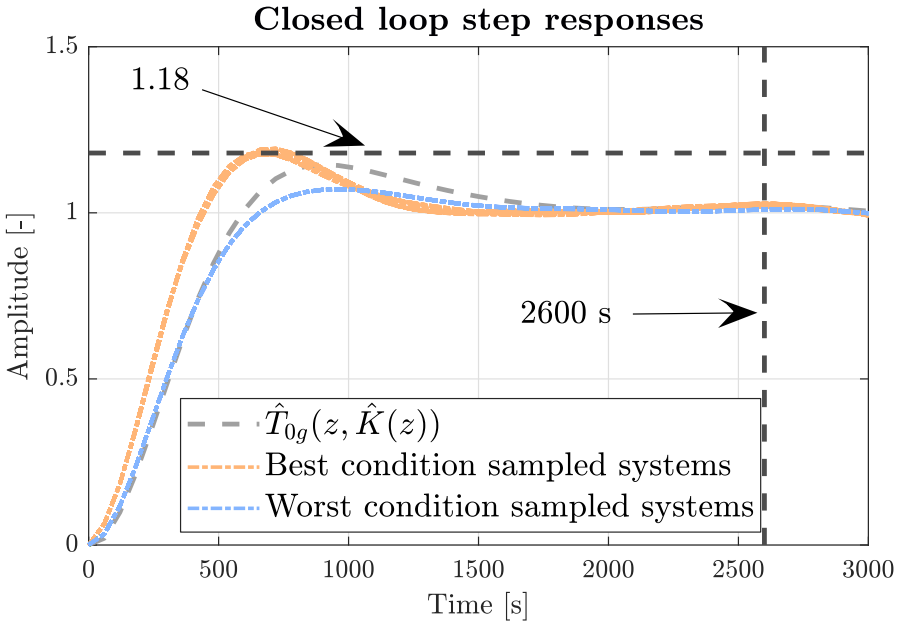


Figure 63: Step responses of the $2 \cdot n_v$ (n_v for each boundary condition) sampled systems coupled with the resulting controller $\hat{K}(z)$, compared to the step response of the average dynamic system denoted as $\hat{T}_{0g}(z)$. The black dashed lines represent the performance constraints for $\hat{G}_{0g}(z)$.

the colored curve depicts the performance obtained with the controller designed with the manual choice of $W_T(z)$. Both curves illustrate the average performance indices obtained with the sampled $2 \cdot n_v$ closed-loop systems. Specifically, the colored curves in:

- The left images represent the curve obtained with the manual design for all couples of h_g and l_g,

- The right images depict a sections of the colored curves represented in the left images. The couples h_g and l_g compose the right curves, if the corresponding cost functions $J(\hat{\rho}, \hat{G}_{0g}(z))$ computed with the manual $W_T(z)$ (designed with h_g and l_g) are less than $1+\gamma$. Since the couples that agree with the cost function constraint are six and all of these share the same h_g value, the right curves are a represented as lines.

For all performance indices, the proposed method behave better than the manual design considering that $W_S(z)$ and the critical frequency of $W_T(z)$ are chosen by employing the results of our automatic weights design.

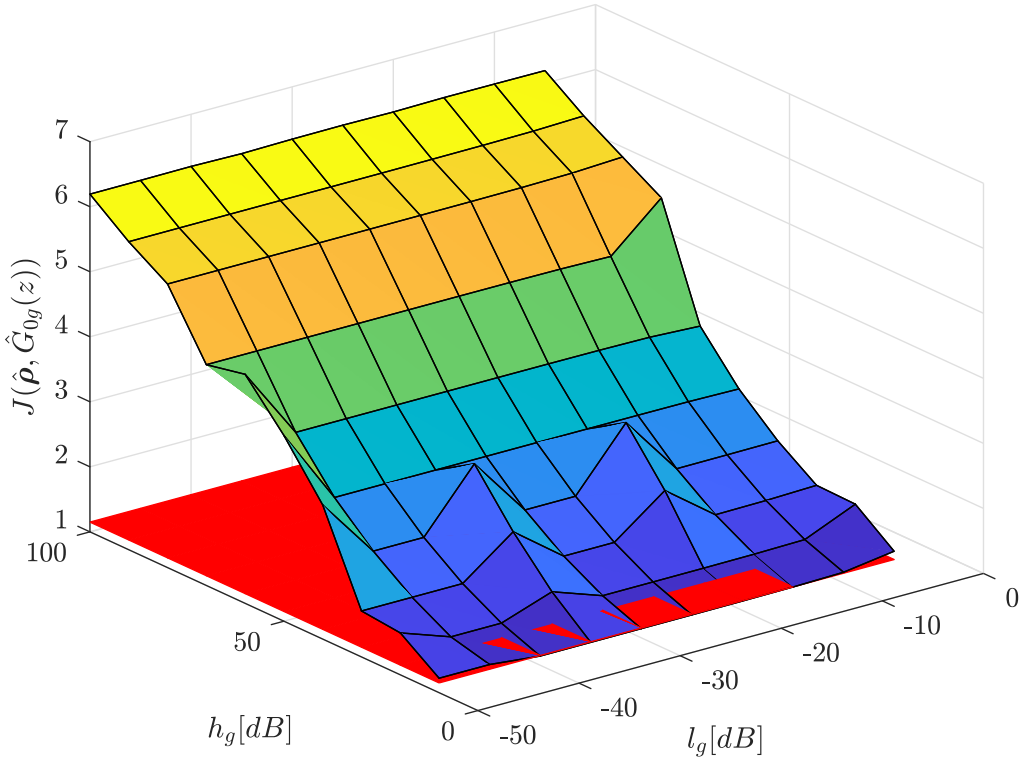
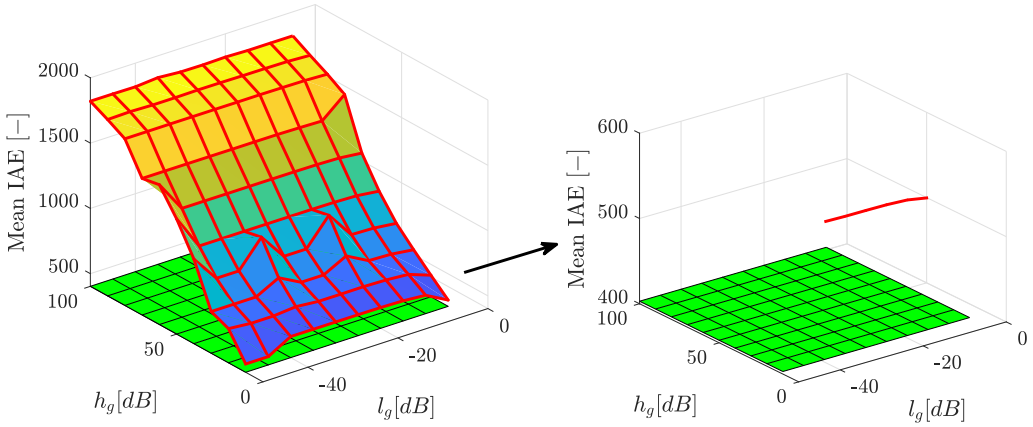


Figure 64: Graphical representation of $J(\hat{\rho}, \hat{G}_{0g}(z))$ by varying h_g and l_g. The red curve depicts the relaxed constraint value $1+\gamma$.

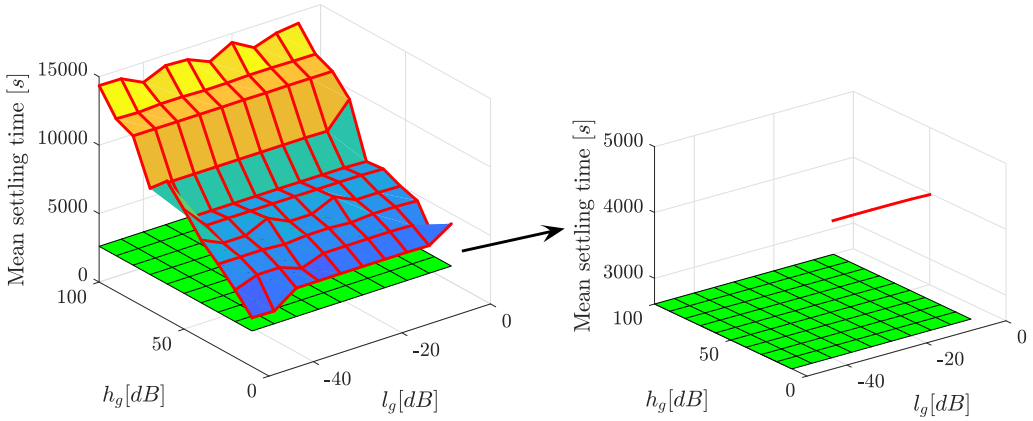
5.6 Conclusions

The automatic weight functions design via kernel-based system identification applied to the mixed-sensitivity loop-shaping, proposed in Chapter 2, is adapted to handle multi-model systems. To do so, a general distribution, which embeds all parameters uncertainties and model identification uncertainties, is developed. A key role is played by the experimental design, as described in Chapter 3.

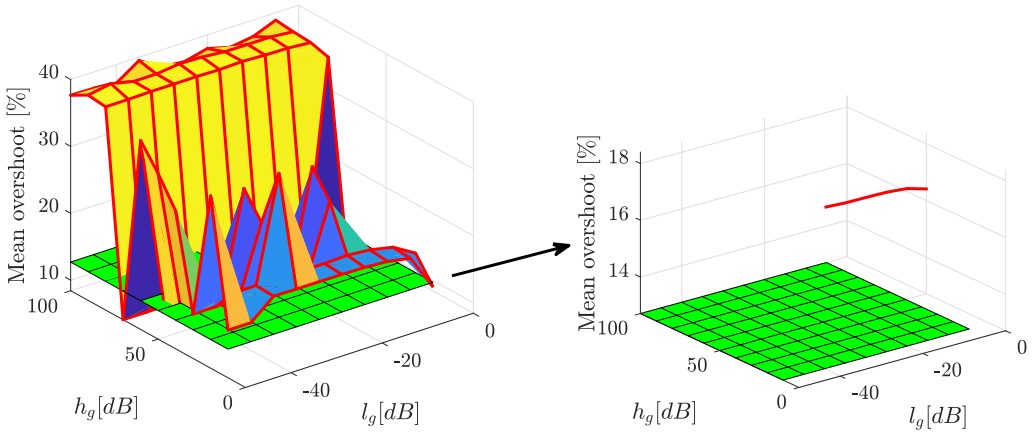
The proposed methodology is applied to an industrial oven which shrinks plastique films. This application is complex since the oven has many configurations to operate with different types of films and wrapped products. The result demonstrates that the data-driven weight functions design:



(a) Graphical representation of the average of IAE index.



(b) Graphical representation of the average of settling time index.



(c) Graphical representation of the average of overshoot curve index.

Figure 65: All images depict the average values of the performance indices computed on the closed-loop system obtained by $\hat{K}(z)$ and the sampled $2 \cdot n_v$ dynamic systems. The colored curves represent the results obtained with the manual design, instead the green curve the results obtained with the proposed design. Specifically: (Left) Illustrates the performance indices obtained with all couples of h_g and l_g, (Right) Report a portion of the curves depicted in left images. This section is composed of the couples that meet with the constraint $J(\hat{\rho}, \hat{G}_{0g}(z)) < 1 + \gamma$ (see Figure 64).

- Avoids the trial and error procedure;

- Outperforms all combinations of manual design of $W_T(z)$, despite, in this case, the degrees of freedom of the problem of weight functions selection are reduced to simplify the comparison.

As already mentioned, the overall uncertain system can be employed to control aims, thanks to the low bias property of kernel-based identification. On the other hand, if we use a parametric identification with a wrong selected model family, the resulting overall distribution is biased and thus the uncertain multi-model system does not represent the real plant. This phenomenon leads to a useless controller that works with the model, but not with the real plant.

Chapter 6. Experimental fault detection of input gripping pliers in bottling plants

This chapter presents a signal-based fault detection scheme for input gripping pliers of the blow molding machine in plastic bottling plants, using accelerometers data. The focus of the diagnosis is on the bearings that support the pliers movements on their mechanical cam. The rationale of the algorithm lies in interpreting the pliers' bearings as the balls in a traditional rolling bearing. Then, strategies inspired by bearing diagnosis are employed and adapted to the specific case of this work. The developed algorithm is validated with experimental tests, following a fault injection step, directly on the real blow molding machine

6.1 Introduction

A bottling plant is a complex system that manages the entire production cycle for the sale of beverages. These plants can be large, distributed, and have a complex layout, see [139]. This work focuses on the production process of PolyEthylene Terephthalate (PET) plastic bottles. In this case, the input material is usually a rigid plastic preform. The process is composed by a series of sequential operations, covering e.g.: (i) preform feeding; (ii) heating; (iv) blowing; (iv) bottles filling and capping; (v) labeling; (vi) transportation along the production line; (vii) packaging and (viii) palletizing, see Figure 66. Each one of these processing steps is performed by a specialized machine, with their respective mechanical and electrical components. These machines are most of the times independently controlled, and their synchronization is performed during the first setup of the plant by an expert operator.

The entire process stops if a machine or a component fails in one the main production steps, often causing large wastes of wrecked bottles, liquid and caps. Furthermore, if e.g., bottles get stuck in the blow molding machine, the operator has to manually extract them from the molds or from the pliers, wasting a lot of useful production time. As an example, the filling and capping machine stops if there is a lack of bottles from the input pliers, or if there is a tailback of filled bottles at output pliers. The works of [140, 141] showed how the unscheduled downtime in bottling plants can vary between 10% and 60% of the total production time. Thus, a fault diagnosis system is highly envisaged to reduce (and prevent) the time wastes in this kind of production lines.

The concept of fault diagnosis refers to the general usage of specific techniques to assess the status of a system with respect to its possible faults (for further information see Section 1.3).

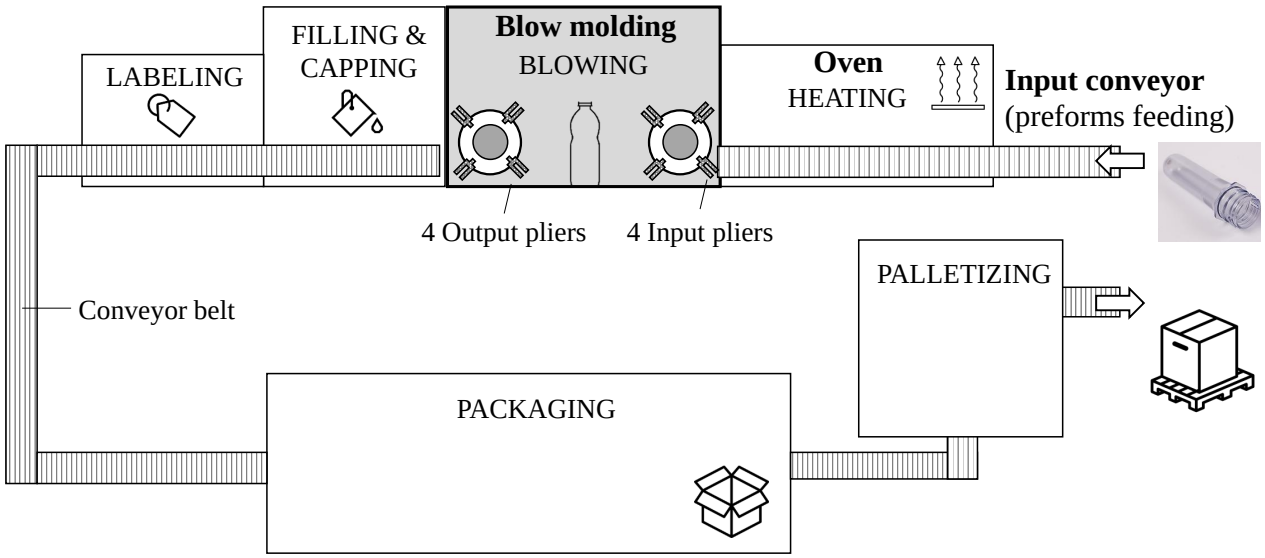


Figure 66: Example of a bottling plant with main machines. The blow molding machine is highlighted as the main machine considered in this work.

Fault diagnosis approaches in bottling plants have been relatively little addressed in the literature. Models of chained production lines can be found in the queuing theory of [142]. In [143], the authors focused on the diagnosis of the filling machine, by employing a type of model-based fault diagnosis known as consistency-based approach. Here, the whole plant has been modeled with a set of components, and anomalies in the bottle transport flows are detected. An alternative modeling of the plant is proposed in [144], where Petri-nets are employed for the detection of bottle overflows, improver filling valve operation and Infra-Red (IR) sensors monitoring. A decision-tree expert system is employed in [145], where transition time data are used for detecting faults in a brewery plant.

This work presents a *signal-based FD scheme for gripping pliers in bottling plants*. The pliers are present in the blow molding machine. The aim of the input and output pliers is to carry the preforms and the bottles, respectively, from the input conveyor belt into the blow molding machine and from the blow molding machine to further processing. The presented literature investigated the modeling the entire production process, mainly with an higher-level outlook and a focus on the flow of the bottles material. Instead, we focus our attention on a *specific component and on its working behavior*. The aim of this work is to *detect the degradation of the posterior bearings* used for the movement of pliers that carry the heated preforms from the oven into the molds of the blow molding machine. The devised FD algorithm interprets the pliers' bearing like the ball element in a traditional rolling bearing. Thanks to this interpretation, signal processing techniques can be employed on accelerometers data to extract diagnostic information. The application of the proposed diagnostic algorithm on experimental data shows the goodness of the method.

6.2 Experimental setup

This section shows: the machinery description, the fault injection design, the testing procedure and the characteristics of the employed sensors.

6.2.1 System description

The bottling plant under consideration is composed by the following components, see also Figure 67:

1. *Input conveyor*, that feeds the raw plastic preforms into the plant;
2. *Oven*, used to heat the plastic preforms;
3. A set of *input gripping pliers*, that take the overheat preforms and bring them into the blow molding stage;
4. *Blow molding machine*, that blows the preforms, now fixed in a mold, into the final bottle format;
5. A set of *output gripping pliers*, that take the blown bottles from the blow molding machine and bring them into the filling stage;
6. *Filling and capping machine*, that fills the blown bottles with a liquid and put the caps;
7. *Conveyor belt*, that transports the filled and capped bottles;
8. *Packing and palletizing machines*, that create bottles packs and arrange them for loading and transportation.

In this work, we consider only the blow molding machine, see Figure 67. This machine is composed by three important components:

- *Carousel*, a rotating component that supports one or more stations;
- *Stations*, i.e. the main component for blowing the preforms. Each station is made up of:
  - *Mold*, that defines the form of the blown bottle;
  - *Rod*, which is used to stretch the heated preform prior to its blowing;
  - the blowing system;
- The output and input *gripping pliers* (the main focus of this study).

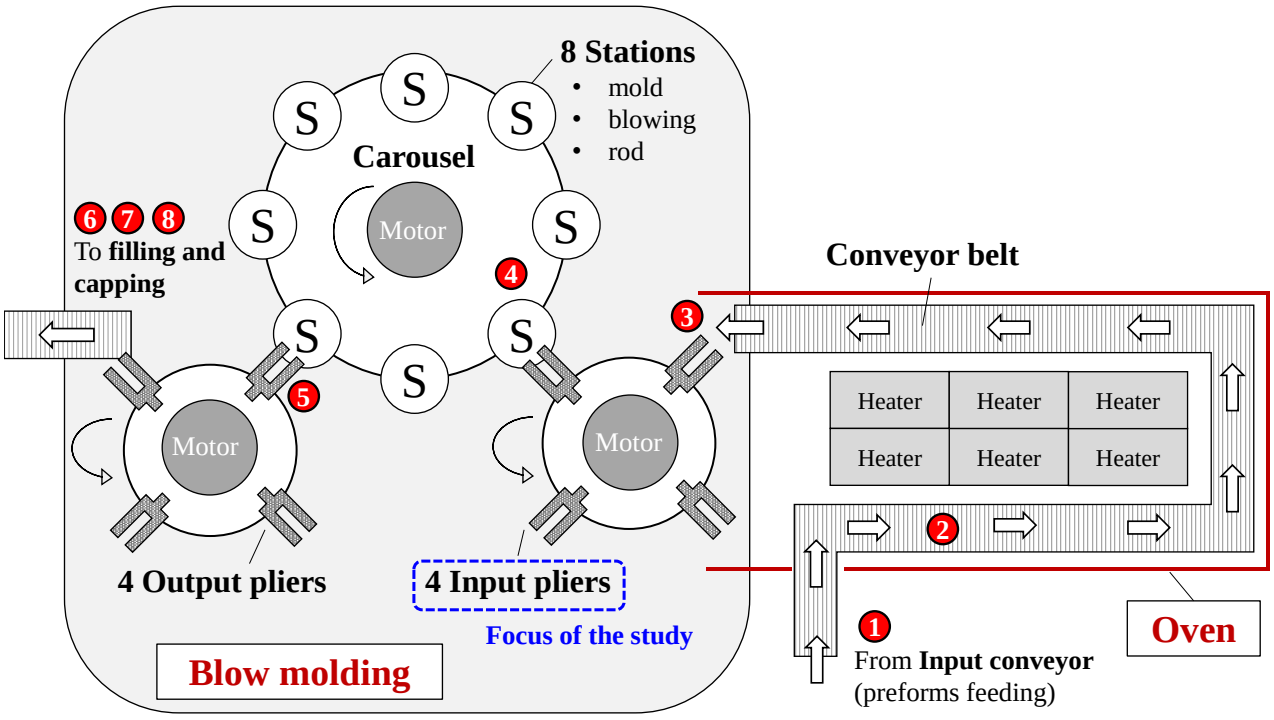


Figure 67: Schematic of the oven and blow molding machines for a plastic bottling plant. The steps of the production process, from (1) to (8), are highlighted with respect to the components responsible for each step.

The blow molding machine under consideration has 8 stations and 4 input/output pliers. An example of a gripping plier is depicted in Figure 68-(left). The pliers lie on a mechanical cam via two sets of bearings: two posterior and a frontal one. The arms of the plier, responsible for the gripping, are connected to the frontal bearing. The two sets of bearings (posterior and frontal) are connected by two springs. The cam is made in such a way that, when the springs stretch (i.e. when the distance between the two set of bearings is large), the plier arms open due to the retracting of the springs. When this happens, the preform can enter between the arms. Then, the cam mechanics release the tension on the springs and the arms close, this time holding the preform. This behaviour is schematized in Figure 69.

When the springs are stretched and the pliers arms open, the posterior and frontal bearings are tightly attached to the cam structure. In all other cases, the bearings may not be always in contact with the cam. Thus, we expect to detect a damage on the posterior bearings in the first case, i.e. when the plier arms are fully open. From Figure 69 we observe that the pliers arms are open when the plier has to take a preform from the oven, or when it has to release a preform inside the mold of a station.

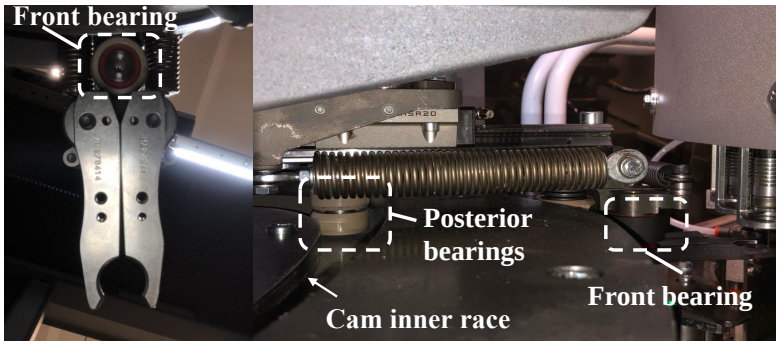


Figure 68: (Left) Upper-view of an input gripping plier with front bearing detail. (Right) Upside down side-view with posterior bearings, springs and cam detail.

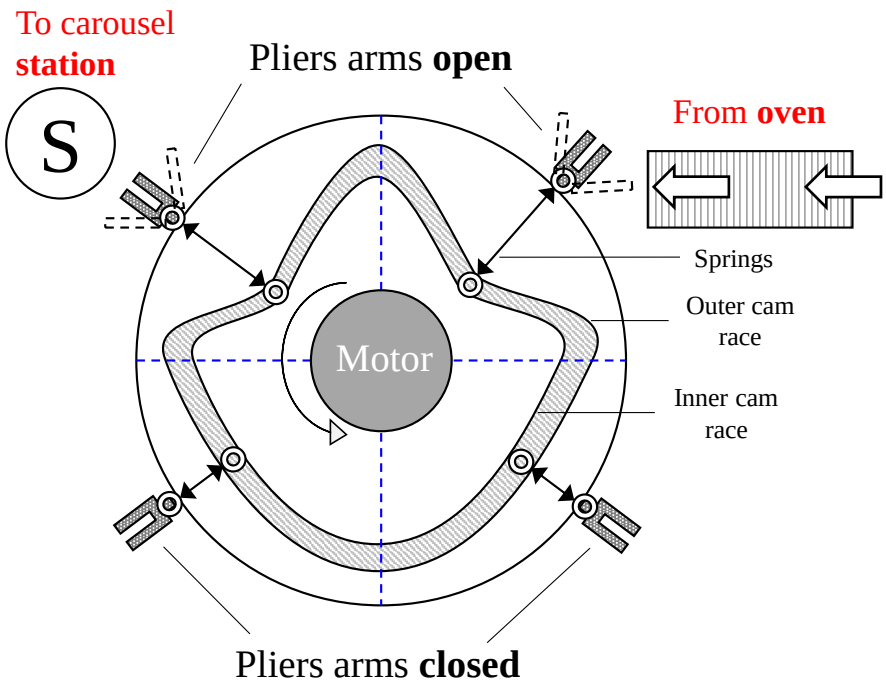


Figure 69: Opening and closing mechanism of the pliers arms as function of the plier position on the cam.

6.2.2 Fault injection and testing procedure

A damage was injected on posterior bearings in the plier mechanism, by partially removing material from the bearing surface, see Figure 70.

The blow molding machine operates at a set of production rates. We performed experiments at 1800 rph (revolutions per hour), where each test lasts about 3 min. The tests were performed both with all healthy pliers and a faulty one. The first preform arrives at the input pliers after 40 s, since they need to travel inside the oven. Thus, the first 40 s of the input pliers operation are without the load given by the heated preform.

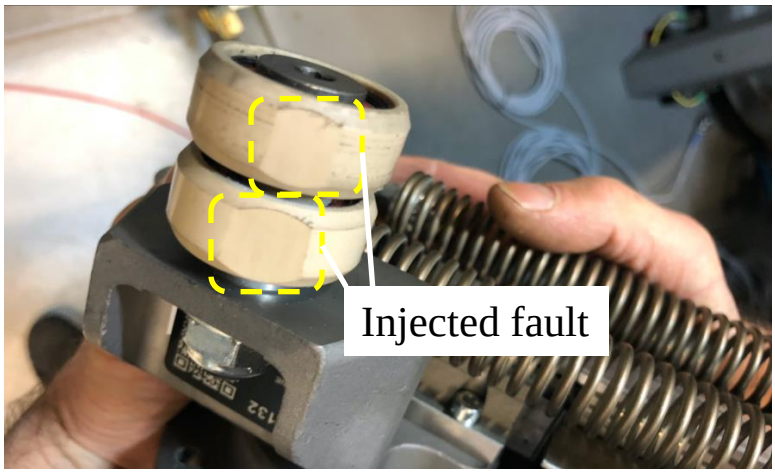


Figure 70: Fault injection of posterior pliers bearings.

As can be seen in Figure 70, the fault on the plier bearings can be interpreted as a fault on rolling balls inside a mechanical bearing. Thus, accelerometers are the best candidate sensor to monitor this specific kind of fault, see [62].

6.2.3 Sensors and data acquisition

We employed a single axis Hansford HS-170S piezoelectric accelerometer to measure the vibrations produced by the pliers during their operation. The accelerometer sense over the Z axis, which is the one orthogonal to the rotation axis of the pliers bearings. Since the pliers rotate, it was necessary to insert the accelerometer on the fixed structure that supports the pliers mechanics, see Figure 71. The accelerometers data are then acquired at 12.8 kHz using a NI CompactDAQ hardware.

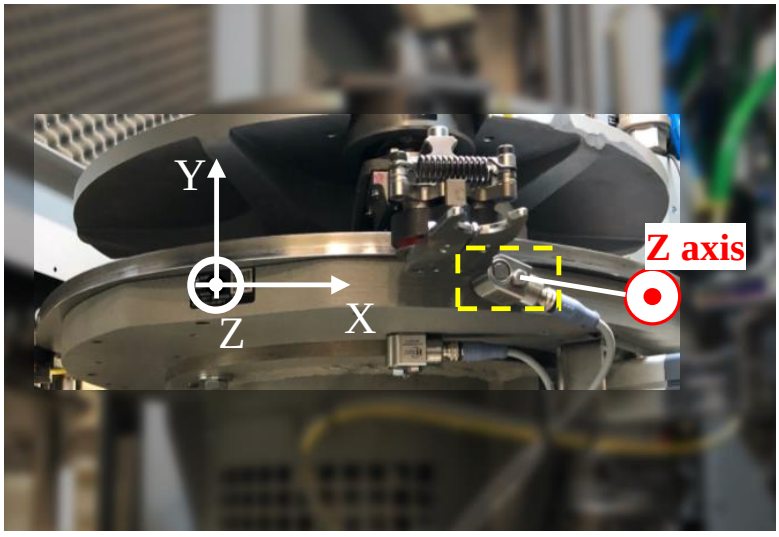


Figure 71: Considered accelerometer over the Z axis and its positioning on the pliers structure.

The blow molding machine is able to store variables related to the motors that actuate the input/output pliers and the carousel. Specifically, it is possible to measure the following motor-related variables: (i) quadrature current, proportional to their torque; (ii) reference and measured positions; (iii) temperature. Furthermore, a *binary indicator signal* $i(t)$ is present, that changes logical status (from 0 to 1) when a plier passes in front of a Infrared Sensor sensor, where t is the time index. Thus, there are 4 impulses of this indicator signal for each complete round of the input/output pliers. These machine-related signals are sampled sampled at 1 kHz.

Due to the fact that the accelerometers and the machine-related signals are sampled with different sampling frequencies, a synchronization signal has been devised to synchronize the accelerometer and motors measurements.

6.3 Fault detection of gripping pliers

The accelerometer signals $a(t)$ can be divided into segments relying on the indicator signal $i(t)$. Each segment contains the data between two consecutive impulses in $i(t)$. Each complete round of the pliers is thus divided into four segments, that correspond to the four "working quadrants" of the gripping plier mechanism, see Figure 72.

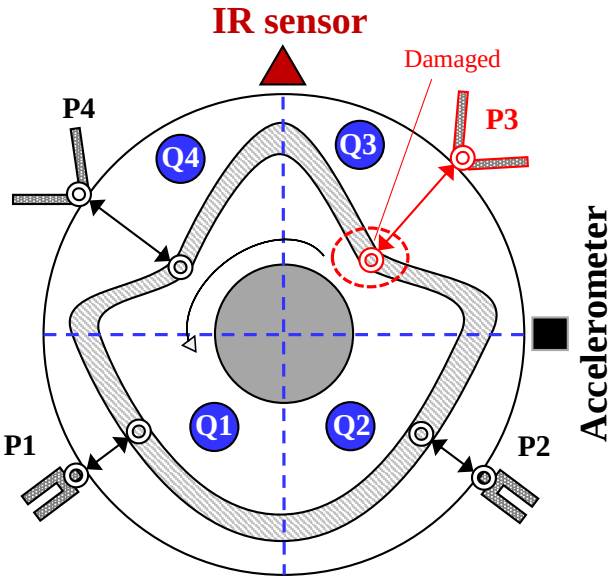


Figure 72: Schematic of the pliers mechanism. The plier P3 is supposed to have damaged posterior bearings.

Suppose that the damaged plier is the third one (**P3**). The bearings of the damaged plier are in close contact with the inner race of the cam where the springs of the damaged plier are stretched and its

arms are open. This happens in the in the **Q3** and **Q4** quadrants. So, we expect the accelerometer measurements to be sensitive to the fault in these operating conditions.
The main idea behind the fault detection algorithm is to *consider the entire gripping pliers machinery as like as a rolling bearing structure*. The analogies between these two mechanical systems are:

- *Posterior bearings* that support the pliers are thought as a *ball* inside a rolling bearing;
- *Inner race of mechanical cam* of the pliers is thought as the inner race of the bearing;
- *Outer race of mechanical cam* of the pliers is thought as the outer race of the bearing.

Using this abstraction, the accelerometer signals are processed with techniques inspired from bearing fault diagnosis.

Remark 6.1

As in standard bearing diagnosis, the accelerometer data are relative to a constant rotation speed.

6.3.1 Fault detection algorithm

Inspired from bearing diagnosis, see [146], the proposed algorithm for posterior bearings fault detection in input gripping pliers involves the following steps: (i) filtering the raw accelerometer data; (ii) envelope analysis; (iii) computation of fault indicators.

Data filtering

As a first processing step, it is often useful to bandpass-filter the raw vibration signal, in order to enhance the fault symptoms with respect to background noise and normal operational vibrations. The Spectral Kurtosis (SK) provides a mean to determine which frequency bands contain a signal of "highest impulsiveness". These impulsive behaviours are supposed to be originated from a fault. The SK algorithm divides the spectrogram of the signal in frequency bands. For each of these frequency bands, the kurtosis with respect to time is computed. The result is a kurtosis as function of the frequency. The kurtogram plot allows to evaluate the kurtosis for different frequencies and frequency resolutions (length of the frequency window considered for the computation). The kurtogram is used to select the frequency band $[f_c - b_f, f_c + b_f]$ for filtering the raw vibration signal $a(t)$ into its filtered version $r(t)$.

Envelope computation

A consolidated technique is that of envelope analysis, where the signal is amplitude demodulated to form the envelope $r_e(t)$, that can be more suitable for diagnostic purposes. In standard bearing analysis, the spectrum of the envelope is computed to look for specific fault frequencies. In our case, time-domain indicators are more useful since, due to the low rotation speed, it is hard to distinguish frequency components sensitive to the fault.

Fault indicators extraction

We propose two indicators to monitor the posterior bearings of the pliers:

1. The kurtosis values K of the envelope signal $r_e(t)$;
2. The Root Mean Square (RMS) value R of $r_e(t)$.

The steps for computing the indicators are summarized in Algorithm 7. Then, fault detection is achieved by comparing one or more of the indicators with specified thresholds defined on healthy data.

Algorithm 7: Fault indicators for gripping pliers

Input: $a(t)$, f_c, b_f

$r(t) \leftarrow$ Filter the signal $a(t)$ in $[f_c - b_f, f_c + b_f]$ Hz
$r_e(t) \leftarrow$ Compute the envelope signal of $r(t)$
$K \leftarrow$ Extract the Kurtosis of $r_e(t)$
$R \leftarrow$ Extract the RMS of $r_e(t)$
$R^q \leftarrow$ Extract the RMS of each quadrant q of $r_e(t)$

Output: K, R

6.4 Experimental results

The experimental campaign, as described in Section 6.2.2, is conducted only on the input gripping pliers. Figure 73 represents the first 40 s of healthy and faulty vibration signals. Although a difference is already visible on raw data, the proposed processing steps allow to enhance the diagnostic capabilities of the extracted indicators.

To this end, the raw signal $a(t)$ is bandpass filtered after bandwith selection with the kurtogram method. Figure 74 depicts the kurtogram of a faulty signal, where the optimal bandwith with $f_c = 1533$ Hz and $b_f = 133$ Hz is highlighted.

Figure 75 depicts the computed envelope $r_e(t)$ of the filtered vibration signal $r(t)$. Based on this, the kurtosis and RMS indicators are computed on two healthy experiments (test **H1** and test **H2**) and

a faulty one (test **F**). The results, summarized in Table 6, indicate that the kurtosis is particularly sensitive to the fault, with a percentage variation of about 1617% between healthy and faulty tests. This variation is computed (both for K and R) between the average value of the indicators from the two healthy tests, and the indicator from the faulty test.

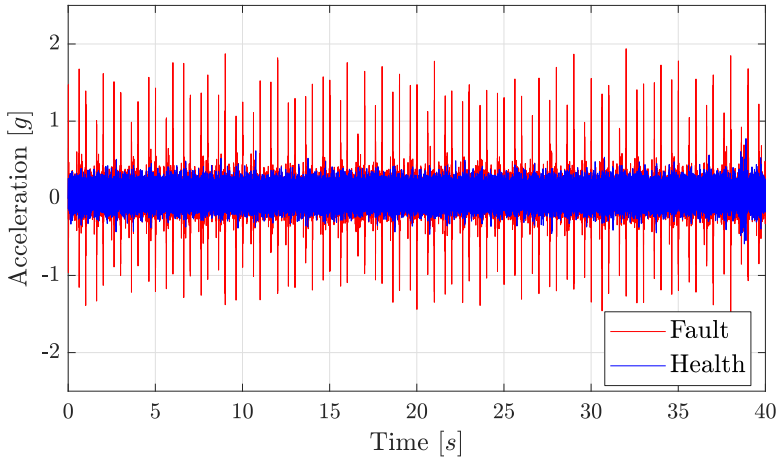


Figure 73: Example of healthy and faulty vibration signals.

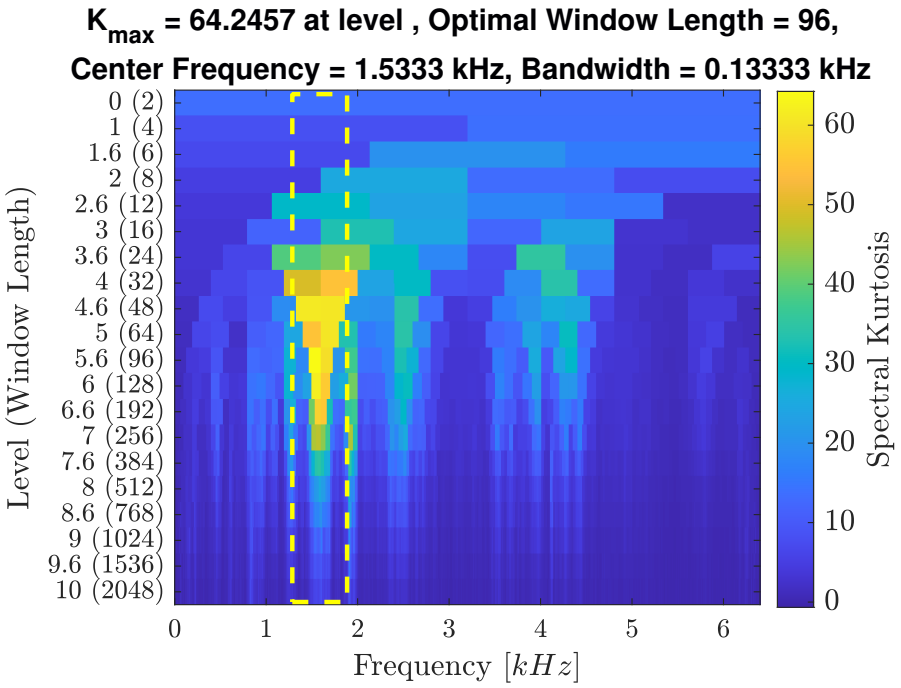


Figure 74: Kurtogram on faulty data, with indication of the best filtering bandwith.

To better assess the validity of the proposed rationale, we selected the segments of data over one full round of the pliers, for one healthy and one faulty test. These segments were aligned over a common time axis for visualization purposes in Figure 76.

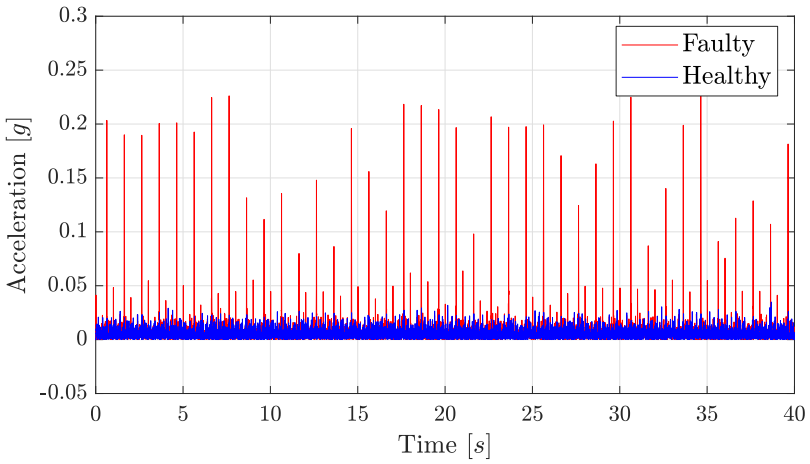


Figure 75: Envelope of healthy and faulty vibration signals.

	H1	*H2*	*F*	*% variation*
Kurtosis	6.1	5.7	101.3	+1617%
RMS	0.0075	0.0075	0.0165	+120%

Table 6: Kurtosis and RMS of $r_e(t)$ for each tests.

First, it can be noticed that the envelope signal presents a very high repeatability. Second, the faulty envelope clearly presents fault symptoms when the damaged plier **P3** passes in the quadrants **Q3** and **Q4**. We detect higher spikes when **P3** steps over **Q3** since the accelerometer is placed closer to **Q3**. When **P3** steps over **Q4**, we detect a lower spike.

Thus, a further possibility would be to compute the kurtosis or the RMS indicators *only in the quadrants* **Q3** or **Q4**, i.e. where the springs stretches and fault is more detectable. Figure 77 shows the boxplots of RMS value of the signal portions depicted in Figure 76, in each one of the four quadrants. The plot suggests the same conclusions of Figure 76, i.e. the fault is mainly detectable in **Q3**, due to the proximity of the accelerometer to the regions of the plier mechanics that are most sensible to the fault.

Remark 6.2

Fault isolation, that is, to understand which plier is faulty, can be accomplished by an additional sensor able to differentiate the passage of one plier with respect to the others, e.g. by generating an impulsive signal with a different pulse length for one of the pliers.

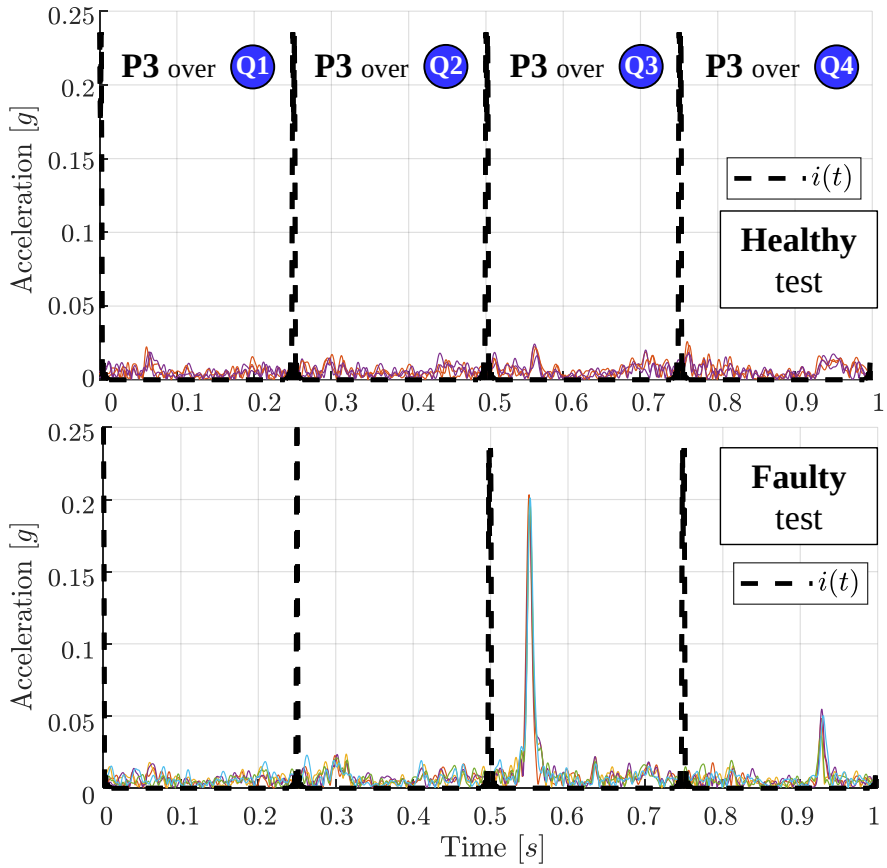


Figure 76: Overlapped portions of data for each full round of the pliers. The indicator signal $i(t)$ groups the data into the four operating quadrant of the pliers. The passages of the faulty plier P3 over the quadrants are showed.

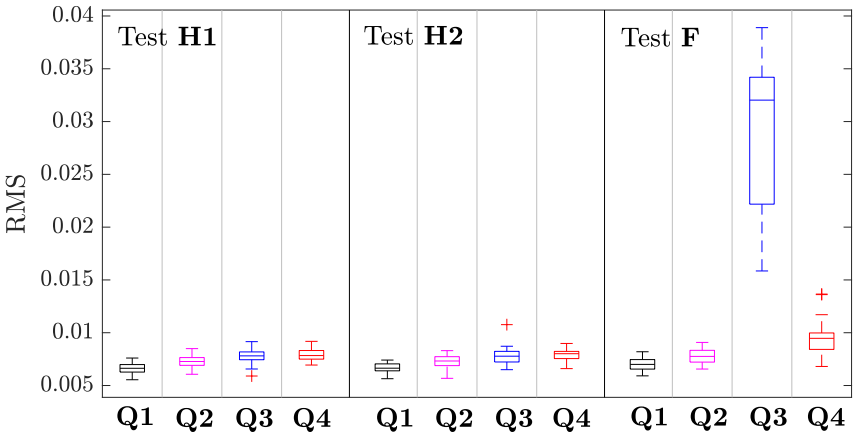


Figure 77: Boxplots of the RMS of the envelope signal over each quadrant.

6.5 Conclusions

We presented a signal-based approach for detecting the fault of the posterior bearings in an input plier mechanism for a blow molding machine in a bottling system. The approach employs accelerometers

data, processed with a workflow inspired by the diagnostic of mechanical bearings. Experimental data validated the effectiveness of the fault detection method.

Chapter 7. Model-based fault diagnosis of sliding gates electro-mechanical actuators transmission components with motor-side measurements

This chapter presents a model-based fault detection and isolation scheme for the transmission components of Electro-Mechanical Actuator (EMA), applied to the actuation of sliding gates. The most important failures are investigated by a Failures Mode, Effects and Criticality Analysis (FMECA) procedure. Following FMECA, the components selected for the development of the diagnostic algorithm are the nylon gear and pinion of the EMA, and the rack of the gate. The proposed diagnostic algorithm is able to isolate two out of the three types of faults. The overall procedure is validated by experimental results.

7.1 Introduction

The present work is devoted to the development of a model-based fault detection and isolation algorithm for the transmission components of Electro-Mechanical Actuator (EMA). Specifically, we mainly focus on the *diagnosis of gear wheels-like transmission components*. The proposed diagnostic scheme is applied to an experimental setup of EMAs that actuate sliding gates, where components, and their failures, have been selected using a Failures Mode, Effects and Criticality Analysis (FMECA). This applicative context is characterized by the employment of low-cost components. Thus, a major challenge is to detect faults without additional measurements.

The diagnosis of gears has been the focus of many studies in the literature, see [62, Chapter 5.4]. Starting from [147], who proposed a number of indicators for gearbox diagnostics based on Time Synchronous Averaging (TSA) and frequency analysis of a vibration signal, many other works followed with same diagnostic rationale, see [148, 149]. In [150], the author introduces the use of Hilbert transformation to demodulate the vibration signal for a diagnostic purpose. The computation of a signal envelope by amplitude demodulation is one of the main techniques for detecting faults also in rolling bearings [151, 29, 62]. The aforementioned works consider the measurement of a vibration signal. Other methods for gear diagnosis make use of the so-called *transmission error* signal [62, Chapter 5.4], and then analyze its envelope in frequency domain, as in [152]. The transmission error represents the difference between the angular motion of a driven gear and that which it would have if the transmission were perfectly conjugate, that is constant speed out for constant speed in.

However, as in our application, it is not possible to compute this signal, since e.g. only a single encoder is present (usually motor-side), and it is not possible to add accelerometers due to the low-cost of the EMA equipments and envelope constraints, see [153]. The use of only motor-related measurements to perform diagnosis of EMA has been previously investigated in the literature, especially for critical applications such as the aerospace industry [154, 155]. Contrary to signal-based or knowledge-based methods for fault diagnosis, as in [156, 157], here we propose a *model-based fault detection and isolation* scheme for EMA, that *relies only on common motor-side measurements* such as input voltage and motor speed.

The main contribution of this work relies in the *residual evaluation*: after the residual signal has been generated, it is processed by an envelope analysis to highlight the main fault frequencies that can raise when a component of the transmission chain is damaged. Then, a classifier is trained on features extracted on the frequency representation of the residual signal envelope. The designed classifier isolates two out of three faults, along with the healthy condition.

7.2 Failure Mode, Effects, and Criticality Analysis

In this work, we applied a Failures Mode, Effects and Criticality Analysis (FMECA) to investigate the failures of the EMA and gate system. The FMECA is a reliability procedure that determines all potential failure modes of the various system's components, their effects, causes and degree of criticality. The criticality analysis in the FMECA aids to define the so-called *criticality matrix*, used to classify the failures into a severity and frequency of occurrence levels, see [158]. This classification will be used to guide the selection of which failures should be considered by the proposed diagnostic algorithm.

The most occurring failures, appearing in years of maintenance reports, are: (1) Rack with broken tooth; (2) Pinion with broken tooth; (3) Broken shaft; (4) Broken nylon gear; (5) Short circuit of Direct Current Motor; (6) Corrupted wheel. The resulting criticality matrix is depicted in Figure 78. By using this representation, each failure is allocated to a matrix cell, with assigned probability level (frequency of occurrence) and a severity level (criticality). The probability level is qualitatively defined as (A) probable, (B) remote, (C) extremely remote, (D) extremely improbable. The severity level is qualitatively described as indicating (I) catastrophic, (II) hazardous, (III) major and (IV) minor consequences, respectively. These qualitative levels should be understood as dependent on the specific system considered, see [29, 158, 159].

The (3) Broken shaft failure is classified in the cell (C-I) in the probability/severity table, since it occurs with extremely remote frequency and, moreover, it is a catastrophic phenomenon, because

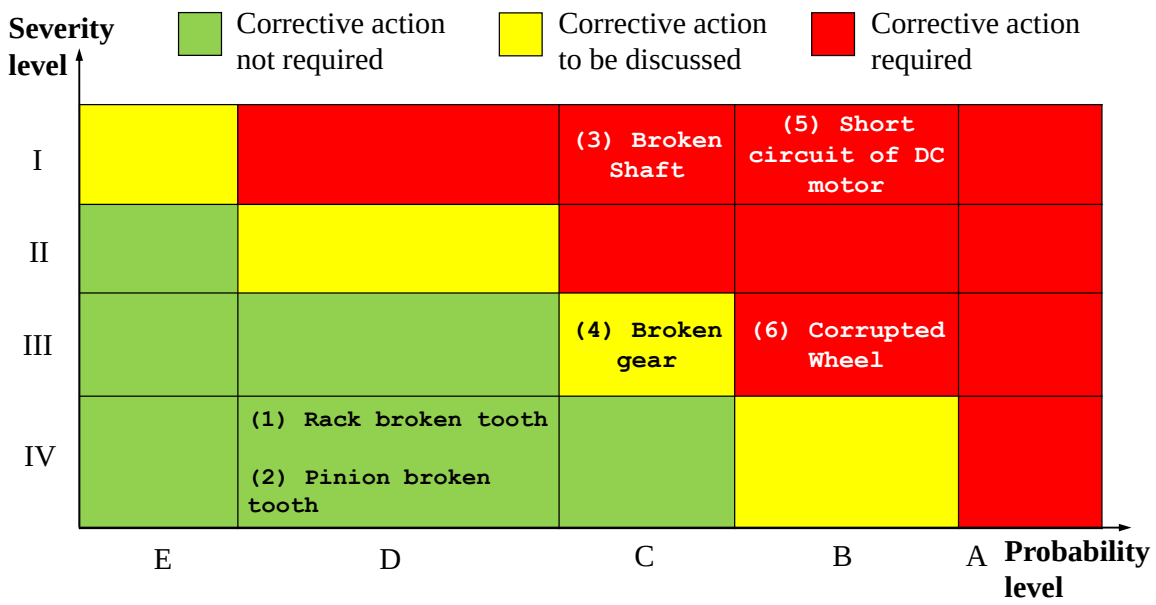


Figure 78: Criticality matrix resulting from FMECA on the considered actuation system.

the shaft is disrupted into two pieces. Thus, in this condition, the EMA does not work. Another catastrophic event, that compromises the electronic of Direct Current Motor (DCM), is the failure (5) Short circuit of the DCM. Since this failure happens remotely, it was classified as (B-I).

Two failures that are not catastrophic, but still have major consequences on the system, are: (4) broken gear, classified as (C-III) and (6) corrupted wheel, classified as (B-III). These failures do not stop the EMA operation, but they make difficult the sliding of the gate. The operators common knowledge is that a broken nylon gear is the result of a natural notch on the gear that breaks through the entire gear radius. Due to this, the gear structure becomes weaker. After some movements, the less solid gear causes a misalignment that induces other cracks in the gear structure, leading to its disruption. Instead, the (6) corrupted wheel fault is due to the environmental corrosion that exhausts the wheel bearing. The failures that have minor consequences on the system are: (1) Rack broken tooth and (2) Pinion broken tooth, both classified as (D-IV). These are failures of the same type, i.e. a tooth breaks totally, on two different but connected components, see Figure 81. The EMA works properly with these failures.

The criticality matrix is used to assign a qualitative risk level for each failure, denoted by the cell color in Figure 78. Green and yellow cells contain the failures that do not require redesign of the actuation system. Instead, the red cells indicate the failures that can be prevented only by a system redesign, since those failures are too critical and/or too probable. The focus of the diagnostic algorithms will therefore be on the *failures that are not at the maximum risk level*, i.e. those in the green and yellow

cells of the criticality matrix: (1) Rack with broken tooth, (2) Pinion with broken tooth and (4) Broken nylon gear.

7.3 Experimental setup

The system setup consists of a sliding gate actuated by a Direct Current Motor (DCM) with nominal voltage of $V_0 = 24$ V. The control of the DCM is made possible by a Pulse-Width Modulation (PWM) at 100 Hz of the input voltage signal. The gate moves by means of steel wheels on a steel rail. The motor is connected to the gate through a transmission that converts the DCM rotation to a linear movement. The transmission is composed of: (i) a worm gear, (ii) a nylon gear, (iii) a shaft, (iv) a pinion and (v) a rack. We denote with the term Electro-Mechanical Actuator (EMA) the connection of the DCM with the transmission elements (i)-(iv), while the rack is a component that belongs to the gate.

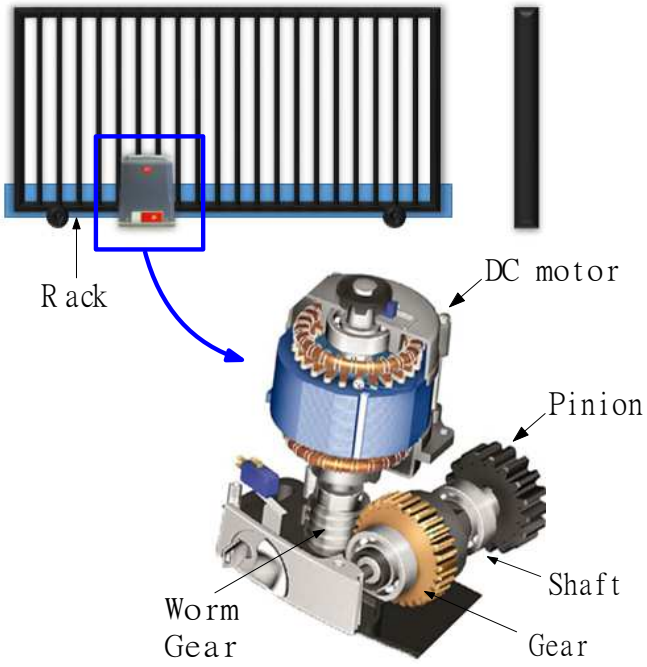


Figure 79: Schematic representation of the overall system.

Figure 79 depicts how these elements are connected. In particular, the worm gear is welded to the rotor of the DCM and it is coupled to the nylon gear with a primitive radius of $r = 28 \cdot 10^{-3}$ m, that consists of 44 teeth, see Figure 80-(left). Since the rack is external to the EMA cover, a shaft connects the gear to the pinion, which in turn it is paired with the gate's rack. The rotation at the output of the EMA is transformed into linear motion by the pinon and the rack. The pinion is made of stainless steel and it has 14 teeth, see Figure 80-(right). An encoder measures the motor speed $\omega_M(t)$. The conversion from $\omega_M(t)$ to axial speed $v_M(t)$ can be made by the transmission ratio $\bar{\tau} = 1/47 \cdot r$.

The motor resistance is $R = 0.7473\,\Omega$, and $k_t = k_e = K = 0.0696$ are the mechanical and electrical constants of the motor, respectively.

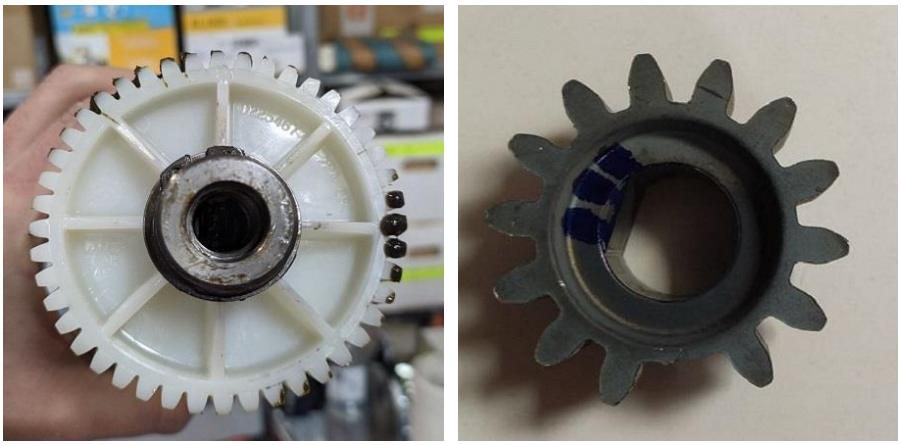


Figure 80: Healthy nylon gear (left) and pinion (right).

7.4 Fault injection and test protocol

7.4.1 Fault Injection

A fault injection procedure has been devised in order to collect measurements from a faulty system, considering the rack, pinion and nylon gear components. The rack and pinion faults were injected by removing a tooth using a vise, see Figure 81-(left) and 81-(right) respectively.

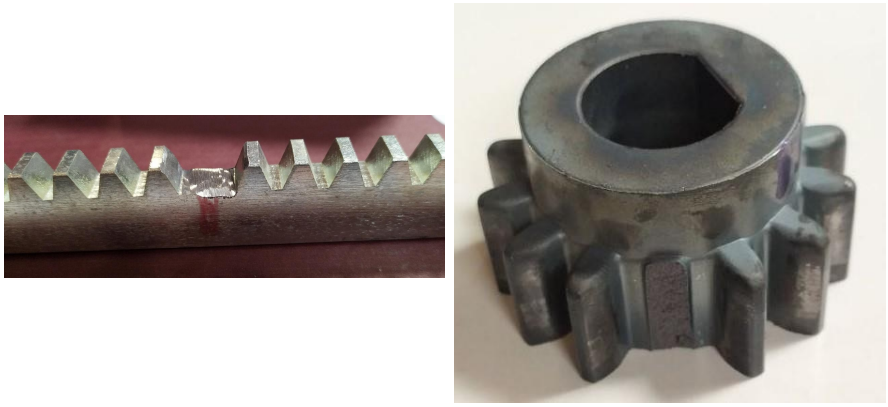


Figure 81: Faulty rack (left) and pinion (right).

To reproduce the break of the nylon gear, we performed about one hundred gate movements (opening and closing) to break-in the component. After that, the fault is injected by carving perpendicularly the 80% of the total gear radius using a saw. The width of this notch is about 1 mm, see Figure 82-(left) where the area that contains the injected notch is highlighted in blue. In Figure 82-(left), it is possible

to notice four mechanical housings, which aim is to hold the shaft joint. These are the most critical parts of the gear because they are subject to the force applied by the shaft through its joint. Thus, the fault is injected in this area. In this notched condition, the inner ring of the gear, i.e. the part that delimits the shaft slot, is still not broken. Thus, to induce its breakage, the carved gear is mounted on the EMA and about fifty openings and fifty closings are performed.

Figure 82-(right) represents the condition of the gear after the 100 movements. As depicted, the width of the natural notch is less than the width of the artificial one (highlighted in blue). It is important to remark that if the inner ring is completely broken by an artificial carve, e.g. with depth 100% of the gear radius, the structure would be too weak to perform any useful experiment.

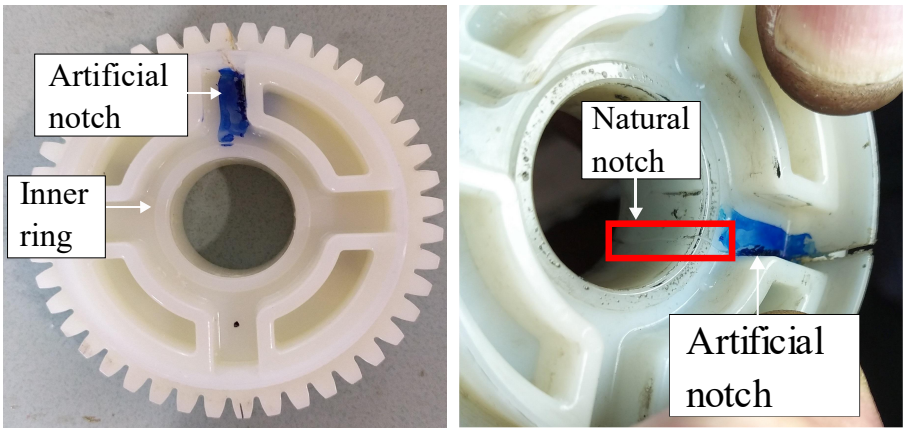


Figure 82: Faulty nylon gear without breaking the inner ring (left); natural notch that breaks the inner ring (right).

7.4.2 Test protocol

The experimental protocol is composed of five different test plans: (i) healthy tests; (ii) gear fault tests; (iii) pinion fault tests; (iv) rack fault tests. All experiments share the same gate, binary and environment, but the fault injected is different. Only one fault at a time has been considered. In order to validate the diagnostic algorithm, the test protocol is performed twice with two different EMA.

Each test plan consist in opening and closing gate movements, interspersed with a break of 7 seconds, in order to not overheat the motor. The motor is commanded in open-loop with trapezoidal voltage profiles, that define acceleration, constant speed, and deceleration phases. The rise and fall times of the acceleration and deceleration phases have been set to 1 s (the minimum settable acceleration/deceleration time). This choice is motivated by two ideas: the first one is that we wanted to perform movements that were stressful for the system (to enhance the fault detectability); the second reason

regards the practical use of the gates, where the fastest opening and closing movements (but within laws regulations) are usually desirable.

The EMA hardware allows the acquisition of the following measurements with a sampling frequency of $f_s = 5000\,\text{Hz}$:

1. *Motor speed* $\omega_M(t)$, measured by the motor encoder;

2. *Motor working phase* $p(t)$, showing which working phase the motor is currently performing (acceleration phase, constant velocity phase, deceleration phase);

3. *Motor current* $i(t)$, flowing in the DCM coils;

4. *Motor voltage* $V(t)$, powering the motor.

7.5 Model-based fault detection and isolation algorithm

This section describes the procedures employed to design the fault detection algorithm. Since, the chosen fault detection methodology is model-based, the first part describes the modeling and identification procedures. Instead the second part focus on the residual generation, evaluation and decision logic.

7.5.1 Modeling and identification

A mathematical model of the DCM is presented in Figure 83, where: $F_L(t)$ is the load force opposing to the motor; L is the motor inductance; R is the motor resistance; D is the motor equivalent friction coefficient; J is the motor equivalent inertia; $\bar{\tau}$ is the transmission ratio; $K = k_t = k_e$ represents the motor mechanical and electrical constants; $V(t)$, $\hat{i}(t)$, $\hat{T}_M(t)$, $\hat{\omega}_M(t)$, $\hat{v}_M(t)$ are the applied voltage input and simulated current, torque, motor rotational speed and load axial speed, respectively.

It is interesting to notice that most of the parameters of this model are known from the motor datasheet or can be computed. The motor equivalent inertia can be expressed, in this rigid model, as $J = J_M + \bar{\tau}^2 \cdot m$, where J_M is the motor inertia and m is the weight of the gate. Both are known and correspond to $J_M = 3.5 \cdot 10^{-5}\,\text{kg} \cdot \text{m}$ and $m = 608\,\text{kg}$, respectively. To obtain the simplest possible model, we set $L = D = 0$.

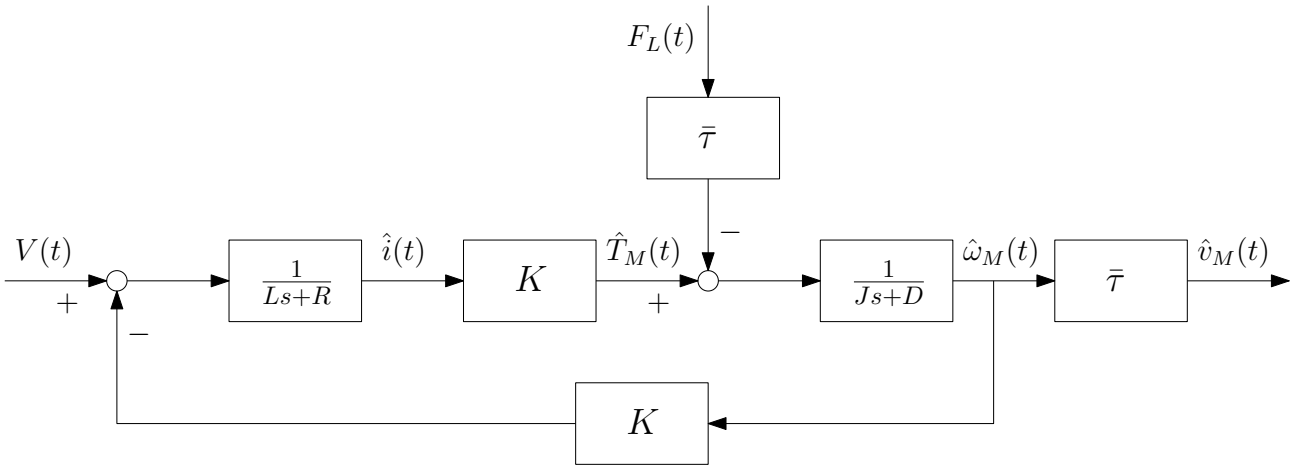


Figure 83: Blocks scheme of the DC motor model.

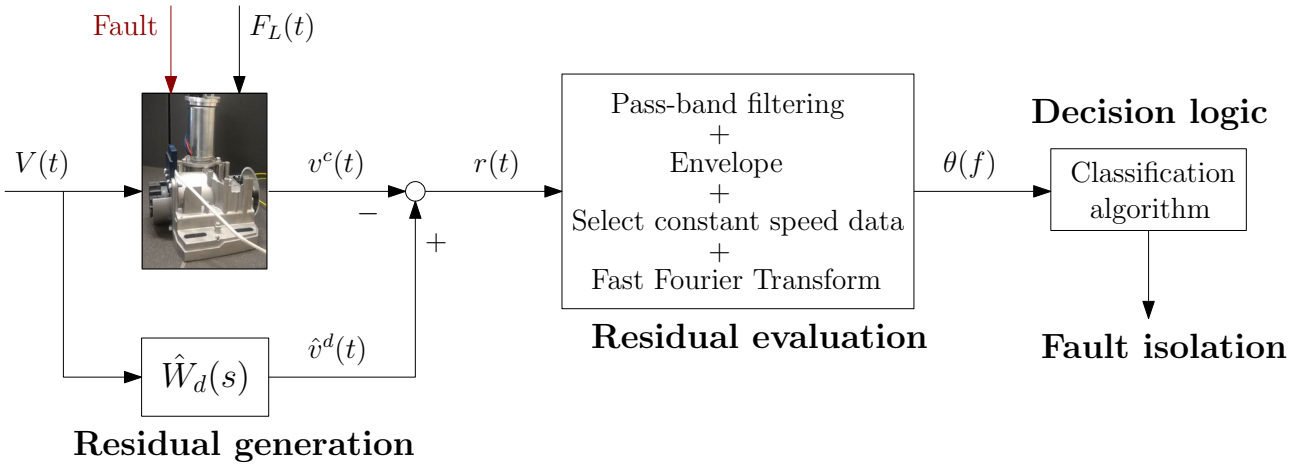


Figure 84: Proposed model-based fault diagnosis scheme.

From these hypotheses, it is possible to compute the complete transfer function from the input $V(t)$ to the output $\hat{v}_M(t)$ as:

$$\hat{W}(s) = \left(\hat{G}(s) - \hat{A}(s)\right) \cdot \bar{\tau} \tag{7.1a}$$

$$= \left(\frac{K}{JR \cdot s + K^2} - \frac{R\bar{\tau}}{JR \cdot s + K^2}\right) \cdot \bar{\tau}, \tag{7.1b}$$

where:

- $\hat{G}(s) = \frac{\Omega_M(s)}{V(s)}$ is the estimated transfer function from voltage $V(t)$ to motor speed $\omega_M(t)$;
- $\hat{A}(s) = \frac{\Omega_M(s)}{F_L(s)}$ is the estimated transfer function from the load force $F_L(t)$ to motor speed $\omega_M(t)$.

The force $F_L(t) = m \cdot g \cdot c_{\text{steel}}$ represents the sliding friction force of steel wheels on steel rail, with friction coefficient $c_{\text{steel}} = 3 \cdot 10^{-4}$ (see [160]), and g is the gravitational force. We assume $F(t)$ to be a constant force equal to $F_L(t) = \bar{F}_L = 0.19\,\text{N}$.

The model (7.1) is thus completely known, but experimental data reveal the presence of an input-output delay. Thus, we identified a model of the form:

$$\hat{G}_d(s) = \frac{\mu}{T \cdot s + 1} \cdot e^{-s \cdot d} \,, \tag{7.2}$$

where μ is the gain of $\hat{G}(s)$, T is the time constant of $\hat{G}(s)$ and d is the delay of the system[1]. A gray-box simulation error minimization is performed by minimizing the cost

$$J(d) = \frac{1}{N} \sum_{i=1}^{N} \left(v^c(t) - \hat{v}_d(t) \right)^2 , \tag{7.3}$$

where the computed load axial speed $v^c(t)$ is obtained as $v^c(t) = \omega_M(t) \cdot \bar{\tau}$ (by considering the transmission as rigid), and $\hat{v}_d(t)$ is the simulated output of the model

$$\hat{W}_d(s) = \hat{G}_d(s) - \hat{A}(s) \,. \tag{7.4}$$

The behaviour of the model in (7.4), for a opening gate movement not used for the identification, is depicted in Figure 85, where good simulation results can be observed.

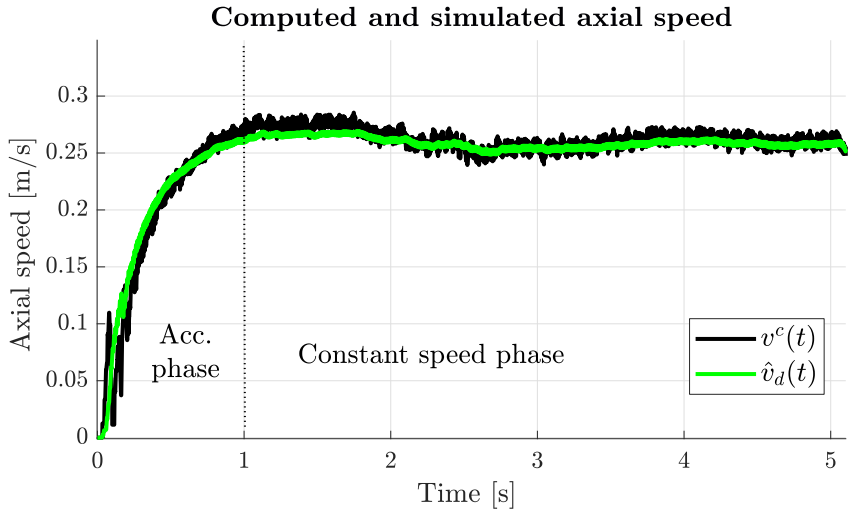


Figure 85: Computed axial speed $v^c(t)$ from measurements and simulated axial speed $\hat{v}_d(t)$ by the model $\hat{W}_d(s)$.

7.5.2 Model-based fault detection and isolation scheme

Residual generation

Figure 84 shows the proposed model-based fault detection strategy. The residual $r(t)$ is computed by an output-error based scheme, see [48, Chapter 10], that compares the computed axial speed $v^c(t)$

[1] In the literature there exists a fault diagnosis method that detect faults through recursive parameter estimations, see [161]. This approach needs an input that excites the system sufficiently. Therefore, it is not applicable to this problem, because the available step input is not enough exciting to be used in black-box identification procedures.

(computed from the measure of $\omega_M(t)$), with the simulated output $\hat{v}_d(t)$ of model (7.4), given the same input $V(t)$:

$$r(t) = v^c(t) - \hat{v}_d(t) \ . \tag{7.5}$$

The employment of the model in (7.4) allows to generate a residual signal that is able to *compensate for variations in speed*, even in the constant speed phase where speed oscillations are visible, see Figure 85. The disturbance $F_L(t)$ can not be decoupled from the residual since we only use one output measurement to perform fault detection. Instead, there exists diagnostic schemes that try to reduce the effect of disturbances on the residuals, see [28, Chapter 7], in our application the effect of $F_L(t)$ on the true axial speed $v_M(t)$ is negligible since the gain of $\hat{A}(s)$ can be found to be $\mu_A = {R \cdot \bar{\tau}^2}/{K^2} \approx 5 \cdot 10^{-4}$, so that the signal $F_L(t)$, filtered through $\hat{A}(s)$, as an amplitude in the order of 10^{-5} m/s, which is negligible with respect to the amplitude of $v_M(t)$ (around 10^{-1} m/s).

Residual evaluation

The proposed residual evaluation scheme, that is the main methodological contribution of this work, allows to *detect all three fault types* and to *isolate them in two categories*. The starting point is the observation that the mechanical transmission components of interest (e.g. rack, pinion and nylon gear) behave as a single component, since they are all connected together and also linked to the motor through the nylon gear. We thus expect that faults on the selected transmission components can be detected by using the same set of input/output measurements. Furthermore, the nylon gear and the pinion are basically gear wheels, where a tooth is removed by the fault injection procedure. As stated in the introduction, the envelope analysis of the transmission error can be employed to diagnose faults in gears.

The main idea of this work is to *apply the envelope analysis on the residual signal* $r(t)$ in (7.5). In this view, the signal $\hat{v}_d(t)$ can be effectively interpreted as the axial speed that one would have if the transmission were perfectly health and rigid, thus acting as a "virtual load encoder" signal. The true axial speed $v_M(t)$ will be affected by faults, but since the components are all linked together, we expect that also the computed axial speed $v^c(t)$ (or equivalently, the measured motor speed $\omega_M(t)$) will be affected.

The aim of the envelope analysis is to look for specific fault frequencies, as commonly done for rolling bearings [151, 62]. Here, we look for the fault frequency f_{fault} that may appear on the shaft that connects the nylon gear to the steel pinion (which is then linked to the rack of the gate). Since, in our experiments, the rotational motor speed is known and it is about $\omega_M(t) = 4100\,\text{rpm}$, the shaft rotational speed is ${4100\,\text{rpm}}/{44} \approx 93\,\text{rpm}$, where 44 is the ratio reduction from motor to shaft (i.e. the

number of teeth of the nylon gear). Hence, the fault frequency of the components that are coupled to the shaft is about $f_{\text{fault}} = 1.55\,\text{Hz}$.

The envelope analysis of the residual signal $r(t)$ proceeds as follows. First, a bandpass filtering with bandwidth $[0.5, 10]$ Hz is employed to remove the continuous frequency and high-frequency noise. Then, the envelope of the filtered $r(t)$ signal (amplitude demodulation) is computed and constant phase speed data are retained for frequency analysis. The use of constant speed data allows to focus on a single fault frequency without employing advanced techniques like orders tracking[2], see [62, Chapter 3.6]. The constant speed phase is always available in our application. A Fast Fourier Transform (FFT) of the envelope signal is then computed, and its modulus $\theta(f)$ analyzed for fault detection and isolation.

Finally, it is important to remark that the nylon gear fault and pinion fault (missing tooth) can be observed many times during the gate motion, but the faulty portion of the rack is visible only one time per gate movement (so that a frequency analysis is of limited utility in this case).

Decision logic

A linear Support Vector Machines (SVM) classification algorithm is used to perform fault isolation. To this end, two features are computed from $\theta(f)$:

$$F_1 = \sum_{k=1}^{3} \theta(k \cdot f_{\text{fault}}) \; ; \quad F_2 = \sum_{k=1}^{3} \sum_{j=k\cdot f_{\text{fault}}\cdot 0.95}^{k\cdot f_{\text{fault}}\cdot 1.05} \theta(j) \; . \tag{7.6}$$

The indicators in (7.6) extract the frequency amplitude at the first three harmonics of the fault frequency f_{fault} and the area in their neighborhoods, respectively. This idea is inspired from [162], where those features are computed on the motor current signal.

The SVM algorithm classifies the features in (7.6) into three classes: health, pinion fault + nylon gear fault, rack fault. The nylon gear and pinion faults are difficult to isolate since these components are connected to the same shaft, so their rotation frequency is the same. Thus, a single "fault condition" has been considered for their isolation.

7.6 Experimental results

As stated in Section 7.4, we have at disposal two datasets, from two exemplars of the same EMA model. The first dataset is used to tune the parameters of the proposed procedure, such as the filtering

[2]The importance of employing a model of the system relies in the fact that the residual signal in the constant speed phase, contrary to the axial speed $v_M(t)$, is much less subject to little variations due to external factors, so that the effects due to the faults can be better assessed.

band for the bandpass filter in the residual evaluation state. The training of the FFT classifier makes use of both the first and second datasets.

An example of FFT of the residual envelope $\theta(f)$ is depicted in Figure 86, where the fault frequency f_{fault} appear visible for the nylon gear and pinion fault. As expected, this frequency is not visible in the case of the rack fault, but a general increase of the frequency content is visible. Figure 87 depicts the

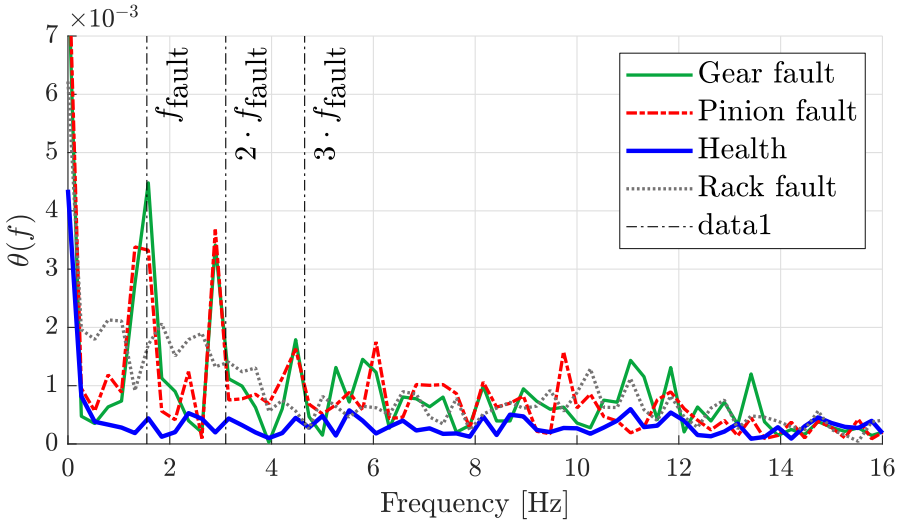


Figure 86: Frequency analysis of the residual envelope of all EMA conditions.

2D-plane composed of F_1 and F_2 features, computed using both datasets. The classifier performance have been evaluated by 10-fold cross-validation, resulting in an average cross-validation accuracy of 86.18% and variance of 0.33%. The classification boundaries, of the classifier *trained on all the data*, show a good capability of isolating the various kind of faulty conditions (considering the gear and pinion faults as a unique category), with an accuracy of 96.15%[3]. Table 7 presents the confusion matrix of this classifier, using all the data.

7.7 Conclusions

We presented a model-based fault detection and isolation algorithm for the transmission components of electro-mechanical actuators. The considered failures were chosen by a FMECA procedure, and specific faults were artificially injected or induced. The procedure employs only the input voltage and the motor speed, and it is based on the output speed residuals generated by a model of the DCM. By adequately considering the type of faults to be diagnosed, we proposed a residual evaluation strategy based on the envelope analysis of the residual signal. Based on its frequency content, specific features

[3]The model trained on all the data, its accuracy and confusion matrix should be considered as only a mean to highlight its decision boundaries and to represent the feature space. A more correct evaluation of the classifier's performance is given by the cross-validation routine.

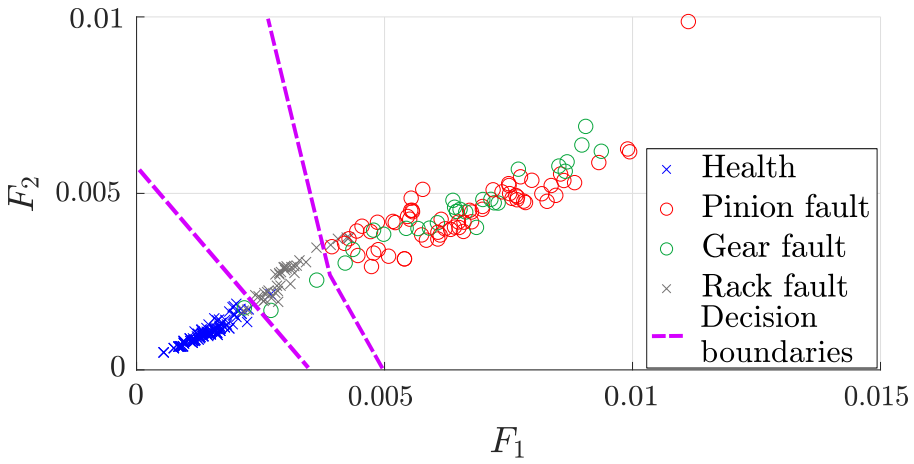


Figure 87: Features plane and classification boundaries.

		Real EMA state		
		Healthy	**Pinion and gear faults**	**Rack fault**
Estimated EMA state	**Healthy**	130	1	2
	Pinion and gear faults	0	89	4
	Rack fault	1	2	31

Table 7: Confusion matrix of the classifier trained on all the data.

have been extracted. An experimentally validated classifier trained on those features was shown to perform fault isolation of two out of three fault with good accuracy.

Conclusions and future developments

This book proposed three theoretical and three applicative contributions. Chapter 2 introduced a data-driven robust control. We studied how to link robust control and system identification worlds. In literature, there are some techniques, such as Robust identification, which tie these two kinds of literature, but their developments are highly related to the user's choices. Instead, we suggest an automatic data-driven methodology that produces a robust controller without requiring any user's knowledge. The kernel-based system identification played a key role in this procedure, since it builds a low-bias model without choosing the model family. Furthermore, it gives a posterior distribution that represents the model variance. Therefore, we exploited these two characteristics to make a robust control through the S/T mixed-sensitivity loop-shaping approach. The proposed methodology can be applied to robust stability-nominal performance and robust performance aims. We tested the method to a well-known benchmark dynamic system. The results proved the efficiency with respect to the traditional method, where the uncertainty information is modeled manually. In addition, a comparison with respect to the PEM is done. Specifically, this identification method was developed with a wrong and a correct model family, to show how the user's knowledge, applied to the identification procedure, affects the performance of the resulting robust controller. The simulations showed that a wrong model family choice can cause a catastrophic impact to the closed loop system obtained with the true plant coupled with the robust controller.

An extension of the presented method was described in Chapter 3. This dealt with multiple uncertainty sources, in particular with respect to: the model identification uncertainty and some parametric uncertainties considered as a multi-model dynamic system. The experimental design was fundamental in the extended method, since, as with the identification procedure of multi-model dynamic systems, it allows to gather the necessary information to model correctly the plant. We proposed an experimental design with the aim to obtain a nominal model endowed with a multiplicative unstructured uncertainty that represents all multiple uncertainties. In this way, we designed a robust control that works with all the systems that belong to the modeled uncertain system. In Chapter 5, we described a practical application of the data-driven robust control procedure for multi-model dynamic systems. This contribution showed the performance of the second theoretical contribution applied to a real system. Moreover, a comparison between the trial and error modeling (the traditional weight functions design) with respect to the proposed method showed that the obtainable performances, with the manual setting, were always worse than our method.

The last theoretical contribution was described in Chapter 4. We present a procedure that exploits the uncertainty information derived from the model identification to the robust fault detection problem. In particular, the uncertainty information was considered as a fictitious noise. So the uncertainty transfer function, derived from the kernel-based identification, was employed to define the impact of the fictitious noise to the residual signal. In addition, we designed the robust residual generator through the Approximate Fault Detection Problem (AFDP). Also, we proposed a simple threshold tuning to further reduce the false alarms. Our method was tested on a benchmark problem and the results were compared to a not robust method. The goal was achieved since the robust detector avoided false alarms and also detected those faults that were invisible to the not robust detector.

The other two applicative contributions was written in Chapters 6 and 7. Those showed two proposed fault detection methodologies. The former was based on the signal-based branch. In particular, we adapted an algorithm, developed for detecting the bearing faults, to diagnose a fault of some gripping pliers belonging to a rotating machine. Instead, the latter, described in Chapter 7, was based on the model-based fault diagnosis. We illustrated the entire process of fault diagnosis: from the Failures Mode, Effects and Criticality Analysis to the fault diagnosis algorithm, through the fault injection procedure. Furthermore, we presented a residual evaluation strategy based on the envelope analysis of the residual signal. The experimental results validated our method.

The future developments, identified by the authors, are devoted to investigate the employment of kernel-based identification to model the uncertainty given by the residual signal computed as the difference between the estimated output and the measured data, as in the Model Error Modeling (MEM) problem. Doing so, we would like to avoid the user's choices to model the dynamic system that ties the input of the system to the residual signal. This proposal should be an alternative way to our developed method. Another future development is to design our proposed data-driven robust control, presented in Section 2, for MIMO systems. Moreover, we would like to replace the Chernoff bound with a more efficient theorem, explained in [163]. It allows to obtain an uncertainty bound reconstruction with minor number of samples. Instead, for the robust fault diagnosis theme, future works are dedicated to comparing the presented method with respect to another robust fault diagnosis procedure. In addition, we would like to expand the proposed robust fault detection methodology to fault identification and fault estimation.

Appendices

Appendix A

Model complexity selection in parametric dynamic system identification

The *bias-variance decomposition* defines the relationship between bias, variance and the expected E_{out}, see [99, Chapter 3] [87, Chapter 2]. The latter is defined by:

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}\left[E_{out}\left(G^{(\mathcal{D})}\right)\right] &= \mathbb{E}_{\mathcal{D}}\left[\mathbb{E}_x\left[\left(G^{(\mathcal{D})}(\boldsymbol{x}) - G_T(\boldsymbol{x})\right)^2\right]\right] \\ &= \mathbb{E}_x\left[\mathbb{E}_{\mathcal{D}}\left[\left(G^{(\mathcal{D})}(\boldsymbol{x}) - G_T(\boldsymbol{x})\right)^2\right]\right] \\ &= \mathbb{E}_x\left[\mathbb{E}_{\mathcal{D}}\left[G^{(\mathcal{D})}(\boldsymbol{x})^2\right] - 2\mathbb{E}_{\mathcal{D}}\left[G^{(\mathcal{D})}(\boldsymbol{x})\right] G_T(\boldsymbol{x}) + G_T(\boldsymbol{x})^2\right]\end{aligned} \quad , \tag{A.1}$$

where:

- $\mathbb{E}_x$ denotes the expected value with respect to all possible points $\boldsymbol{x} \in \mathbb{R}^{n\times 1}$, where $\boldsymbol{x}$ is the input vector employed in the train phase;
- $\mathcal{D}$ is the training dataset;
- $\mathbb{E}_{\mathcal{D}}$ denotes the expected value with respect to all possible datasets $\mathcal{D}$;
- $G \in \mathcal{M}$ is a model that belongs to a chosen model family $\mathcal{M}$;
- $G^{(\mathcal{D})}$ are the identified model on different realizations of $\mathcal{D}$;
- G_T represents the true plant.

If we consider the conceptual tool: $\mathbb{E}_{\mathcal{D}}\left[G^{(\mathcal{D})}(\boldsymbol{x})\right]$ as $\bar{G}_q$ (i.e. the mean model obtained by averaging q models trained on q different dataset.), the equation (A.1) can be rewrite as:

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}\left[E_{out}\left(G^{(\mathcal{D})}\right)\right] &= \mathbb{E}_x\left[\mathbb{E}_{\mathcal{D}}\left[G^{(\mathcal{D})}(\boldsymbol{x})^2\right] - 2\bar{G}_q(\boldsymbol{x})G_T(\boldsymbol{x}) + G_T(\boldsymbol{x})^2\right] \\ &= \mathbb{E}_x\left[\mathbb{E}_{\mathcal{D}}\left[G^{(\mathcal{D})}(\boldsymbol{x})^2\right] - \bar{G}_q(\boldsymbol{x})^2 + \bar{G}_q(\boldsymbol{x})^2 - 2\bar{G}_q(\boldsymbol{x})G_T(\boldsymbol{x}) + G_T(\boldsymbol{x})^2\right] \\ &= \mathbb{E}_x\left[\text{variance}(\boldsymbol{x}) + \text{bias}(\boldsymbol{x})^2\right] \\ &= \text{variance} + \text{bias}^2\end{aligned} \quad , \tag{A.2}$$

with:

$$\text{bias}(\boldsymbol{x})^2 = \bar{G}_q(\boldsymbol{x})^2 - 2\bar{G}_q(\boldsymbol{x})G_T(\boldsymbol{x}) + G_T(\boldsymbol{x})^2 = (\bar{G}_q(\boldsymbol{x}) - G_T(\boldsymbol{x}))^2 \,. \tag{A.3}$$

This term measures the distance between the true model and the average model $\bar{G}_q$. Specifically, it contains both *model bias* and *estimation bias* [14, Chapter 7]. The model bias is due to the poor complexity of the model family, instead the estimation bias represents the estimation error of the model parameters. Specifically, the conceptual tool $\bar{G}_q$ may not belong to $\mathcal{M}$. Since $\bar{G}_q$ is made by averaging the q models that belong to $\mathcal{M}$, the only limit of $\bar{G}_q$ are the bounds imposed by $\mathcal{M}$ therefore, if $q \to \infty$. Thus, the bias represents how much the model family can approximate the real model.
The variance is:

$$\text{variance}(\boldsymbol{x}) = \mathbb{E}_{\mathcal{D}}\left[G^{(\mathcal{D})}(\boldsymbol{x})^2\right] - \bar{G}_q(\boldsymbol{x})^2 = \mathbb{E}_{\mathcal{D}}\left[\left(G^{(\mathcal{D})}(\boldsymbol{x}) - \bar{G}_q(\boldsymbol{x})\right)^2\right] \,. \tag{A.4}$$

This term corresponds to a measure of the deviation between $\bar{G}_q$ and the trained model estimated with a defined $\mathcal{D}$.
A graphical representation of bias and variance is reported in Figure 88. The left image shows the graphical concept of bias, i.e. the model family $\mathcal{M}$ is too small and so $\mathcal{M}$ does not include the real model G_T. Instead, the right image depicts the variance, i.e. the area highlighted in red. Note that the real model stands in the model family space, but since the model family is too high, the obtainable model changes by varying $\mathcal{D}$. The set of obtainable models represents the variance.

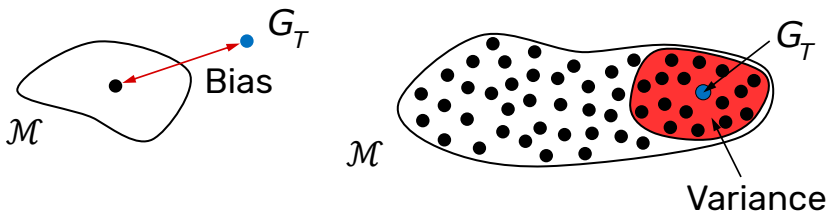


Figure 88: The graphical representation of the bias and variance concepts.

Bias-variance decomposition is a conceptual tool and it is not applicable to a real problem, since it requires the true plant G_T, which is usually unknown. Hence, this technique cannot be used for the model complexity selection. In literature, there are different methodologies that are usable with a real plant. Some of these are:

- Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC),
- Cross Validation (CV),
- Regularization.

A.1 Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC)

The model order selection can be performed by exploiting a metric which is proportional to the complexity of the model, if the model family is fixed and the available dataset is limited. The Akaike Information Criterion (AIC) is a metric term employed to penalize the model complexity [164], i.e.:

$$\mathrm{AIC} = \ln\left(J_n(\boldsymbol{\beta})\right) + 2\frac{m}{n}\,, \tag{A.5}$$

with:

- $\boldsymbol{\beta} \in \mathbb{R}^{m\times 1}$ is the parameters vector,
- n is the number of available data,
- $J_n(\boldsymbol{\beta})$ is the cost function of the employed method (for instance: the cost function of PEM (1.80)).

The employment of this criteria is basically a comparison between the AIC values computed on a pool of models which share the same model class, but have different model complexity. Thus, the smallest AIC value corresponds to the model with the best selectable model complexity.
Another metric is called Bayesian Information Criterion (BIC) [13] and it is defined as:

$$\mathrm{BIC} = \ln\left(J_n(\hat{\boldsymbol{\beta}})\right) + \ln(n)\frac{m}{n}\,. \tag{A.6}$$

The model complexity selection with BIC is identical to the procedure employed with AIC. This criteria is also known as Rissanen's Minimum Description Length (MDL) [165].
Both information criteria are evaluated by exploiting the train set, therefore the model selection procedure and the model parameters estimation share the same dataset. This aspect is the main advantage of these methods, since they works also with a small number of data. Instead, the main con is that these methods are not accurate if the out-of-sample data are too different from the training dataset. To overcome this phenomenon an estimation of E_{out} can be obtained by applying the so-called *Cross Validation (CV)* approach. It is applicable only if the dataset has a huge number of samples.

A.2 Cross-Validation (CV)

The Cross Validation (CV) is one of the most widespread method for estimating E_{out} of a learning model [14, 15, Chapter 7]. This technique splits the dataset into K parts, called *folds*. Iteratively, the model is trained on $K-1$ folds and the out-of-sample error is estimated on the held-out fold, i.e. the

fold not employed during the model training. The final estimation of E_{out} consists of averaging the K estimations. The model complexity selection is done by selecting the lowest value in a set of E_{out} obtained by performing P times the Cross Validation method, i.e. one for each model complexity [166].

CV is also applied in the machine learning literature to execute model performance evaluation and model selection. In [153], we have proposed a new CV method under *dataset shift*, i.e. a condition where the train set is acquired on different items, which have different statistical properties [167]. To solve this issue, we developed a variation of the CV, that we named *Object-wise Cross Validation (OCV)*. The main innovation is that the OCV composes the folds with data that comes from a specific item, see Figure 89, instead the traditional method makes the folds randomly from the train set, thus, under the dataset shift condition, the model performance evaluation and model selection are problematic. Formally, let $\delta : \{1, \ldots, n\} \rightarrow \{1, \ldots, K\}$ be an indexing function that indicates the item that generated the j-th observation, with K number of folds. Then, the error is computed as:

$$\mathrm{OCV}(\hat{f}, \boldsymbol{\beta}) = \frac{1}{n} \sum_{j=1}^{n} \ell \left(y_j, \hat{f}^{-\delta(j)} \left(\boldsymbol{x}_j, \boldsymbol{\beta} \right) \right) , \tag{A.7}$$

where:

- $\boldsymbol{x} \in \mathbb{R}^{n \times 1}$ corresponds to the feature vector;
- y is the result of the machine learning algorithm;
- $\ell(\cdot)$ is the loss function;
- $\hat{f}$ is the trained machine learning model endowed with $\boldsymbol{\beta} \in \mathbb{R}^{m \times 1}$ parameters.

An alternative of CV are the *Bootstrap methods*. An example of these is the "632+" estimator [168].

A.3 Regularization methods

Regularization is a methodology that differs from the information criterion and CV methods, because this does not perform a comparison between some metrics computed on a pool of model complexities. Instead, this method reduces the variance by introducing a small bias.

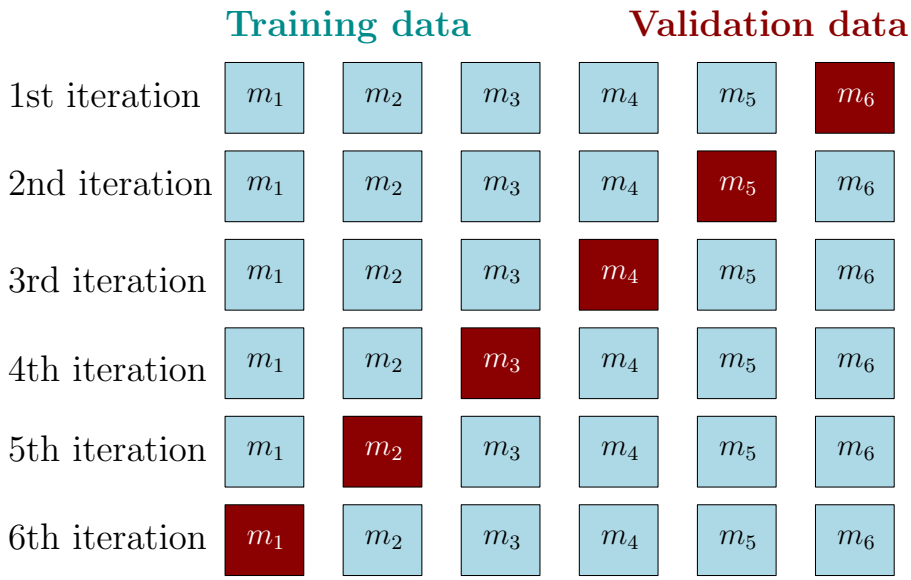


Figure 89: Example of fold extraction using the Object-wise Cross Validation with six items, denote with m_i, $i = 1, \ldots, 6$.

Consider a generic Finite Impulse Response (FIR) model:

$$\begin{aligned} y(t) &= G(z, \boldsymbol{\beta}) + e(t) \\ &= \textstyle\sum_{k=0}^{m-1} \beta_k u(t-k) + e(t) \quad , \\ \boldsymbol{y} &= \boldsymbol{\Phi}^\top \boldsymbol{\beta} + \boldsymbol{e} \end{aligned} \tag{A.8}$$

where:

- β_k are the impulse response coefficients. The parameters vector is denoted as $\boldsymbol{\beta} \in \mathbb{R}^{m \times 1}$;
- m is the order of the FIR model;
- n is the number of acquired data;
- $e(t)$ is the additive noise (assumed independent from $\boldsymbol{\Phi}$). The corresponding vector is denoted by $\boldsymbol{e} \in \mathbb{R}^{n \times 1}$;
- $y(t)$ is the output data. The corresponding vector is denoted as $\boldsymbol{y} \in \mathbb{R}^{n \times 1}$;
- $u(t)$ is the input data;
- $\boldsymbol{\Phi} \in \mathbb{R}^{m \times n}$ is the regression matrix[1];

[1]The regression matrix is the composition of n observed $u(t)$ stacked in m rows.

This can be cast as a linear regression problem. In particular, we assume to exploit the PEM method, with a FIR model:

$$\begin{aligned}\hat{\boldsymbol{\beta}} &= \arg\min_{\boldsymbol{\beta}} \mathbb{E}\left[\boldsymbol{y} - \boldsymbol{\Phi}^\top \boldsymbol{\beta}\right]^2 \\ &= \arg\min_{\boldsymbol{\beta}} \left\|\boldsymbol{y} - \boldsymbol{\Phi}^\top \boldsymbol{\beta}\right\|_2^2 .\end{aligned} \tag{A.9}$$

Regularization aims to consider the impact of $\boldsymbol{\beta}$ directly in the optimization problem by adding a term to the cost function. Thus, the result will be a solution which considers not only the mismatch between the measured and the estimated values, but also the penalization of $\boldsymbol{\beta}$. Therefore, the optimization problem becomes [169]:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \left(\left\|\boldsymbol{y} - \boldsymbol{\Phi}^\top \boldsymbol{\beta}\right\|_2^2 + \lambda l(\boldsymbol{\beta})\right) , \tag{A.10}$$

with: $\lambda \in \mathbb{R}^+$ is a *hyperparameter* that modulates the regularization impact and $l(\boldsymbol{\beta})$ represents the regularization term.

The most common regularization terms are: *Ridge regression* and *Lasso regression* [170]. The former considers the regularization term as a 2-norm squared, i.e.:

$$l(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_2^2 = \sum_{j=0}^{m-1} \beta_j^2 . \tag{A.11}$$

Instead,the latter corresponds to:

$$l(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_1 = \sum_{j=0}^{m-1} \left|\beta_j\right| . \tag{A.12}$$

Figure 90 depicts the graphical representation of both methods from a geometrical point of view with $m = 2$. The resulting regularized parameters are highlighted with the yellow star. These are the intersection between the cost function[2], represented by the red level curves , and the bounded region defined by the applied regularization term, highlighted in light blue. Ridge regression, depicted in the left image, allows to tighten $\boldsymbol{\beta}$ to low values, instead Lasso regression, illustrated in the right image, aims to set some parameters to zero. The center of the level curves, depicted as a red dot, is the estimation without regularization.

Another difference between the two regularization terms lies in the resolution because the Ridge regression has a closed form [14, Chapter 3], instead the Lasso regression is solved by quadratic programming [171].

[2]The cost function is supposed to be convex.

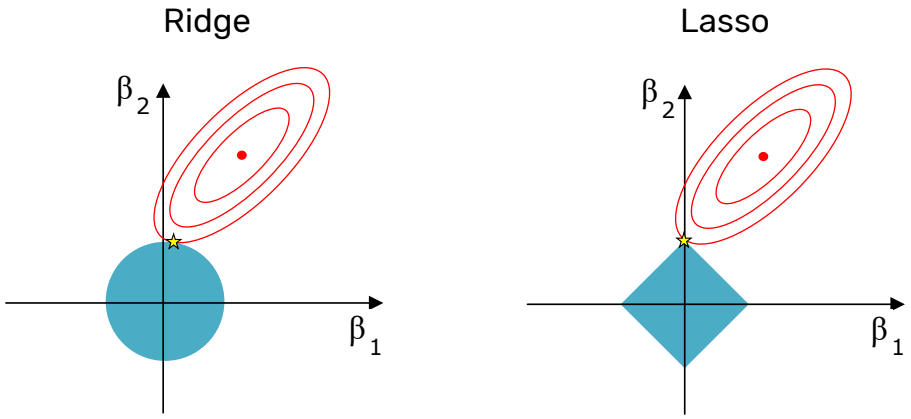


Figure 90: Geometric representation of Lasso and Ridge regression.

Note that the regularized optimization problem (A.10) can be cast into a constrained optimization problem, as:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \left\|\boldsymbol{y} - \boldsymbol{\Phi}^\top \boldsymbol{\beta}\right\|_2^2 \quad , \quad s.t. : l(\boldsymbol{\beta}) \leq C \tag{A.13}$$

where C is inverse proportional of λ.

Another typology of regularization is called *Tikhonov regularization* [172]. This is defined as:

$$l(\boldsymbol{\beta}) = \boldsymbol{\beta}^\top \boldsymbol{K}^{-1} \boldsymbol{\beta} \, . \tag{A.14}$$

Tikhonov regularization corresponds to a more general formulation of the Ridge regression. It is straightforwardly demonstrable by setting $\boldsymbol{K} \in \mathbb{R}^{m \times m}$, called *Kernel matrix*, equal to $\boldsymbol{I}_m$. The optimization problem (A.10) regularized with (A.14) is often referred as *Regularized Least Squares (ReLS)* [6]. ReLS is solvable by the following closed forms:

$$\begin{aligned} \hat{\boldsymbol{\beta}} &= \arg\min_{\boldsymbol{\beta}} \left(\left\|\boldsymbol{y} - \boldsymbol{\Phi}^\top \boldsymbol{\beta}\right\|_2^2 + \lambda \boldsymbol{\beta}^\top \boldsymbol{K}^{-1} \boldsymbol{\beta} \right) \\ &= \boldsymbol{K}\boldsymbol{\Phi} \left(\boldsymbol{\Phi}^\top \boldsymbol{K} \boldsymbol{\Phi} + \lambda \boldsymbol{I}_n \right)^{-1} \boldsymbol{y} \quad \text{or} \\ &= \left(\boldsymbol{K}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \lambda \boldsymbol{I}_m \right)^{-1} \boldsymbol{K}\boldsymbol{\Phi}\boldsymbol{y} \end{aligned} \tag{A.15}$$

The choice of $\boldsymbol{K}$ defines some priors of the problem. The priors are mainly the properties that $\boldsymbol{\beta}$ must have. Table 8 reports some types of Kernel matrices.

Since we considers a FIR model, the priors reflect the property of the impulse response coefficients β_k. For instance, it is possible to assume that:

Name	*Structure*	*Characteristics*
DC	$K_{ij}(\boldsymbol{\eta}) = \delta\alpha^{\frac{i+j}{2}}\varrho^{\lvert i-j\rvert}$	$\delta \geq 0, 0 \leq \alpha < 1, \lvert\varrho\rvert \leq 1, \boldsymbol{\eta} = [\delta, \alpha, \varrho]$
TC	$K_{ij}(\boldsymbol{\eta}) = \delta\alpha^{\max(i,j)}$	$\delta \geq 0, 0 \leq \alpha < 1, \boldsymbol{\eta} = [\delta, \alpha]$
SS	$K_{ij}(\boldsymbol{\eta}) = \delta\left(\frac{\alpha^{i+j+\max(i,j)}}{2} - \frac{\alpha^{3\max(i,j)}}{6}\right)$	$\delta \geq 0, 0 \leq \alpha < 1, \boldsymbol{\eta} = [\delta, \alpha]$

Table 8: Some types of Kernel matrix.

- The system is exponentially stable, therefore β_k should decay exponentially;
- The impulse response is smooth, hence the neighbouring values of β_k should have a positive correlation.

The *Diagonal-Correlated Kernel (DC)* [173] fits perfectly with these prior since α defines the exponential decay of the impulse response coefficients, instead ϱ describes the correlation between neighbouring impulse response coefficients [6].
Note that, the *Tuned/Correlated Kernel (TC)* (or *First-order Stable Spline*) is equal to DC Kernel by setting $\varrho = \sqrt{\alpha}$, therefore TC Kernel is a special case of DC Kernel [174, 175]. Therefore, also the TC Kernel can be used to model the prior of a FIR system.
The procedure to estimate the kernel matrix hyperparameters $\boldsymbol{\eta} \in \mathbb{R}^{n_\eta \times 1}$ (with $n_\eta \in \mathbb{N}^+$ is the number of hyperparameters) can be defined by exploiting the Bayesian interpretation of the Tikhonov regularization. This framework is applicable by assuming:

- $\boldsymbol{\beta}$ is a Gaussian random variable $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, with zero mean and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times m}$;
- $e(t)$ is Gaussian and independent from $\boldsymbol{\beta}$ with $\boldsymbol{e} \sim \mathcal{N}(\mathbf{0}, \sigma_e^2 \boldsymbol{I}_n)$;
- $\boldsymbol{e}$ and $\boldsymbol{\Phi}$ are known.

Thus, $\boldsymbol{y}$ and $\boldsymbol{\beta}$ will be jointly Gaussian variable [99, Chapter 2], therefore:

$$\begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{y} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\Sigma}\boldsymbol{\Phi} \\ \boldsymbol{\Phi}^\top\boldsymbol{\Sigma} & \boldsymbol{\Phi}^\top\boldsymbol{\Sigma}\boldsymbol{\Phi} + \sigma_e^2\boldsymbol{I}_n \end{bmatrix}\right). \tag{A.16}$$

Exploiting this property, the posterior distribution $\boldsymbol{\beta}|\boldsymbol{y} \sim \left(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\Sigma}}\right)$ can be computed by:

$$\hat{\boldsymbol{\beta}} = \boldsymbol{\Sigma}\boldsymbol{\Phi}\left(\boldsymbol{\Phi}^\top\boldsymbol{\Sigma}\boldsymbol{\Phi} + \sigma_e^2\boldsymbol{I}_n\right)^{-1}\boldsymbol{y} \tag{A.17a}$$

$$= \left(\boldsymbol{\Sigma}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma_e^2\boldsymbol{I}_m\right)^{-1}\boldsymbol{\Sigma}\boldsymbol{\Phi}\boldsymbol{y} \tag{A.17b}$$

$$\hat{\boldsymbol{\Sigma}} = \boldsymbol{\Sigma} - \boldsymbol{\Sigma}\boldsymbol{\Phi}\left(\boldsymbol{\Phi}^\top\boldsymbol{\Sigma}\boldsymbol{\Phi} + \sigma_e^2\boldsymbol{I}_n\right)^{-1}\boldsymbol{\Phi}^\top\boldsymbol{\Sigma} \tag{A.17c}$$

If we choose the regularization matrix as $\boldsymbol{K} = \boldsymbol{\Sigma}$ and the regularization term as $\lambda = \sigma_e^2$ the mean of the posterior distribution (Maximum A Posteriori (MAP) estimate) matches with the ReLS closed form (A.15).

Finally, since $\boldsymbol{y}$ is a Gaussian random vector, as denoted in (A.16), the hyperparameters $\boldsymbol{\eta}$ can be estimated by maximizing the *Marginalized Likelihood (MargLik)* (called also *Empirical Bayes*):

$$\hat{\boldsymbol{\eta}} = \arg\min_{\boldsymbol{\eta}} \left(\boldsymbol{y}^\top\boldsymbol{Z}(\boldsymbol{\eta})^{-1}\boldsymbol{y} + \log|\boldsymbol{Z}(\boldsymbol{\eta})|\right), \tag{A.18}$$

where $\boldsymbol{Z}(\boldsymbol{\eta}) = \boldsymbol{\Phi}^\top\boldsymbol{\Sigma}(\boldsymbol{\eta})\boldsymbol{\Phi} + \sigma_e^2\boldsymbol{I}_n$ is the covariance matrix of $\boldsymbol{y}$.

By solving optimization problem (A.18), some numerical errors can occur. This is due to the computation of the $\log|\boldsymbol{Z}\boldsymbol{\eta}|$ term. Since $\boldsymbol{Z}\boldsymbol{\eta}$ is positive definite, we can employ the Cholesky decomposition, see [120, Chapter 4]. This method decomposes $\boldsymbol{Z}\boldsymbol{\eta} \in \mathbb{R}^{n\times n}$ into $\boldsymbol{T}\boldsymbol{T}^\top = \boldsymbol{Z}\boldsymbol{\eta}$ such that: $\boldsymbol{T} \in \mathbb{R}^{n\times n}$ is lower triangular matrix with positive diagonal elements, called *Cholesky factor*. Doing so, it is possible to replace the $\log|\boldsymbol{Z}\boldsymbol{\eta}|$ term with $2\sum_{i=1}^{n}\log(T_{ii})$. Thus, the Empirical Bayes problem (A.18) becomes:

$$\hat{\boldsymbol{\eta}} = \arg\min_{\boldsymbol{\eta}} \left(\boldsymbol{y}^\top\hat{\boldsymbol{c}} + 2\sum_{i=1}^{n}\log(T_{ii})\right), \tag{A.19}$$

By performing the Cholesky decomposition we obtain the Cholesky factor, then we can compute $\hat{\boldsymbol{c}}$ by solving two triangular systems:

$$\begin{aligned}\boldsymbol{T}\boldsymbol{a} &= \boldsymbol{y}\\ \boldsymbol{T}^\top\hat{\boldsymbol{c}} &= \boldsymbol{a}\end{aligned} \tag{A.20}$$

The efficient computing of the MargLik estimation is explained in [176, 177].

In this description, the noise variance is assumed to be known, but usually it is unknown. Nevertheless, it can be estimated in multiple ways. As suggested by [173], if a FIR model is estimated using Least Squares the sample variance is equal to the estimation of σ_e^2. The authors of [7] have explained that it is also possible to do so with a low bias ARX. Another way is to consider σ_e^2 as an additional hyperparameter and estimating it by maximizing the MargLik (A.18), e.g. see [178, 179].

Appendix B

Functional analysis fundamentals

This appendix reports some definitions of functional analysis. This literature can be found in [180, 85].

Definition B.1: Vector Space

A **vector space** over the real field $\mathbb{R}$ is a set V endowed with two operations:

1. Sum: $V \times V \rightarrow V$;
2. Inner product: $\mathbb{R} \times V \rightarrow V$;

and that satisfies the following axioms:

- $u + (v + w) = (u + v) + w, \quad \forall u, v, w \in V$;
- $u + v = v + u$;
- $\exists 0 \in V : v + 0 = v, \quad \forall v \in V$;
- $\exists (-v) \in V : v + (-v) = 0, \quad \forall v \in V$;
- $\lambda_1(\lambda_2 v) = (\lambda_1 \lambda_2)v, \quad \forall \lambda_1, \lambda_2 \in \mathbb{R}, v \in V$;
- $\exists 1 \in V : 1 \cdot v = v, \quad \forall v \in V$;
- $\lambda_1(u + v) = \lambda_1 u + \lambda_1 v, \quad \forall \lambda_1 \in \mathbb{R}, u, v \in V$;
- $(\lambda_1 + \lambda_2)v = \lambda_1 v + \lambda_2 v, \quad \forall \lambda_1, \lambda_2 \in \mathbb{R}, v \in V$.

By supposing V and W are two vector spaces, it is possible to define the following definition:

Definition B.2: Linear operator, Linear functional

$F : V \rightarrow W$ is a **linear operator** if, $\forall \lambda_1, \lambda_2 \in \mathbb{R}$ one has that $F(\lambda_1 u_1 + \lambda_2 u_2) = \lambda_1 F(u_1) + \lambda_2 F(u_2), \quad \forall u_1, u_2 \in V$. If W is $\mathbb{R}$ the linear operator is called **linear functional**.

Definition B.3: Bilinear form, Symmetric

A **bilinear form** is a function $a : V \times V \rightarrow \mathbb{R}$ that is a linear in both arguments, that is: $a(u, \cdot) : V \rightarrow \mathbb{R}$ and $a(\cdot, v) : V \rightarrow \mathbb{R}$ are linear functionals where the former has fixed u and $\forall u \in V$ and the latter has fixed v and $\forall v \in V$. If the bilinear form a is $a(u, v) = a(v, u), \forall u, v \in V$, then it is said **symmetric**.

Definition B.4: Norm, Normed space

By letting a vector space V, the **norm** over V is a function $\|\cdot\| : V \to \mathbb{R}$ that holds:

- $\|u\| \geq 0; \quad \|u\| = 0 \iff u = 0 \quad \forall u \in V;$
- $\|\lambda_1 u\| = |\lambda_1| \cdot \|u\|, \quad \forall u \in V, \forall \lambda_1 \in \mathbb{R};$
- $\|u + v\| \leq \|u\| + \|v\|, \quad \forall u, v \in V.$

In this case $(V, \|\cdot\|)$ is called **normed space**.

Definition B.5: Convergent, Cauchy sequence

A sequence u_n is:

- **Convergent**: if $\exists u \in V$ s.t. $u_n \to u$, that is $\lim_{n\to\infty} \|u_n - u\| = 0$;
- **Cauchy sequence**: if $\|u_n - u_m\| < \epsilon$ with $\epsilon \in \mathbb{R}^+$ and when $n, m \to \infty$;

with the normed space $(V, \|\cdot\|)$. Observe that: every convergent sequence is a Cauchy sequence in a normed space.

Definition B.6: Complete, Banach space

A normed space is said to be **complete** if every Cauchy sequence converges (in the same space).
A **Banach space** is a complete normed space.

Definition B.7: Pre-Hilbert space, Induced norm, Hilbert space

A **Pre-Hilbert space** is a vector space V endowed with a inner product of a bilinear symmetric form $\langle \cdot, \cdot \rangle : V \times V \to \mathbb{R}$, subject to: $\langle u, u \rangle > 0 \quad \forall u \neq 0$.
The **induced norm** is defined as $\|u\| = \sqrt{\langle u, u \rangle}$. Furthermore, the following propositions hold:

- Schwarz inequality: $|\langle u, v \rangle| \leq \|u\| \cdot \|v\| \quad \forall u, v \in V;$
- The induced norm is effectively a norm.

An **Hilbert space** is a complete pre-Hilbert space.

Lists of Notations

Important sets

- $\mathbb{N}$ is the set of all natural numbers;
- $\mathbb{Z}$ is the set of all integer numbers;
- $\mathbb{Z}^+$ is the set of all strictly-positive integer numbers;
- $\mathbb{Q}$ is the set of all rational numbers;
- $\mathbb{Q}^+$ is the set of all strictly-positive rational numbers;
- $\mathbb{R}$ is the set of all real numbers;
- $\mathbb{R}^+$ is the set of all strictly-positive real numbers;
- $\mathbb{C}$ is the set of all complex numbers;
- $\mathbb{C}^+$ is the set of all complex numbers in the right half plane;
- Other sets $\mathcal{X}$ are denoted by calligraphic letters;
- $\mathcal{H}_\infty = \mathcal{H}_\infty(\mathbb{C}^+)$ is the Hardy space of scalar of matrix valued functions which are bounded in $\mathbb{C}^+$;
- $\mathcal{RH}_\infty$ is the subspace of $\mathcal{H}_\infty$ consisting of real-rational scalar or matrix valued functions.

Vectors and Matrices

Let $n; m \in \mathbb{N}\backslash\{0\}$

- Generic scalars are indicated with a lower-case letter, e.g. a;
- Generic vectors are indicated with a lower-case bold letter, e.g. $\boldsymbol{x}$;
- x_i denotes the i-th element of the vector $\boldsymbol{x}$;
- Generic matrices are indicated with a upper-case bold letter, e.g. $\boldsymbol{A}$;
- a_{ij} denotes the element of the matrix $\boldsymbol{A}$, with i-th row and j-th column;
- The set of all real matrices with n rows and m columns: $\mathbb{R}^{n\times m}$;

- Identity matrix with n rows: $\boldsymbol{I}_n \in \mathbb{R}^{n \times n}$;
- Matrix with suitable rows and columns, in which all elements are equal to 0, is denoted as $\mathbf{0}$;
- The transpose of a matrix $\boldsymbol{A}$: $\boldsymbol{A}^\top$;
- The conjugate transpose (or Hermitian transpose) of a matrix $\boldsymbol{A}$: $\boldsymbol{A}^*$;
- The inverse of an invertible square matrix $\boldsymbol{A}$: $\boldsymbol{A}^{-1}$;
- The determinant of a square matrix $\boldsymbol{A}$: $\det(\boldsymbol{A})$;
- The trace of a square matrix $\boldsymbol{A}$: $\mathrm{Tr}(\boldsymbol{A})$;
- The rank of a matrix $\boldsymbol{A}$: $\mathrm{rank}(\boldsymbol{A})$.

Dynamical system theory

- The transfer functions are denoted with an upper-case letter, e.g. $G(\cdot)$;
- The Laplace variable is indicated with $s \in \mathbb{C}$;
- The Z-transform variable is indicated with $z \in \mathbb{C}$;
- The magnitude of a transfer function is indicated as $|\cdot|$;
- The phase of a transfer function is indicated as $\angle(\cdot)$;
- A continuos-time dynamic system transfer function is represented by $G(s)$;
- A discrete-time dynamic system transfer function is represented by $G(z)$
- The Laplace transform of $x(t)$ is indicated with $X(s) = \mathcal{L}[x(t)]$;
- The Laplace anti-transform of $X(s)$ is indicated with $x(t) = \mathcal{L}^{-1}[X(s)]$;
- The Z-transform of $x(t)$ is indicated with $X(z) = \mathcal{Z}[x(t)]$;
- The inverse of Z-transform of $X(z)$ is indicated with $x(t) = \mathcal{Z}^{-1}[X(z)]$;
- The convolution of two functions is indicated with $[a * b](t)$;
- The frequency response of a continuos-time dynamic system is indicated with $G(j\omega)$ at frequency $\omega \in \mathbb{R}$;

- The frequency response of a discrete-time dynamic system is indicated with $G(e^{j\omega})$ at frequency $\omega \in [0, \pi f_s]$.

Throughout the book, with a little abuse of notation, the equations that ties the input with output of a discrete dynamic system are denoted, for instance, as: $y(t) = G(z)u(t)$.

Norms

The norms properties are the following:

1. $\|u\| \geq 0$;
2. $\|u\| = 0 \Leftrightarrow u(t) = 0 \quad \forall$t;
3. $\|au\| = |a| \, \|u\| \quad \forall a \in \mathbb{R}$;
4. $\|u + v\| \leq \|u\| + \|v\|$.

Norms for signals

Consider a signals $u(t)$, where t is the temporal instant. They are assumed to be piecewise continuos. The norms for these signals are:

- 1-Norm: $\|u\|_1 = \int_{-\infty}^{\infty} |u(t)| \, dt$;
- 2-Norm: $\|u\|_2 = \left(\int_{-\infty}^{\infty} |u(t)|^2 \, dt \right)^{\frac{1}{2}}$;
- ∞-Norm: $\|u\|_\infty = \sup_t |u(t)|$.

Norms for vector

Given a vector $\boldsymbol{x}$ of length n the most important norms are:

- 1-Norm: $\|\boldsymbol{x}\|_1 = |x_1| + \ldots + |x_n|$;
- 2-Norm: $\|\boldsymbol{x}\|_2 = \left(|x_1|^2 + \ldots + |x_n|^2 \right)^{\frac{1}{2}}$. It can be seen as the Frobenius norm (in the vector case);
- ∞-Norm: $\|\boldsymbol{x}\|_\infty = \max_{1 \leq i \leq n} |x_i|$.

Figure 91 depicts the norms shape of a two dimensional vector $\boldsymbol{x} \in \mathbb{R}^{2 \times 1}$.

Norms for Systems

Consider a system $G(s)$ that is Linear Time-Invariant (LTI) and Single input Single output (SISO). Two norms of $G(s)$ are:

- 2-Norm: $\|G(s)\|_2 = \left(\frac{1}{2\pi}\int_{-\infty}^{\infty} |G(j\omega)|^2 \, d\omega\right)^{\frac{1}{2}}$;
- ∞-Norm: $\|G(s)\|_\infty = \max_\omega \; |G(j\omega)|$.

Now, let a Linear Time-Invariant (LTI) Multiple input Multiple output (MIMO) system $\boldsymbol{G}(s)$. Two norms of $\boldsymbol{G}(s)$ are:

- 2-Norm: $\|\boldsymbol{G}(s)\|_2 = \sqrt{\frac{1}{\sqrt{2}}\int_0^{\infty} \|g(t)\|_F^2 \, dt}$, where $g(t)$ is the impulse response of the system and is the Frobenius norm, defined as $\|\boldsymbol{A}\|_F = \sqrt{\mathrm{Tr}(\boldsymbol{A}\cdot\boldsymbol{A})}$ with a generic matrix $\boldsymbol{A}$;
- ∞-Norm: $\|\boldsymbol{G}(s)\|_\infty = \max_\omega \bar{\sigma}\left(\boldsymbol{G}(j\omega)\right)$, where $\bar{\sigma}(\cdot)$, denotes the maximum singular value.

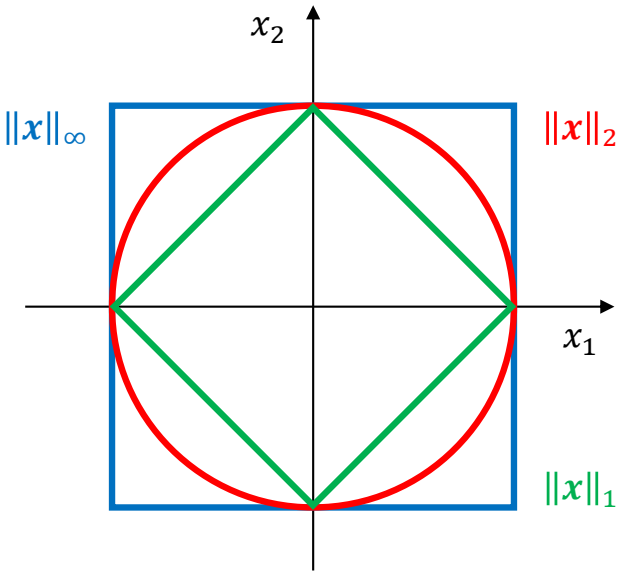


Figure 91: Contours of $\|x\|_p$, where $p = 1, 2, \infty$ and $x = [x_1, x_2]$.

Glossary of the most repeated symbols

Generic symbols

Symbol	*Description*
t	time index
ω	frequency index
ω_k	sampled frequencies
$\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, D$	state space matrices of a dynamical system
$\boldsymbol{x}(t)$	states
$\hat{\boldsymbol{x}}(t)$	estimated states
$K(s)$	controller
$L_0(s)$	nominal loop function
$S_0(s)$	nominal sensitivity function
$T_0(s)$	nominal complementary sensitivity function
$Q_0(s)$	nominal control sensitivity function
$u(t)$	control input signal
$y(t)$	sensed output signal
$v(t)$	additive noise signal
$\hat{y}(t)$	estimated output signal
$e(t)$	white noise signal
$z(t)$	generic output
$w(t)$	generic input
$G(z)$	uncertain system
$G_T(z)$	true system
$G_0(s)$	nominal model
$\hat{G}_0(s)$	nominal model identified with black-box identification
σ_e^2	variance of $e(t)$
f_s	frequency sampling
T_s	time sampling

Uncertainty in system modeling

Symbol	*Description*
$\boldsymbol{q}(t)$	uncertain parameters vector
l	number of uncertain parameters
n_z, n_w, n_x	respectively number of $z(t)$, $w(t)$, $\boldsymbol{x}(t)$ signals
$\boldsymbol{M}(s)$	transfer function matrix that represents how the uncertainty affects the system
$\boldsymbol{\Delta}$	parametric uncertainty
$\boldsymbol{w}_\Delta(t), \boldsymbol{z}_\Delta(t)$	signals that connects $\boldsymbol{M}(s)$ with $\boldsymbol{\Delta}$
$n_{w_\Delta}, n_{z_\Delta}$	number of signals of respectively $\boldsymbol{w}_\Delta(t), z_\Delta(t)$
$\mathcal{F}_u(\boldsymbol{M}(s), \boldsymbol{\Delta})$	upper Linear Fractional Transformation
$\boldsymbol{\Delta}(s)$	parametric uncertainty with mixed linear and nonlinear uncertainties
$\Delta(s)$	nonparametric uncertainty
$W_a(s)$	weight transfer function for additive uncertainty
$W_i(s)$	weight transfer function for multiplicative input uncertainty
$W_o(s)$	weight transfer function for multiplicative output uncertainty
$D_{lfc}(s), N_{lfc}(s)$	left coprime factorization terms of $G_0(s)$
$\Delta_d(s), \Delta_n(s)$	uncertainty in left coprime factorization
$W_d(s), W_n(s)$	weights transfer function for left coprime factorization uncertainty

Uncertainty for robust control

Symbol	*Description*
$\mathcal{A}$	structured set with fixed frequency ω
$\mathcal{B}_\mathcal{A}$	norm bounded set
$\mu_\mathcal{A}(\boldsymbol{M}(j\omega))$	structured singular value of $\boldsymbol{M}(j\omega)$ with respect to $\mathcal{A}$
$\boldsymbol{P}(s)$	open loop system that represents the known part of the system
$\boldsymbol{N}(s)$	transfer function that represents the group composed of $\boldsymbol{P}(s)$ with $\Delta(s)$

$\mathcal{F}_l(\boldsymbol{N}(s), K(s))$	lower Linear Fractional Transformation
$W_S(s)$	weight function for $S_0(s)$
$W_T(s)$	weight function for $T_0(s)$
$W_Q(s)$	weight function for $Q_0(s)$
$\boldsymbol{H}(s)$	transfer function from $\boldsymbol{z}_\Delta$ to w_Δ with left coprime factorization uncertainty
$D(s)$	scaling transfer function for μ synthesis

Uncertainty for robust fault diagnosis

Symbol	***Description***
$r(t)$	residual signal
$\theta(t)$	processed residual signal
$d(t)$	disturbance signal
$f(t)$	fault signal
$\iota(t)$	diagnostic decision signal
$\hat{N}(z), \hat{M}(z)$	left coprime factorization terms
$N(z), M(z)$	right coprime factorization terms
$G_d(z)$	transfer function from disturbance to output
$G_v(z)$	transfer function from noise to output
$G_f(z)$	transfer function from fault to output
$Q(z)$	post-filter
$\hat{Q}(z)$	estimated post-filter
$\tilde{\boldsymbol{Q}}(z)$	post-filter designed with the algorithm proposed in [27]
γ	peak gain of $\hat{Q}(z)\hat{M}(z)\boldsymbol{G}_v(z)$
$\boldsymbol{L}$	gain matrix of a full-order observer
V	weight matrix of a full-order observer
$b+1$	length of parity vector
$\boldsymbol{v}_b, \boldsymbol{y}_b, \boldsymbol{u}_b, \boldsymbol{f}_b$	signals vector of length b
$\boldsymbol{V}_b$	parity vector
$\boldsymbol{H}_{o,b}, \boldsymbol{H}_{u,b}, \boldsymbol{H}_{f,b}, \boldsymbol{H}_{d,b}$	matrices composed by the state space matrices
P_b	parity space
τ	threshold

System identification

Symbol	*Description*
$\mathcal{M}$	model family
$\boldsymbol{\beta}$	parameters vector
$\boldsymbol{\beta}_T$	true parameters vector
$\hat{\boldsymbol{\beta}}$	estimated parameters vector
m	number of parameters
$H(z, \boldsymbol{\beta})$	transfer function from $e(t)$ to $v(t)$
$G(z, \boldsymbol{\beta})$	transfer function from $u(t)$ to $y(t)$
$\mathcal{D}$	dataset
$J_n(\boldsymbol{\beta})$	cost function with finite length dataset $\mathcal{D}$
n	number of data
E_{out}	out of sample error
E_{in}	in sample error
Γ	infinite-dimensional function space
g	non-parametric function
$\hat{g}_0$	identified non-parametric function
$\boldsymbol{\Phi}$	regression matrix
$J(g)$	regularization term for non-parametric function
$l(\boldsymbol{\beta})$	regularization term for parameters
λ	regularization strength term
x	regressors
$\mathcal{K}_x$	reproducing kernel
$\boldsymbol{K}$	kernel matrix
$\mathcal{K}(\cdot)$	kernel section
$\boldsymbol{c}$	weight vector of the Representer theorem
$\hat{\boldsymbol{c}}$	identified weight vector
$\boldsymbol{\eta}$	hyperparameters vector
$\bar{\boldsymbol{P}}_\beta$	covariance matrix of the parameters vector estimated with Prediction Error Method
$\bar{\boldsymbol{R}}_\beta$	asymptotic autocorrelation of the parameters vector

$\hat{\boldsymbol{R}}_{\beta}$	estimated asymptotic autocorrelation of the parameters vector
$\mathcal{S}_T$	true model family
α	size of the confident region
p_{β}	probability of the confidence interval
$\boldsymbol{\Lambda}_G^{\top}\left(e^{j\omega}\right)$	partial derivative of $G(e^{j\omega}, \boldsymbol{\beta})$
$\boldsymbol{\Sigma}$	covariance matrix of kernel based identification
$\mathcal{B}_e^2, \mathcal{B}_e^{\infty}$	uncertainty set in the measurement space of respectively 2-norm bound, ∞-norm bound
$\Delta_G(s)$	true model error model
$\hat{\hat{G}}(j\omega_k)$	nominal model plus model error model plus noise term
$\boldsymbol{O}$	basis function vector
Λ	random walk process over ω
$\bar{\boldsymbol{\beta}}$	a priori known parameters
$\epsilon(s)$	residual signals of model error
$G_e(s)$	identified model error model
$\boldsymbol{\beta}\|\boldsymbol{y}$	posterior distribution

Mixed-sensitivity loop-shaping

Symbol	***Description***
$J(\boldsymbol{\rho}, G_T)$	cost function of mixed-sensitivity loop-shaping
$\boldsymbol{H}(z, \boldsymbol{\rho})$	transfer function from $\boldsymbol{z}_{\Delta}$ to w_{Δ}
$\boldsymbol{\rho}$	controller parameters
$S(z, \boldsymbol{\rho})$	true sensitivity function
$T(z, \boldsymbol{\rho})$	true complementary sensitivity function
$G_p(z)$	random sample form the posterior distribution
$\Omega(e^{j\omega})$	nonparametric sample of $W_T(z)$
$r(t)$	reference signal
ζ	confidence level
ε	accuracy level
n_p	number of sampled systems $G_p(z)$
ϵ	damping factor

o	overshoot
l	settling time
ω_c	critical frequency
$\omega_{c,\hat{W}_T}$	critical frequency of $\hat{W}_T$
$(\mathcal{O}, \mathcal{L})$	two sets of Pareto-optimal dominant solutions for the overshoot and settling time
l_{max}, l_{min}	maximum and minimum of the allowable settling time
o_{min}	minimum of the allowable overshoot
$\omega_{c\_\min}$	minimum of the allowable critical frequency
$h_{g\_\max}, h_{g\_\min}$	maximum and minimum of the high frequency gain
l_g, h_g	respectively low and high frequency gain
$\mathcal{W}$	discrete frequency grid
n_m	number of samples in $\mathcal{W}$
γ	feasibility margin
$(\mathcal{Z}, \mathcal{Q})$	respectively critical frequency and high-frequency gain sets of Pareto-optimal dominant solutions
$\mathcal{G}$	Output Error model set
$\hat{y}_{step}(t)$	closed loop unit step responses
$G_{BU}(z), G_{BL}(z)$	upper and lower boundary conditions
$\hat{G}_{BU}(z), \hat{G}_{BL}(z)$	estimation of upper and lower boundary conditions
$\mathcal{D}_c$	dataset composed of both boundary conditions
$\mathcal{D}_u, \mathcal{D}_l$	respectively dataset of upper and lower conditions
b	number samples of transient sector
$y_{BU}(t), y_{BL}(t)$	output signals of upper and lower conditions
$\hat{G}_{0g}(z)$	average dynamic system
$\hat{T}_{0g}(z)$	closed loop of the average dynamic system
$\boldsymbol{\beta}_{BU}\|\boldsymbol{y}_{BU}, \boldsymbol{\beta}_{BL}\|\boldsymbol{y}_{BL}$	posterior distribution of upper and lower boundary condition

List of Acronyms

AFDP Approximate Fault Detection Problem. 3, 39, 40, 45, 113–115, 120, 164

AIC Akaike Information Criterion. 2, 51, 93, 168, 169

AR AutoRegressive. 48, 49

ARMAX AutoRegressive Moving Average with an eXogenous variable. 48, 49

ARX AutoRegressive with an eXogenous variable. 48, 49, 58, 175

BIC Bayesian Information Criterion. 2, 51, 168, 169

BJ Box-Jenkins. 49

CV Cross Validation. 2, 51, 168–170

DC Diagonal-Correlated Kernel. 174

DCM Direct Current Motor. 150–152, 155, 160

EFDP Exact Fault Detection Problem. 39, 115

EMA Electro-Mechanical Actuator. 4, 5, 149–152, 154, 159, 160, 199

EUS Estimated Uncertainty Set. 63, 64

FD Fault Detection. 136

FFT Fast Fourier Transform. 159, 160

FIR Finite Impulse Response. 49, 55, 77, 93, 110, 117, 127, 171–175

FMECA Failures Mode, Effects and Criticality Analysis. 4, 5, 149, 150, 160, 164

FPS Feasible Parameter Set. 64, 65, 69

GP Gaussian Processes. 76

HHT Hilbert-Huang Transform. 34

HVAC Heat, ventilation and air conditioning system. 123, 124

IAE Integral Absolute Error. 97, 98, 100, 102, 129, 132, 195, 196

IQC Integral Quadratic Constraint. 13

LFC Left Coprime Factorization. 36

LFT Linear Fractional Transformation. 10, 11, 13, 14, 18, 20, 24–26, 28, 70, 83, 87, 107, 113, 114, 184, 185, 193

LTI Linear Time-Invariant. 19, 47, 77, 78, 105, 111, 114, 182, 193

MA Moving Average. 48, 49

MAP Maximum A Posteriori. 55, 175

MargLik Marginalized Likelihood. 55, 175

MCSA Motor-Current Signature Analysis. 33

MDL Rissanen's Minimum Description Length. 169

MEM Model Error Modeling. 2, 66, 67, 69, 164

MIMO Multiple input Multiple output. 7, 15, 18, 24, 29, 35, 124, 164, 182

ML Maximum Likelihood. 51

MPC Model Predictive Control. 123, 124

MUS Measurement Uncertainty Set. 63, 64

OCV Object-wise Cross Validation. 170, 171, 199

OE Output Error. 49, 93, 188

PEM Prediction Error Method. 2, 4, 7, 46–48, 51, 56–58, 60, 62, 63, 67, 68, 75, 93–104, 163, 169, 172, 186, 195, 196

PET PolyEthylene Terephthalate. 4, 5, 135

PWM Pulse-Width Modulation. 125, 127, 152

ReLS Regularized Least Squares. 55, 173, 175

RFC Right Coprime Factorization. 36

RKHS Reproducing Kernel Hilbert Space. 46, 51–55

RMS Root Mean Square. 33, 46, 143, 145, 146, 199, 201

SE Stochastic Embedding. 2, 4, 66, 69

SISO Single input Single output. 3, 7, 9, 14, 15, 22, 25, 35, 47, 77, 78, 80, 81, 87, 105, 110, 111, 114, 182, 193

SK Spectral Kurtosis. 142

SS Stable-spline Kernel. 174

STFT Short-Time Fourier Transform. 34

SVM Support Vector Machines. 159

TC Tuned/Correlated Kernel. 174

TSA Time Synchronous Averaging. 149

WT Wavelet Transform. 34

WVD Wigner-Ville Distribution. 34

List of Figures

List of Tables

List of Algorithms

References

[1] L. Ljung, *Model validation and model error modeling*. Linköping University Electronic Press, 1999.

[2] P. Stoica and T. Söderström, *System Identification*. Prentice Hall international series in systems and control engineering, 1989.

[3] L. Ljung, "Prediction error estimation methods," *Circuits, Systems and Signal Processing*, 2002.

[4] F. C. Schweppe, *Uncertain dynamics systems*. Practice Hall, 1973.

[5] M. Milanese and A. Vicino, "Optimal estimation theory for dynamic systems with set membership uncertainty: An overview," *Automatica*, vol. 27, no. 6, pp. 997–1009, 1991.

[6] G. Pillonetto, F. Dinuzzo, T. Chen, G. De Nicolao, and L. Ljung, "Kernel methods in system identification, machine learning and function estimation: A survey," *Automatica*, 2014.

[7] G. Pillonetto and G. De Nicolao, "A new kernel-based approach for linear system identification," *Automatica*, vol. 46, no. 1, pp. 81–93, 2010.

[8] Y. Fujimoto, I. Maruta, and T. Sugie, "Input design for kernel-based system identification from the viewpoint of frequency response," *IEEE Transactions on Automatic Control*, vol. 63, no. 9, pp. 3075–3082, 2018.

[9] G. C. Goodwin, M. Gevers, and B. Ninness, "Identification and robust control: Bridging the gap," in *Proceeding of the 7th IEEE Mediterranian Conference on Control and Automation*, 1999.

[10] W. Reinelt, A. Garulli, and L. Ljung, "Comparing different approaches to model error modeling in robust identification," *Automatica*, vol. 38, no. 5, pp. 787–803, 2002.

[11] B. Wahlberg and L. Ljung, "Hard frequency-domain model error bounds from least-squares like identification techniques," *IEEE Transactions on Automatic Control*, vol. 37, no. 7, pp. 900–912, 1992.

[12] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Selected papers of hirotugu akaike*, pp. 199–213, Springer, 1998.

[13] G. Schwarz, "Estimating the dimension of a model," *The annals of statistics*, pp. 461–464, 1978.

[14] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*, vol. 2. Springer, 2009.

[15] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *Journal of the royal statistical society: Series B (Methodological)*, vol. 36, no. 2, pp. 111–133, 1974.

[16] S. Skogestad and I. Postlethwaite, *Multivariable feedback control: analysis and design*, vol. 2. Citeseer, 2007.

[17] D. Youla and J. Bongiorno, "A feedback theory of two-degree-of-freedom optimal wiener-hopf design," *IEEE Transactions on Automatic Control*, vol. 30, no. 7, pp. 652–665, 1985.

[18] D. McFarlane and K. Glover, "A loop-shaping design procedure using $\mathcal{H}_{\text{inf}}$ synthesis," *IEEE transactions on automatic control*, vol. 37, no. 6, pp. 759–769, 1992.

[19] H. Kwakernaak, "Robust control and $\mathcal{H}_\infty$-optimization—tutorial paper," *automatica*, vol. 29, no. 2, pp. 255–273, 1993.

[20] J. Chen and R. J. Patton, *Robust model-based fault diagnosis for dynamic systems*, vol. 3. Springer Science & Business Media, 2012.

[21] R. J. Patton and J. Chen, "A review of parity space approaches to fault diagnosis," *IFAC Proceedings Volumes*, vol. 24, no. 6, pp. 65–81, 1991.

[22] R. J. Patton and J. Chen, "Robust fault detection using eigenstructure assignment: A tutorial consideration and some new results," in *Proceedings of the 30th IEEE Conference on Decision and Control*, vol. 3, pp. 2242–2247, 1991.

[23] R. J. Patton, "Robust model-based fault diagnosis: the state of the art," *IFAC Proceedings Volumes*, vol. 27, no. 5, pp. 1–24, 1994.

[24] J. Doyle, B. Frances, A. Tannenbaum, and K. Moore, *Feedback control theory*, vol. 39. IEEE, 1994.

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

[26] K. Zhou, J. Doyle, and K. Glover, *Robust and optimal control*. NJ: Prentice Hall, 1996.

[27] A. Varga, "Solving fault diagnosis problems," *Studies in Systems, Decision and Control, 1st ed.*, 2017.

[28] S. X. Ding, *Model-based fault diagnosis techniques: design schemes, algorithms, and tools, 2nd ed.* Springer Science & Business Media, 2013.

[29] M. Mazzoleni, G. Di Rito, and F. Previdi, *Electro-Mechanical Actuators for the More Electric Aircraft*. Springer International, 2021.

[30] F. Blanchini and P. Colaneri, "Uncertain systems: Time-varying versus time-invariant uncertainties," in *Uncertainty in Complex Networked Systems*, pp. 3–91, Springer, 2018.

[31] B. R. Barmish and E. Jury, "New tools for robustness of linear systems," *IEEE Transactions on Automatic Control*, pp. 2525–2525, 1994.

[32] R. Tempo, G. Calafiore, and F. Dabbene, *Randomized algorithms for analysis and control of uncertain systems: with applications*. Springer, 2013.

[33] A. Megretski and A. Rantzer, "System analysis via integral quadratic constraints," *IEEE Transactions on Automatic Control*, vol. 42, no. 6, pp. 819–830, 1997.

[34] I. R. Petersen and R. Tempo, "Robust control of uncertain systems: Classical results and recent developments," *Automatica*, vol. 50, no. 5, pp. 1315–1335, 2014.

[35] I. R. Petersen, "A stabilization algorithm for a class of uncertain linear systems," *Systems & control letters*, vol. 8, no. 4, pp. 351–357, 1987.

[36] K.-Z. Liu, M. Ono, X. Li, and M. Wu, "Robust performance synthesis for systems with positive-real uncertainty and an extension to the negative-imaginary case," *Automatica*, vol. 82, pp. 194–201, 2017.

[37] G. Zames, "On the input-output stability of time-varying nonlinear feedback systems part one: Conditions derived using concepts of loop gain, conicity, and positivity," *IEEE transactions on automatic control*, vol. 11, no. 2, pp. 228–238, 1966.

[38] J. C. Doyle, J. E. Wall, and G. Stein, "Performance and robustness analysis for structured uncertainty," in *Proceeding of the 21st IEEE conference on decision and control*, pp. 629–636, 1982.

[39] G. E. Dullerud and F. Paganini, *A course in robust control theory: a convex approach*, vol. 36. Springer Science & Business Media, 2013.

[40] K. Glover and J. C. Doyle, "A state space approach to $\mathcal{H}_\infty$ optimal control," *Three decades of mathematical system theory*, pp. 179–218, 1989.

[41] J. C. Doyle, "A review of μ for case studies in robust control," *IFAC Proceedings Volumes*, vol. 20, no. 5, pp. 365–372, 1987.

[42] P. Apkarian and D. Noll, "The $\mathcal{H}_\infty$ control problem is solved," *Aerospace Lab*, vol. -, no. 13, pp. pages–1, 2017.

[43] P. Dorato and A. Kestenbaum, "Application of game theory to the sensitivity design of optimal systems," *IEEE Transactions on Automatic Control*, vol. 12, no. 1, pp. 85–87, 1967.

[44] S. Chang and T. Peng, "Adaptive guaranteed cost control of systems with uncertain parameters," *IEEE Transactions on Automatic Control*, vol. 17, no. 4, pp. 474–483, 1972.

[45] D. Carlucci and F. Donati, "Control of norm uncertain systems," *IEEE Transactions on Automatic Control*, vol. 20, no. 6, pp. 792–795, 1975.

[46] I. Horowitz, "Quantitative feedback theory," in *Proceeding of the IEEE (Control Theory and Applications)*, vol. 129, pp. 215–226, 1982.

[47] P. Apkarian and D. Noll, "Nonsmooth $\mathcal{H}_\infty$ synthesis," *IEEE Transactions on Automatic Control*, vol. 51, no. 1, pp. 71–86, 2006.

[48] R. Isermann, *Fault-Diagnosis Systems - An Introduction from Fault Detection to Fault Tolerance*. Springer-Verlag Berlin Heidelberg, 2006.

[49] D. Miljković, "Fault detection methods: A literature survey," in *Proceedings of the 34th international convention MIPRO*, pp. 750–755, IEEE, 2011.

[50] S. J. Qin, "Survey on data-driven industrial process monitoring and diagnosis," *Annual reviews in control*, vol. 36, no. 2, pp. 220–234, 2012.

[51] N. M. Nor, C. R. C. Hassan, and M. A. Hussain, "A review of data-driven fault detection and diagnosis methods: Applications in chemical process systems," *Reviews in Chemical Engineering*, vol. 36, no. 4, pp. 513–553, 2020.

[52] R. Isermann and P. Balle, "Trends in the application of model-based fault detection and diagnosis of technical processes," *Control engineering practice*, vol. 5, no. 5, pp. 709–719, 1997.

[53] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki, and J. Schröder, *Diagnosis and fault-tolerant control*, vol. 2. Springer, 2006.

[54] Y. C. Yeh, "Triple-triple redundant 777 primary flight computer," in *Proceeding of the IEEE Aerospace Applications Conference*, vol. 1, pp. 293–307, IEEE, 1996.

[55] H. Hashemian, "Maintenance of process instrumentation in nuclear power plant," *Berlin Heidelberg*, 2006.

[56] E. Chow and A. Willsky, "Analytical redundancy and the design of robust failure detection systems," *IEEE Transactions on automatic control*, vol. 29, no. 7, pp. 603–614, 1984.

[57] Z. Gao, C. Cecati, and S. X. Ding, "A survey of fault diagnosis and fault-tolerant techniques—part i: Fault diagnosis with model-based and signal-based approaches," *IEEE transactions on industrial electronics*, vol. 62, no. 6, pp. 3757–3767, 2015.

[58] L. Hong and J. S. Dhupia, "A time domain approach to diagnose gearbox fault based on measured vibration signals," *Journal of Sound and Vibration*, vol. 333, no. 7, pp. 2164–2180, 2014.

[59] H. Chen and S. Lu, "Fault diagnosis digital method for power transistors in power converters of switched reluctance motors," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 2, pp. 749–763, 2012.

[60] N. M. Freire, J. O. Estima, and A. J. M. Cardoso, "Open-circuit fault diagnosis in pmsg drives for wind turbine applications," *IEEE Transactions on Industrial electronics*, vol. 60, no. 9, pp. 3957–3967, 2012.

[61] S. Nandi, H. A. Toliyat, and X. Li, "Condition monitoring and fault diagnosis of electrical motors—a review," *IEEE transactions on energy conversion*, vol. 20, no. 4, pp. 719–729, 2005.

[62] R. B. Randall, *Vibration-based condition monitoring: industrial, automotive and aerospace applications*. John Wiley & Sons, 2021.

[63] N. Pan, X. Wu, Y. Chi, X. Liu, and C. Liu, "Combined failure acoustical diagnosis based on improved frequency domain blind deconvolution," in *Proceeding of the Journal of Physics: Conference Series*, vol. 364, p. 012078, 2012.

[64] H. Liu, L. Li, and J. Ma, "Rolling bearing fault diagnosis based on stft-deep learning and sound signals," *Shock and Vibration*, vol. 2016, 2016.

[65] Y. Gritli, L. Zarri, C. Rossi, F. Filippetti, G.-A. Capolino, and D. Casadei, "Advanced diagnosis of electrical faults in wound-rotor induction machines," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 9, pp. 4012–4024, 2012.

[66] R. Yan and R. X. Gao, "Hilbert–huang transform-based vibration signal analysis for machine health monitoring," *IEEE Transactions on Instrumentation and measurement*, vol. 55, no. 6, pp. 2320–2329, 2006.

[67] V. Climente-Alarcon, J. A. Antonino-Daviu, M. Riera-Guasp, and M. Vlcek, "Induction motor diagnosis by advanced notch fir filters and the wigner–ville distribution," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 8, pp. 4217–4227, 2013.

[68] G. H. B. Foo, X. Zhang, and D. M. Vilathgamuwa, "A sensor fault detection and isolation method in interior permanent-magnet synchronous motor drives based on an extended kalman filter," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 8, pp. 3485–3495, 2013.

[69] A. S. Willsky and H. L. Jones, "A generalized likelihood ratio approach to state estimation in linear systems subjects to abrupt changes," in *Proceeding of the IEEE Conference on Decision and Control including the 13th Symposium on Adaptive Processes*, pp. 846–853, IEEE, 1974.

[70] C. Bakiotis, J. Raymond, and A. Rault, "Parameter and discriminant analysis for jet engine mechanical state diagnosis," in *Proceeding of the IEEE Conference on Decision and Control*, pp. 1–11, 1979.

[71] R. Isermann, "Process fault detection based on modeling and estimation methods—a survey," *automatica*, vol. 20, no. 4, pp. 387–404, 1984.

[72] S. Simani and C. Fantuzzi, "Dynamic system identification and model-based fault diagnosis of an industrial gas turbine prototype," *Mechatronics*, vol. 16, no. 6, pp. 341–363, 2006.

[73] E. Frisk, *Residual generation for fault diagnosis*. PhD thesis, Linköpings universitet, 2001.

[74] A. Marcos, S. Ganguli, and G. J. Balas, "An application of $\mathcal{H}_\infty$ fault detection and isolation to a transport aircraft," *Control Engineering Practice*, vol. 13, no. 1, pp. 105–119, 2005.

[75] G. Murad, I. Postlethwaite, and D. Gu, "A robust design approach to integrated controls and diagnostics," *IFAC Proceedings Volumes*, vol. 29, no. 1, pp. 6518–6523, 1996.

[76] E. Frisk and L. Nielsen, "Robust residual generation for diagnosis including a reference model for residual behavior," *Automatica*, vol. 42, no. 3, pp. 437–445, 2006.

[77] J. CHEN and H. ZHANG, "Robust detection of faulty actuators via unknown input observers," *International Journal of Systems Science*, vol. 22, no. 10, pp. 1829–1839, 1991.

[78] R. V. Beard, *Failure accomodation in linear systems through self-reorganization*. PhD thesis, Massachusetts Institute of Technology, 1971.

[79] H. L. Jones, *Failure detection in linear systems*. PhD thesis, Massachusetts Institute of Technology, 1973.

[80] R. N. Clark, D. C. Fosth, and V. M. Walton, "Detecting instrument malfunctions in control systems," *IEEE Transactions on Aerospace and Electronic Systems*, vol. -, no. 4, pp. 465–473, 1975.

[81] M.-A. Massoumnia, "A geometric approach to the synthesis of failure detection filters," *IEEE Transactions on automatic control*, vol. 31, no. 9, pp. 839–846, 1986.

[82] A. Emami-Naeini, M. M. Akhter, and S. M. Rock, "Effect of model uncertainty on failure detection: the threshold selector," *IEEE Transactions on Automatic Control*, vol. 33, no. 12, pp. 1106–1115, 1988.

[83] P. Frank and N. Kiupel, "Fuzzy supervision and application to lean production," *International journal of systems science*, vol. 24, no. 10, pp. 1935–1944, 1993.

[84] P. M. Frank and B. Köppen-Seliger, "Fuzzy logic and neural network applications to fault diagnosis," *International journal of approximate reasoning*, vol. 16, no. 1, pp. 67–88, 1997.

[85] W. Rudin, *Real and Complex Analysis*. McGraw-Hill Singapore, 1986.

[86] L. Ljung, *System Identification (2nd Ed.): Theory for the User*. USA: Prentice Hall PTR, 1999.

[87] Y. S. Abu-Mostafa, M. Magdon-Ismail, and H.-T. Lin, *Learning from data*, vol. 4. AMLBook New York, 2012.

[88] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American mathematical society*, vol. 68, no. 3, pp. 337–404, 1950.

[89] F. Cucker and S. Smale, "On the mathematical foundations of learning," *Bulletin of the American mathematical society*, vol. 39, no. 1, pp. 1–49, 2002.

[90] G. Kimeldorf and G. Wahba, "Some results on tchebycheffian spline functions," *Journal of mathematical analysis and applications*, vol. 33, no. 1, pp. 82–95, 1971.

[91] T. Poggio and F. Girosi, "Networks for approximation and learning," in *Proceedings of the IEEE*, pp. 1481–1497, IEEE, 1990.

[92] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, no. 3, pp. 293–300, 1999.

[93] L. Ljung, B. Wahlberg, and H. Hjalmarsson, "Model quality: the roles of prior knowledge and data information," in *Proceedings of the 30th IEEE Conference on Decision and Control*, pp. 273–278, 1991.

[94] A. J. Helmicki, C. Jacobson, and C. Nett, "Identification in $\mathcal{H}_\infty$: a robustly convergent, nonlinear algorithm," in *Proceeding of the American Control Conference*, pp. 386–391, IEEE, 1990.

[95] A. Helmicki, C. Jacobson, and C. Nett, "$\mathcal{H}_\infty$ identification of stable lsi systems: A scheme with direct application to controller design," in *Proceeding of the American Control Conference*, pp. 1428–1434, IEEE, 1989.

[96] G. Goodwin, M. Gevers, and D. Mayne, "Bias and variance distribution in transfer function estimation," *IFAC Proceedings Volumes*, vol. 24, no. 3, pp. 811–816, 1991.

[97] X. Bombois, B. Anderson, and M. Gevers, "Frequency domain uncertainty sets with guaranteed probability level in prediction error identification," in *Proceeding of the 16th International Symposium on Mathematical Theory of Networks and Systems*, 2004.

[98] X. Bombois, A. Den Dekker, M. Barenthin, and P. M. Van den Hof, "Effect of model structure and signal-to-noise ratio on finite-time uncertainty bounding in prediction error identification," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pp. 494–499, IEEE, 2009.

[99] M. B. Christopher, *Pattern Recognition and Machine Learning*. Springer, 2006.

[100] Y. Fujimoto and T. Sugie, "Informative input design for kernel-based system identification," *Automatica*, vol. 89, pp. 37–43, 2018.

[101] E. Walter and H. Piet-Lahanier, "Estimation of parameter bounds from bounded-error data: a survey," *Mathematics and Computers in simulation*, vol. 32, no. 5-6, pp. 449–468, 1990.

[102] S. Borchers, S. Raković, and R. Findeisen, "Set membership parameter estimation and design of experiments using homothety," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 9035–9040, 2011.

[103] B. Noack, F. Pfaff, and U. D. Hanebeck, "Combined stochastic and set-membership information filtering in multisensor systems," in *Proceeding of the 15th International Conference on Information Fusion*, pp. 1218–1224, IEEE, 2012.

[104] G. C. Goodwin, J. H. Braslavsky, and M. M. Seron, "Non-stationary stochastic embedding for transfer function estimation," *Automatica*, vol. 38, no. 1, pp. 47–62, 2002.

[105] J. M. Maciejowski, "Multivariable feedback design," *Electronic systems engineering series*, 1989.

[106] M. J. Grimble, *Robust industrial control systems: optimal design approach for polynomial systems*. John Wiley & Sons, 2006.

[107] R. Beaven, M. Wright, and D. Seaward, "Weighting function selection in the $\mathcal{H}_\infty$ design process," *Control Engineering Practice*, vol. 4, no. 5, pp. 625–633, 1996.

[108] H. Oloomi and B. Shafai, "Weight selection in mixed-sensitivity robust control for improving the sinusoidal tracking performance," in *Proceeding of the 42nd IEEE International Conference on Decision and Control*, vol. 1, pp. 300–305, 2003.

[109] I. Jovik and B. Lennartson, "On the choice of criteria and weighting functions of an $\mathcal{H}_2/\mathcal{H}_\infty$design problem," *IFAC Proceedings Volumes*, vol. 29, no. 1, pp. 1351–1356, 1996.

[110] P. Lundström, S. Skogestad, and Z.-Q. Wang, "Performance weight selection for $\mathcal{H}_\infty$ and μ-control methods," *Transactions of the Institute of Measurement and Control*, vol. 13, no. 5, pp. 241–252, 1991.

[111] M. Ortega and F. Rubio, "Systematic design of weighting matrices for the $\mathcal{H}_\infty$ mixed-sensitivity problem," *Journal of Process Control*, vol. 14, no. 1, pp. 89–98, 2004.

[112] A. Lanzon and M. Cantoni, "On the formulation and solution of robust performance problems," *Automatica*, vol. 39, no. 10, pp. 1707–1720, 2003.

[113] A. Lanzon, "Weight optimisation in $\mathcal{H}_\infty$ loop-shaping," *Automatica*, vol. 41, no. 7, pp. 1201–1208, 2005.

[114] M. Osinuga, S. Patra, and A. Lanzon, "State-space solution to weight optimization problem in $\mathcal{H}_\infty$ loop-shaping control," *Automatica*, vol. 48, no. 3, pp. 505–513, 2012.

[115] B. Wahlberg and L. Ljung, "Design variables for bias distribution in transfer function estimation," *IEEE Transactions on Automatic Control*, vol. 31, no. 2, pp. 134–144, 1986.

[116] L. Ljung, G. C. Goodwin, and J. C. Agúero, "Stochastic embedding revisited: A modern interpretation," in *Proceeding of the 53rd IEEE Conference on Decision and Control*, pp. 3340–3345, 2014.

[117] L. Ljung, "Revisiting total model errors and model validation," *Journal of Systems Science and Complexity*, vol. 34, no. 5, pp. 1598–1603, 2021.

[118] G. Goodwin, M. Gevers, and B. Ninness, "Quantifying the error in estimated transfer functions with application to model order selection," *IEEE Transactions on Automatic Control*, vol. 37, no. 7, pp. 913–928, 1992.

[119] H. Hjalmarsson, "From experiment design to closed-loop control," *Automatica*, 2005.

[120] H. Golub and C. F. Van Loan, "Matrix computations, johns hopkins uni," *Press, London*, p. 115, 1996.

[121] K. Ogata *et al.*, *Modern control engineering*, vol. 5. Prentice hall Upper Saddle River, NJ, 2010.

[122] I. Landau, D. Rey, A. Karimi, A. Voda, and A. Franco, "A flexible transmission system as a benchmark for robust digital control*," *European Journal of Control*, vol. 1, no. 2, pp. 77–96, 1995.

[123] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[124] R. Toscano, *Structured Controllers for Uncertain Systems*. Springer, 2013.

[125] A. Boukhris, G. Mourot, and J. Ragot, "Non-linear dynamic system identification: A multi-model approach," *International Journal of Control*, 1999.

[126] H. Lhachemi, D. Saussie, and G. Zhu, "A robust and self-scheduled longitudinal flight control system: a multi-model and structured h-infinity approach," in *AIAA Guidance, Navigation, and Control Conference*, 2014.

[127] J. Shamma and M. Athans, "Gain scheduling: potential hazards and possible remedies," *IEEE Control Systems Magazine*, vol. 12, no. 3, pp. 101–107, 1992.

[128] H. Noura, D. Theilliol, J.-C. Ponsart, and A. Chamseddine, *Fault-tolerant control systems: Design and practical applications*. Springer Science & Business Media, 2009.

[129] V. Puig, J. Quevedo, T. Escobet, F. Nejjari, and S. de las Heras, "Passive robust fault detection of dynamic processes using interval models," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 5, pp. 1083–1089, 2008.

[130] C. Combastel and S.-A. Raka, "A set-membership fault detection test with guaranteed robustness to parametric uncertainties in continuous time linear dynamical systems," *IFAC Proceedings Volumes*, vol. 42, no. 8, pp. 1192–1197, 2009.

[131] P. Khargonekar and T. L. Ting, "Fault detection in the presence of modeling uncertainty," in *Proceedings of the 32nd IEEE Conference on Decision and Control*, pp. 1716–1721, IEEE, 1993.

[132] E. F. Camacho and C. B. Alba, *Model predictive control*. Springer science & business media, 2013.

[133] M. Anderson, M. Buehner, P. Young, D. Hittle, C. Anderson, J. Tu, and D. Hodgson, "Mimo robust control for hvac systems," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 3, pp. 475–483, 2008.

[134] Q. Zhang, L. Fiorentini, and M. Canova, "$\mathcal{H}_\infty$ robust control of an automotive air conditioning system," in *Proceeding of the American Control Conference*, pp. 5675–5680, 2014.

[135] G. Gauthier and B. Boulet, "Robust design of terminal ilc with $\mathcal{H}_\infty$ mixed-sensitivity approach for a thermoforming oven," *Journal of Control Science and Engineering*, vol. 2008, 2008.

[136] R. Zhang, A. Xue, and F. Gao, "Temperature control of industrial coke furnace using novel state space model predictive control," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2084–2092, 2014.

[137] M. Niederer, S. Strommer, A. Steinboeck, and A. Kugi, "Nonlinear model predictive control of the strip temperature in an annealing furnace," *Journal of Process Control*, vol. 48, pp. 1–13, 2016.

[138] M. Krzaczek and Z. Kowalczuk, "Gain scheduling control applied to thermal barrier in systems of indirect passive heating and cooling of buildings," *Control Engineering Practice*, vol. 20, p. 1325–1336, 12 2012.

[139] S. Peter and E. Benjamin, "Post-mortem diagnosis of bottling plants based on recorded data," *7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes (SAFEPROCESS)*, vol. 42, no. 8, pp. 1330 – 1335, 2009.

[140] P. Tsarouhas, "Evaluation of overall equipment effectiveness in the beverage industry: A case study," *International Journal of Production Research*, vol. 51, pp. 1–9, 2012.

[141] F. Castro and F. Araujo, "Proposal for oee (overall equipment effectiveness) indicator deployment in a beverage plant," *Brazilian Journal of Operations & Production Management*, vol. 9, pp. 71–84, 2012.

[142] H. Papadopoulos and C. Heavey, "Queueing theory in manufacturing systems analysis and design: A classification of models for production and transfer lines," *European Journal of Operational Research*, vol. 92, no. 1, pp. 1 – 27, 1996.

[143] V. Tobias, F. Stefan, and S. Peter, "Model-based fault localization in bottling plants," *Advanced Engineering Informatics*, vol. 29, no. 1, pp. 101 – 114, 2015.

[144] K. Renganathan and V. Bhaskar, "Modeling, analysis and performance evaluation for fault diagnosis and fault tolerant control in bottle-filling plant modeled using hybrid petri nets," *Applied Mathematical Modelling*, vol. 37, no. 7, pp. 4842 – 4859, 2013.

[145] D. Troupis, S. Manesis, N. T. Koussoulas, and T. Chronopoulos, "Computer integrated monitoring, fault identification and control for a bottling line," in *Proceedings of the IEEE Industry Applications Conference Thirtieth IAS Annual Meeting*, vol. 2, pp. 1549–1556, 1995.

[146] R. B. Randall and J. Antoni, "Rolling element bearing diagnostics - a tutorial," *Mechanical Systems and Signal Processing*, vol. 25, no. 2, pp. 485 – 520, 2011.

[147] R. M. Stewart, "Some useful data analysis techniques for gearbox diagnostic," in *Proceedings of the Meeting on the Applications of Time Series Analysis*, 1977.

[148] W. Wang and A. K. Wong, "Autoregressive model-based gear fault diagnosis," *Journal Vibration and Acoustic*, vol. 124, no. 2, pp. 172–179, 2002.

[149] P. McFadden, "A technique for calculating the time domain averages of the vibration of the individual planet gears and the sun gear in an epicyclic gearbox," *Journal of Sound and vibration*, vol. 144, no. 1, pp. 163–172, 1991.

[150] M. J. Dowling, "Application of non-stationary analysis to machinery monitoring," in *Proceeding of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 59–62, 1993.

[151] K. Sarda, A. Acernese, L. Russo, and M. Mazzoleni, "A comparison of envelope and statistical analyses for bearing diagnosis in hot steel rolling mill lines," in *Proceedings of the 47th Annual Conference of the IEEE Industrial Electronics Society (IECON)*, 2021.

[152] S. Du and R. Randall, "Encoder error analysis in gear transmission error measurement," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 212, no. 4, pp. 277–285, 1998.

[153] N. Valceschini, M. Mazzoleni, and F. Previdi, "Inertial load classification of low-cost electromechanical systems under dataset shift with fast end of line testing," *Engineering Applications of Artificial Intelligence*, vol. 105, p. 104446, 2021.

[154] M. Mazzoleni, F. Previdi, M. Scandella, and G. Pispola, "Experimental development of a health monitoring method for electro-mechanical actuators of flight control primary surfaces in more electric aircrafts," *IEEE Access*, vol. 7, 2019.

[155] M. Mazzoleni, Y. Maccarana, F. Previdi, G. Pispola, M. Nardi, F. Perni, and S. Toro, "Development of a reliable electro-mechanical actuator for primary control surfaces in small aircrafts," in *Proceeding of the IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 1142–1147, 2017.

[156] M. Mazzoleni, M. Scandella, L. Maurelli, and F. Previdi, "Mechatronics applications of condition monitoring using a statistical change detection method," in *Proceeding of the 21th IFAC World Congress*, 2020.

[157] M. Mazzoleni, M. Scandella, Y. Maccarana, F. Previdi, G. Pispola, and N. Porzi, "Condition monitoring of electro-mechanical actuators for aerospace using batch change detection algorithms," in *Proceedings of the IEEE Conference on Control Technology and Applications (CCTA)*, pp. 1747–1752, 2018.

[158] A. Rausand, Marvin adn Høyland, *System Reliability Theory: Models, Statistical Methods, and Applications, Second Edition*. Springer-Verlag London, 1994.

[159] MIL-STD-1629A, "Procedures for performing a failure mode, effects and criticality analysis," tech. rep., USA Department of defense, Washington DC, 1994.

[160] P. Astakhov, *Resistance to motion of railway rolling stock*. -, 1966.

[161] R. Isermann, "Fault diagnosis of machines via parameter estimation and knowledge processing—tutorial paper," *Automatica*, vol. 29, no. 4, pp. 815–835, 1993.

[162] P. Moster, "Gear fault detection and classification using learning machines," *Sound & vibration*, vol. 38, pp. 22–27, 03 2004.

[163] R. Tempo, E. Bai, and F. Dabbene, "Probabilistic robustness analysis: explicit bounds for the minimum number of samples," in *Proceedings of 35th IEEE Conference on Decision and Control*, 1996.

[164] H. Akaike, "A new look at the statistical model identification," *IEEE transactions on automatic control*, vol. 19, no. 6, pp. 716–723, 1974.

[165] J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, no. 5, pp. 465–471, 1978.

[166] S. Arlot, A. Celisse, *et al.*, "A survey of cross-validation procedures for model selection," *Statistics surveys*, vol. 4, pp. 40–79, 2010.

[167] J. G. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N. V. Chawla, and F. Herrera, "A unifying view on dataset shift in classification," *Pattern Recognition*, vol. 45, no. 1, pp. 521 – 530, 2012.

[168] B. Efron and R. Tibshirani, "Improvements on cross-validation: the 632+ bootstrap method," *Journal of the American Statistical Association*, vol. 92, no. 438, pp. 548–560, 1997.

[169] J. Sjöberg, T. McKelvey, and L. Ljung, "On the use of regularization in system identification," *IFAC Proceedings Volumes*, vol. 26, no. 2, Part 5, pp. 75–80, 1993.

[170] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, 1996.

[171] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *The Annals of statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[172] A. N. Tikhonov, V. J. Arsenin, V. I. Arsenin, V. Y. Arsenin, *et al.*, *Solutions of ill-posed problems*. Vh Winston, 1977.

[173] T. Chen, H. Ohlsson, and L. Ljung, "On the estimation of transfer functions, regularizations and gaussian processes—revisited," *Automatica*, vol. 48, no. 8, pp. 1525–1535, 2012.

[174] T. Chen, H. Ohlsson, G. C. Goodwin, and L. Ljung, "Kernel selection in linear system identification part ii: A classical perspective," in *Proceeding of the 50th IEEE Conference on Decision and Control and European Control Conference*, pp. 4326–4331, IEEE, 2011.

[175] F. P. Carli, "On the maximum entropy property of the first-order stable spline kernel and its implications," in *Proceeding of the IEEE Conference on Control Applications (CCA)*, pp. 409–414, IEEE, 2014.

[176] F. P. Carli, A. Chiuso, and G. Pillonetto, "Efficient algorithms for large scale linear system identification using stable spline estimators," *IFAC Proceedings Volumes*, vol. 45, no. 16, pp. 119–124, 2012.

[177] T. Chen and L. Ljung, "Implementation of algorithms for tuning parameters in regularized least squares problems in system identification," *Automatica*, vol. 49, no. 7, pp. 2213–2220, 2013.

[178] T. Chen, M. S. Andersen, L. Ljung, A. Chiuso, and G. Pillonetto, "System identification via sparse multiple kernel-based regularization using sequential convex optimization techniques," *IEEE Transactions on Automatic Control*, vol. 59, no. 11, pp. 2933–2945, 2014.

[179] D. J. MacKay, "Bayesian interpolation," *Neural computation*, vol. 4, no. 3, pp. 415–447, 1992.

[180] S. Salsa, *Partial differential equations in action: from modelling to theory*, vol. 99. Springer, 2016.