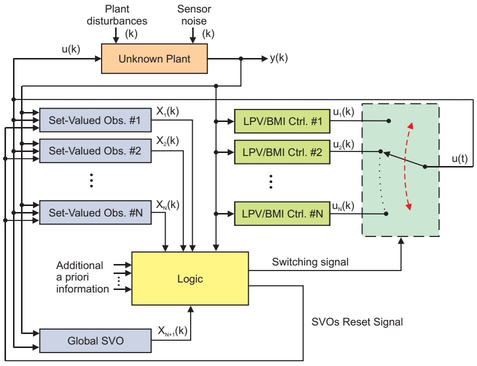
UNIVERSIDADE TÉCNICA DE LISBOA
INSTITUTO SUPERIOR TÉCNICO



Multiple-Model Adaptive Control of Uncertain LPV Systems

Paulo André Nobre Rosa

Supervisor: Professor Doctor Carlos Silvestre
Co-Supervisor: Professor Doctor Michael Athans

Thesis approved in public session to obtain the PhD Degree in Electrical and Computer Engineering

Jury final classification: **Pass with Distinction**

Jury

Chairperson: Chairman of the IST Scientific Board

Members of the Committee:
Professor Doctor Michael Athans
Professor Doctor Giuseppe Casalino
Professor Doctor Jeff Shawky Shamma
Professor Doctor João Manuel Lage de Miranda Lemos
Professor Doctor Miguel Afonso Dias de Ayala Botto
Professor Doctor Carlos Jorge Ferreira Silvestre

2011

To Nicole,

my parents,

and my sister.

“Everything should be made as simple as possible, but no simpler.”

(Albert Einstein)

Abstract

A thorough methodology to design robust adaptive controllers for uncertain Linear Parameter Varying (LPV) systems, with stability and performance guarantees, is presented. Multiple-Model Adaptive Control (MMAC) strategies are adopted, due to their advantages in terms of design and implementation. Hence, the core of this thesis is devoted, on the one hand, to the design of high-performance Local Non-Adaptive Robust Controllers (LNARCs) and, on the other, to the development of enhanced decision subsystems.

The design of LNARCs, robust against parametric and complex-valued uncertainties, and which are able to cope with time-variations of the dynamics of the plant, is tackled resorting to an optimization procedure with Bilinear Matrix Inequalities (BMIs) constraints.

A novel supervisor for MMAC architectures – the Stability Overlay (SO) – is also proposed, enabling closed-loop stability guarantees for uncertain and time-varying environments.

A whole new approach to MMAC is also introduced, that relies on Set-Valued Observers (SVOs) to *falsify* regions of uncertainty. This control architecture, referred to as MMAC/SVO, guarantees, under mild assumptions, stability and performance for the closed-loop system. Moreover, the developed model falsification strategy is also applied to Fault Detection and Isolation (FDI). As a caveat, the computational requirements of the SVOs can be higher than the alternatives.

Resumo

Nesta tese, propõe-se uma metodologia para sintetizar controladores adaptativos para sistemas dinâmicos lineares variantes nos parâmetros, cuja modelação inclui incerteza, garantindo estabilidade e desempenho robustos. Dadas as suas vantagens, tanto a nível de síntese, como a nível de implementação, optou-se por estratégias de controlo adaptativo multi-modelo (*Multiple-Model Adaptive Control* – MMAC). Como tal, esta tese centra-se, por um lado, no desenho de controladores localmente robustos de elevado desempenho, e, por outro, no desenvolvimento de sistemas de decisão.

A síntese de controladores não-adaptativos, robustos em relação a incertezas paramétricas e/ou complexas, é endereçada recorrendo a problemas sujeitos a restrições sob a forma de desigualdades matriciais bilineares, tendo em conta que a dinâmica do sistema pode ser variante no tempo.

Propõe-se, também, um novo supervisor para arquitecturas MMAC, designado por supervisor de estabilidade, que permite garantir estabilidade da malha-fechada, inclusive para sistemas variantes no tempo e com incertezas de modelação.

Finalmente, introduz-se um novo método de MMAC, baseado em observadores de conjuntos para invalidar modelos, e que garante estabilidade e desempenho para a malha fechada. O método de invalidação de modelos proposto é também aplicado à síntese de sistemas de detecção de falhas. Como desvantagem, salienta-se o elevado nível de requisitos computacionais.

Acknowledgments

Firstly, I take this opportunity to express my utmost appreciation to my advisors, Professor Carlos Silvestre and Professor Michael Athans, for their help, support, and guidance, since the beginning of my research work. I will be forever grateful for their outstanding contribution in providing me with solid scientific foundations, while not neglecting the importance of being creative and bold. I truly enjoyed our long debates on adaptive control and it is my deepest belief that I experienced, in the last four years, the greatest opportunity in my life to learn such an impressive amount of new ideas, philosophies, and ways of thinking. Although they have provided me with the necessary freedom to experiment new challenges and pursue novel scientific results, they have taught me to always question my own conclusions, even when they seemed unshakable. Their comments have impacted not only my academic path, but also my personal life.

I am also extremely thankful to Professor Jeff Shamma, for his supreme and insightful comments on my work, for his impressive suggestions of research directions, for sharing a small part of his remarkable knowledge with me, and for so kindly receiving me at Georgia Tech. The last two years of my PhD were tremendously impacted by his contributions and way of thinking.

My deepest gratitude is also addressed to the remaining members of my doctoral committee, namely Professor João Lemos, and Professor Miguel Ayala Botto. Their comments were key in enhancing my PhD, and their insightful concerns led to a far better understanding of the ideas we were trying to develop.

I also wish to acknowledge Professor Gary Balas, for his insightful comments on robust control, and for his tremendous help during the beginning of my PhD.

I will always cherish my stay at the University of Minnesota. It was a remarkable experience, and an incredible opportunity to meet new people.

My thanks go also to Dr. Sajjad Fekri, for the help provided in the use of the RMMAC and mixed-μ synthesis toolboxes, and to Vahid Hassani, for his friendship and for the fruitful and pleasant discussions on adaptive control.

I would like to thank my girlfriend, Nicole, my parents, Madalena and Martinho, my sister, Teresa, my grandparents, and the rest of my family, for their unconditional support, endless love, and utmost tolerance. I will be forever in debt for their encouragement and patience through the years.

At last, but not the least, I am also thankful to all my friends, especially to those at the Electrical and Computer Engineering department of IST, at ISR, at the University of Minnesota, and at the Georgia Institute of Technology. In particular, I would like to thank Samuel Pinto, Carlos Oliveira, João Tiago, George Chasparis, George Kotsalis, Bálint Vanek, João Almeida, and Manuel Rufino, for their support and friendship.

This work was sponsored by a PhD Student Scholarship, SFRH/BD/30470/2006, from the Fundação para a Ciência e a Tecnologia.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Adaptive control laws are required in many practical applications, where a single (non-adaptive) controller is not able to achieve the stability and/or performance requirements of the problem at hand. This happens because every physical system can only be known up to some finite bound on the accuracy, specially when there are uncertain real parameters impacting on the dynamics of the plant and changing with time.

Although several approaches have been proposed to address this problem during the last decade, there are still many open questions. In particular, some of the solutions available in the literature posit assumptions that do not hold in practice, while some other methods lack in terms of theoretical guarantees of stability, let alone performance.

Thus, the goal of this PhD thesis is to provide a theoretically sound methodology, to design robust adaptive controllers for Linear Parameter Varying (LPV) uncertain plants, without posing assumptions that are restrictive from a practical point of view, and achieving high levels of closed-loop performance.

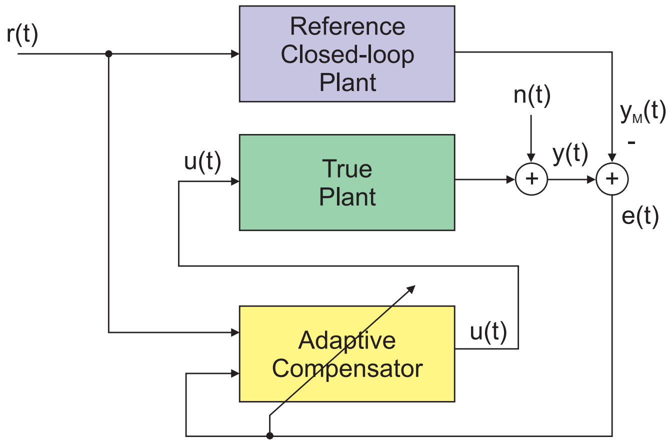


Figure 1.1: General model reference adaptive control architecture.

1.2 Previous Work and Brief Literature Review

Beginning in the mid 1960's, the research in adaptive control led to several different types of architectures, with different sorts of design specifications. Examples of the wide applicability of these adaptive control laws include Fault Detection and Isolation (FDI), Fault Tolerant Control (FTC), aircraft control, vibration control, power systems control, biomedical solutions, automotive suspension systems control, and flexible space structures control [1–24].

A well-known strategy, referred to as Model Reference Adaptive Control (MRAC) and depicted in Fig. 1.1 – see [25–30] and references therein – compares the output of the plant, $y(t)$, at each time, t, with that of a given reference model, $y_M(t)$, and tunes the parameters of the controller according to the output error $e(t)$, so as to follow the reference model output.

Another important class of adaptive control, illustrated in Fig. 1.2, is referred to as Multiple-Model Adaptive Control (MMAC). This architecture has several advantages [31], such as the fast adaptation when the plant dynamics change abruptly, and the ability to provide high levels of performance for different classes of dynamic models. The MMAC architecture uses a *divide-and-conquer* strategy to stabilize/control an uncertain plant. Instead of designing a compensator for a largely uncertain plant, the family of *admissible* plants to be controlled is split into several subsets. For each

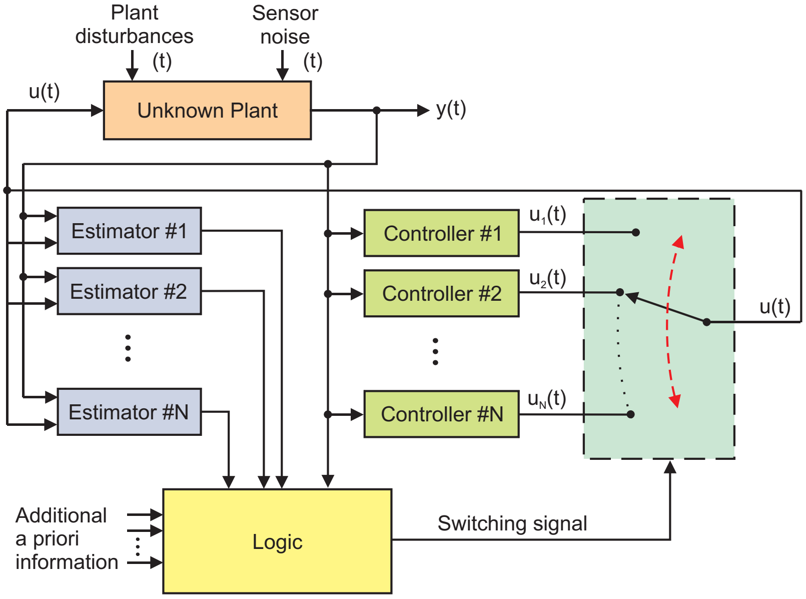


Figure 1.2: General multiple-model adaptive control architecture with N models.

uncertainty subset, a non-adaptive controller is synthesized. Then, an on-line model identification/estimation subsystem blends or switches the control signals that are applied to the plant, depending on the significance of each model [27, 32–38].

However, many adaptive control laws can lead to unstable closed-loop systems when connected to a plant with even the slightest discrepancies from the family of admissible plant models. This issue was first described in [39], in the so-called Rohrs *et al.* counterexample. Very small disturbances can be responsible for destabilizing the closed-loop system because of the unavoidable unmodeled high frequency dynamics, present in every physical system – see [25, 26, 28]. We stress that every non-ideal plant possesses such dynamics, and hence this is one of the primary concerns of this PhD thesis. We are, therefore, interested in the so-called *robust adaptive control problem* [27–29, 40].

To overcome this problem, several solutions are available in the literature of adaptive control, that pose more restrictive conditions on the plant, but that provide stability guarantees, at least for time-invariant plants – see [25–30, 34, 41–43] and references therein. Nevertheless, these strategies are in general very sensitive to unmeasured disturbances and measurement noise. Moreover, such architectures are in general not easily extended to multi-input/multi-output plants.

We address, in particular, the case where the process model has one or more parametric uncertainties, $\rho \in \Omega$. Although several switching MMAC methodologies are available to solve this problem, they all share the same principles: in terms of design, we divide the (large) set of parametric uncertainty, Ω, into N (small) subregions, Ω_i, $i = \{1, \cdots, N\}$ – see Fig. 1.3, for a single uncertain parameter case – and synthesize a non-adaptive compensator for each of them; in terms of implementation, we try to identify which region the uncertain vector of parameters, ρ, belongs to, and then select the controller designed for that region.

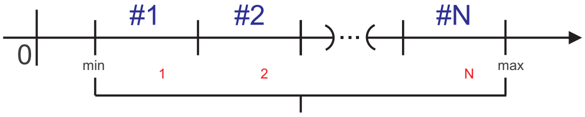


Figure 1.3: Uncertainty region, Ω, for one parameter ρ, split into N subsets.

In summary, a multiple-model adaptive control architecture, as depicted in Fig. 1.2, is in general composed of a bank of estimators, a bank of controllers and a switching logic. In some cases, the bank of estimators can also be substituted by a (single) global parameter identification subsystem. Hence, a MMAC architecture can typically be separated into two autonomous subsystems:

1. an identification/decision subsystem (ID subsystem);

2. a bank of non-adaptive controllers (control subsystem).

Each of these subsystems requires particular information regarding the plant to be controlled and, in some cases, can be designed independently. The identification/decision subsystem is usually composed of an estimator of parameters of the plant, and a switching logic, which should take into account the optimization of a given performance and/or stability criterion. The non-adaptive controllers are usually endowed with performance capabilities for a given region of uncertainty of the parameters, while yielding stability guarantees for a wider family of plants [44–49].

The classical MMAC [25, 29, 30] uses state-feedback controllers, connected to Multiple-Model Adaptive Estimation (MMAE) schemes [33, 35], while more recent approaches only require that one of the *eligible* (output-feedback) controllers is able to stabilize the plant. For instance, the authors of [50–52] evaluate the performance of the closed-loop obtained with each controller, using the so-called *Unfalsified Control Theory*, providing them with “rewards”. Thereafter, based upon the rewards received after its most recent utilization, each controller is disqualified or not. Notwithstanding the stability guarantees obtained with this method, the performance levels are highly conditioned by the frequency-dependent weights of the output and control signals. In a similar line-of-thought, the authors in [53] use the unfalsified control theory to select a controller for a plant, taking into account: a) the behavior each controller would show, had it been connected to the loop; b) the discrepancy between the output of the plant and the estimated output obtained by each of the possible dynamic models. The authors in [54] use a Lyapunov-based approach to select controllers, and hence require an in-depth knowledge of the plant. Another very promising approach is presented in [41], where a parameter estimator is used to select a controller, guaranteeing the stability of the closed-loop. The so-called supervisory control strategies in [31, 36, 37, 55] rely on the output (control or estimation) error to select, at each time instant, the controller that yields the smallest output error, while avoiding fast transients in the control signals.

Other adaptive control laws design strategies are available in the literature [16–18], including the ones based on classical control theory, statistical methods, fuzzy architectures, neural networks, etc. – see [56] and references therein. However, some of the assumptions posed by these methodologies are often unnatural or cannot be verified in practice. Moreover, the stability and performance guarantees provided by these approaches are in general only valid for a restrictive class of plants, and/or for a small region of uncertainty.

Another important MMAC architecture is the so-called Robust Multiple-Model Adaptive Control (RMMAC) – see [57–59] and references therein. The RMMAC is a multiple model approach that computes and uses the posterior probabilities of

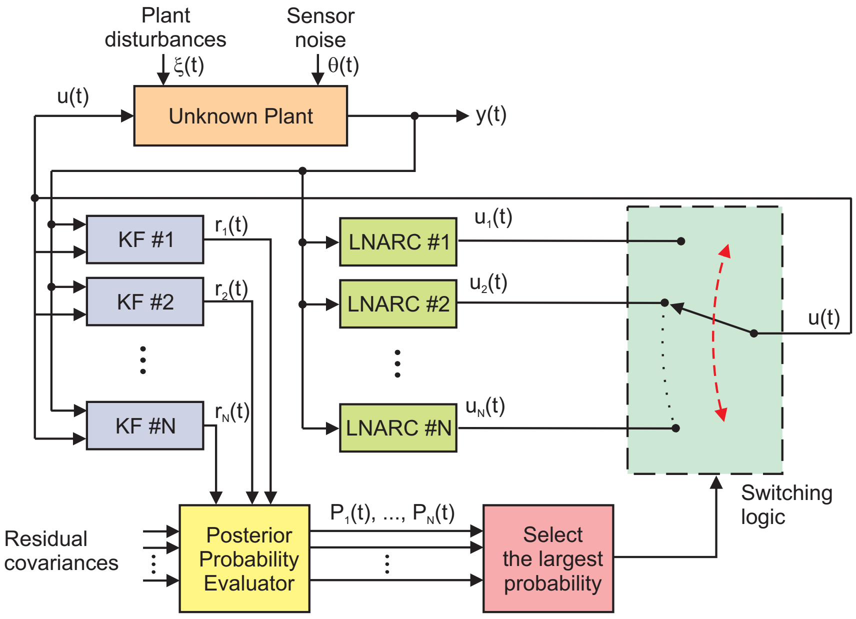


Figure 1.4: The switching RMMAC architecture with N models.

the uncertain parameters of the process model being in a specific region to switch or blend the outputs of a set of controllers, each of which designed for a given uncertainty region. The identification subsystem utilizes a bank of Kalman filters (KFs), while the control subsystem consists of a set of mixed-μ controllers, referred to as Local Non-Adaptive Robust Controllers (LNARCs). Figure 1.4 depicts the switching RMMAC architecture, for the case where N regions for the uncertain real parameters are used.

The KFs are used to generate residuals, which are obtained by subtracting the actual output of the plant to the outputs estimated by each of the KFs – we stress that Kalman filtering is one of the most widespread and significant theories on optimal filtering design. These residuals, in turn, are used by the Posterior Probability Evaluator (PPE), to compute an estimation of the posterior probabilities of a given uncertain parameter being in a specific interval.

In prior RMMAC studies [57–59], it was pointed out that the performance of any adaptive system must be evaluated not only for constant but unknown parameters, but also for time-varying parameters which undergo slow or rapid time-variations. The mass-spring-dashpot (MSD) testbed example presented in [59] and [58], was used to evaluate the RMMAC performance for such time-variations and for different intensities of unmeasurable plant disturbance. The results obtained and summarized in [60, 61], indicate that the violation of assumptions such as the knowledge on the intensity of the disturbances, can lead to a significant deterioration in terms of the closed-loop performance. The most important conclusion of the research to date is that unmeasured disturbances must be modeled well, otherwise the adaptive system "misbehaves".

1.3 Contributions of the Thesis

The main contributions of this PhD thesis are as follows:

- The development of a thorough methodology to design robust LTI controllers for uncertain LPV plants, referred to as LPV/BMI controllers – the goodness of this type of controllers is assessed in simulation, with a detailed comparison with the mixed-μ approach.

- The development of the Stability Overlay algorithms, that are capable of endowing a class of multiple-model adaptive control architectures with stability guarantees, for LTI and LPV plants, presenting, for the first time, a theoretical proof that one can, at least in some cases, detect and correct instability in adaptive control schemes for time-varying plants, without prior assumptions other than feasibility.

- The extension of the work on Set-Valued Observers (SVOs) to uncertain LPV plants, with a particular attention to the development of solutions to the main numerical and computational issues.

- The introduction of the *absolute input distinguishability* concept, with application to model falsification, and the derivation of necessary and sufficient conditions for absolute input distinguishability of uncertain LTI and LPV systems.

- The use of SVOs for model falsification, demonstrating its applicability, under mild assumptions, to: a) robust adaptive control with stability and performance guarantees; b) fault detection and isolation with guarantees of detecting faults and avoiding false alarms.

Each of the following chapters contains a list of its specific contributions.

1.4 Organization of the Thesis

1.4.1 The Control Subsystem

Even assuming perfect model identification, *i.e.*, that the switching signal in Fig. 1.2 is always *correct*, the RMMAC architecture does not provide stability guarantees for time-varying plants, since the mixed-μ performance bounds are only valid in a Linear Time-Invariant (LTI) environment. Furthermore, switching between controllers can lead to transients that may destabilize the plant.

Therefore, in Chapter 2, we start by focusing our attention in the design of non-adaptive controllers that potentially provide robust-stability and -performance guarantees for time-varying plants. In particular, we start by considering the design of robust non-adaptive controllers for Linear Parameter Varying (LPV) systems [62].

The design of robust controllers for LPV systems is able to cope with time-varying uncertain parameters, and can be cast as an optimization problem, – see [63–67] and references therein. One of the main advantages of LPV systems is that they allow for arbitrary dependence of the matrices of the dynamics on the time-varying parameters.

Nevertheless, the synthesis of LTI controllers for LPV plants with unknown parameters, that cannot be measured or estimated on-line with enough accuracy, is cast, in general, as an optimization procedure subject to bilinear matrix inequalities (BMIs). In Chapter 2, some methods to solve this problem are proposed. We stress that both mixed-μ and BMI/LPV controllers are LTI systems and hence do not depend upon the uncertain parameters of the plant.

1.4.2 Standard RMMAC vs RMMAC/BMI

In Chapter 3, the RMMAC/BMI architecture is introduced, by replacing the mixed-μ controllers in the standard RMMAC architecture by the LPV/BMI controllers described in Chapter 2. The classical RMMAC and the RMMAC/BMI are, thereafter, thoroughly compared in simulation, by means of an example.

1.4.3 Stability Overlay

The robustness of the control subsystem is an important part of an MMAC architecture. In spite of that fact, this may not be enough in order to provide global stability guarantees for the closed-loop system. Thus, in Chapter 4, we introduce a novel methodology that can endow virtually any switching MMAC system with stability guarantees.

The strategy developed, referred to as Stability Overlay (SO), takes into account both stability objectives – often robust to a very wide class of disturbances and model uncertainty – and performance requirements – that, in general, assume a stronger knowledge about the plant to be controlled. The algorithms proposed are based upon the work in [68] and were introduced in [69, 70]. They assess the "rewards" received by each controller after its most recent utilization, without any prior information on the bounds of the exogenous disturbances and measurement noise. A control law is then disqualified or not, based upon its rewards.

For the proposed SO methodology, it is neither required to know the model of the plant to be controlled, nor the properties of the disturbances. Still, it is clear that the performance of the closed-loop system can be severely affected if no knowledge is available regarding the plant. Nonetheless, the model-free characteristic of this method ensures robustness to several types of model uncertainty. In a sense, if the actual plant is *close enough* to a model of a plant in the family used to design the adaptive control law, then the adaptation runs as usual, without (or with minor) intervention of the SO. If, however, the actual plant or properties of the disturbances do not match the ones used during the design, the closed-loop system may become unstable. Therefore, instead of blindly continuing to use the adaptation law, the SO assesses the norm of the inputs and outputs of the system, and eventually switches to a controller that is able to stabilize the plant, as long as this controller belongs to the set of *legal* controllers that the SO is allowed to use.

Therefore, the SO can be seen as a *safety net* that can be integrated with many adaptive control algorithms, achieving high levels of performance, while providing robust stability guarantees for several different types of modeling errors. We show that the applicability of the SO is very wide, in the sense that it can be used in parallel with several adaptive control laws, as long as a few set of natural assumptions is satisfied, in particular that at least one controller in the set of *eligible* controllers is able to stabilize the plant. Moreover, the SO provides, for the first time, a theoretical proof that one can, at least in some cases, detect and correct instability for time-varying plants, without prior assumptions other than feasibility[1].

1.4.4 The Decision Subsystem

The SO, however, does not endow these adaptive control laws with any guarantees in terms of performance. This is due to the fact that, as mentioned, only a small number of assumptions is posed regarding the plant to be controlled. Since performance requirements are, in general, of major relevance in practical applications, a more sophisticated decision subsystem is developed in this thesis, by taking into account a deeper knowledge regarding the plant. The discussion on the assumptions required to achieve such goals is discussed in Chapter 5.

Most decision subsystems try to identify the *correct* uncertainty region, *i.e.*, the region where the uncertain parameters take value, by hypothesis testing or parameter estimation. In this PhD thesis, a different approach, referred to as *model falsification* or *model invalidation* and described also in Chapter 5 – see [71] and references therein –, is adopted, where the *wrong* regions are excluded as time goes by. In other words, if the time-evolution of the inputs and outputs of the plant cannot be explained by a dynamic model with uncertain parameters ρ, such that $\rho \in \Omega_i$ – see Fig. 1.3 – ,

[1] In MMAC, we say that the control problem is *feasible* if there is a controller, among the set of eligible ones, that is able to stabilize the plant.

then region Ω_i cannot be the one which the uncertain parameters belong to. Hence, by discarding those models, this technique aims to eventually disqualify all but one model. This remaining model, under certain assumptions, is the *correct* model of the plant, since it is the only one which is compatible with all the input/output sequences.

1.4.5 Set-Valued Observers

The previously described decision subsystem requires, therefore, the development of methods to invalidate models of a dynamic system. For dynamic uncertain models, described by differential inclusions, this can be posed as the problem of tracking a differential inclusion – we remark that there is a rich set of references in the mathematical literature on differential inclusions as discussed in [72–75].

For linear dynamic models, the problem of "disqualifying" regions can be tackled using Set-Valued Observers (SVOs) for linear systems – see [76]. These observers consider that the initial state of the system is uncertain, that there are disturbances acting upon the plant, and that the measurements are corrupted with noise. Therefore, the estimate of the state of the plant, at each time, is a set, instead of a single point. The observers in [76] are extended, in Chapter 6, to uncertain models, providing sufficient conditions for the convergence of the SVOs when implemented in a non-ideal environment.

1.4.6 Model Falsification Using SVOs

The aforementioned observers can, thereafter, be used for *model falsification*, in a very straightforward manner – see Fig. 1.5, where the *Logic* block uses the set-valued state estimates provided by the SVOs, to decide which dynamic models are compatible with the observed input/output sequence. Similarly to other model falsification architectures, a bank of observers – in the present case, SVOs – is used, each of which *tuned* for a pre-specified region of uncertainty. However, the observers can only *discard* regions, rather than identify them.

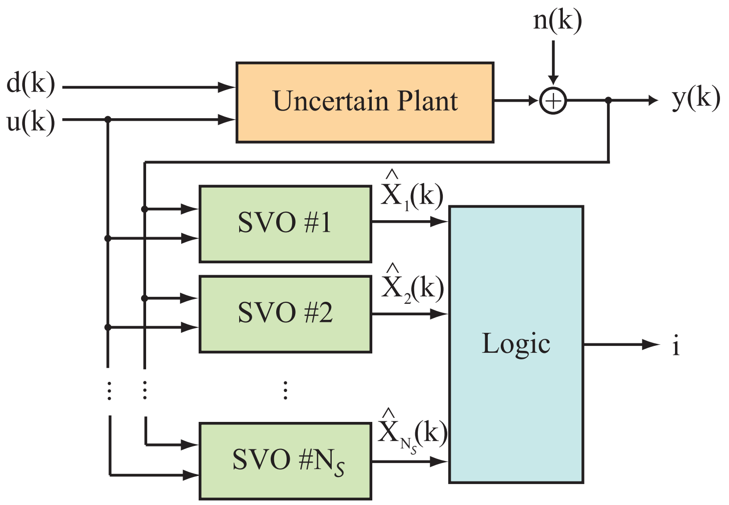


Figure 1.5: Model falsification architecture using set-valued observers.

Under mild assumptions on the disturbances and on the dynamic models, Chapter 7 shows that the *correct* model of the plant is eventually selected, if it is contained in the set of eligible ones. Moreover, under certain circumstances, an upper bound on the number of iterations required to perform this selection can also be computed as a function of the intensity of the disturbances.

MMAC Using SVOs

Using this model falsification scheme, it is also possible to decide which non-adaptive controllers should *not* be selected in an MMAC architecture. This strategy provides robust stability and performance guarantees for the closed-loop system, even when the model of the plant is uncertain and time-varying. Figure 1.6 depicts one of the possible MMAC/SVO architectures which are proposed in Chapter 7

We stress that developing an adaptive control system for uncertain LPV plants is the main goal of this thesis. Therefore, notwithstanding the theoretical stability- and performance-robustness guarantees provided, a series of simulation examples are presented, in order to evaluate the behavior of the proposed method when applied to actual physical systems.

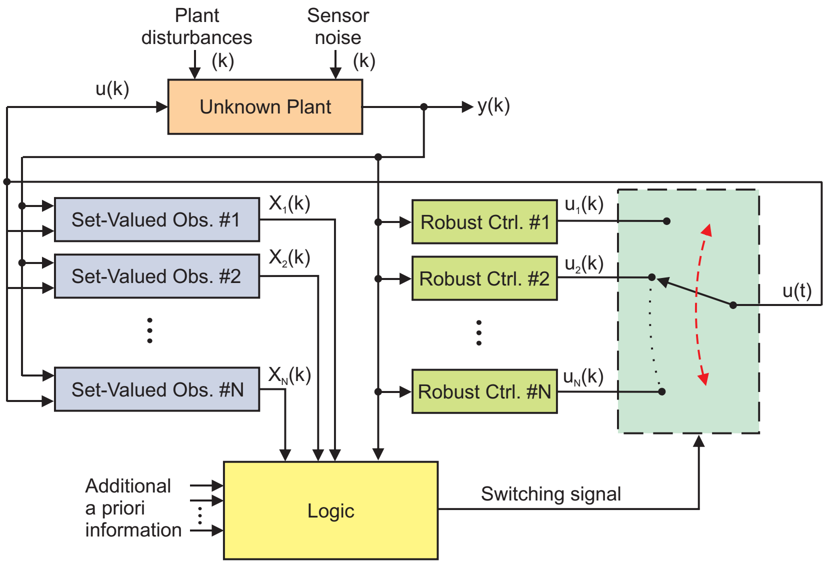


Figure 1.6: Multiple-model adaptive control architecture with set-valued observers.

FDI Using SVOs

As a different example of the applicability of the model falsification scheme using SVOs, Chapter 7 also describes the implementation of this methodology for Fault Detection and Isolation (FDI) of LPV systems. For certain scenarios, it can be proven that a given fault is always detected and isolated. Moreover, guarantees of avoiding false alarms can also be provided. Several examples illustrating the use of this model falsification technique in FDI are also shown in Chapter 7.

1.4.7 Comparison of MMAC/SVO with RMMAC/BMI

This chapter presents several results regarding the comparison of the Robust Multiple-Model Adaptive Control with LPV/BMI controllers (RMMAC/BMI) with the Multiple-Model Adaptive Control using Set-Valued Observers and RMS considerations (MMAC/SVO-RMS), when applied to a double Mass-Spring-Dashpot (MSD) plant. Although the design assumptions of each of the aforementioned adaptive control methodologies are different, it is worthwhile to evaluate the main advantages and shortcomings of each technique, using a system with an important physical meaning, such as the MSD plant. The comparison is performed resorting to a series of simulations that highlight the most significant features of the methods.

1.5 Notation and Definitions

This section introduces some of the mathematical notation used throughout the thesis. Further details will be presented later on, whenever necessary. In particular, definitions which are only useful in one chapter, will be introduced in that same chapter.

$\mathbb{Z}$	set of integers,
$\mathbb{Z}_o^+$	set of non-negative integers,
$\mathbb{R}^n$	set of ordered n-tuples of real numbers,
$\mathbb{R}^{n\times m}$	set of n by m matrices with elements in $\mathbb{R}$,
$\mathbb{C}^n$	set of ordered n-tuples of complex numbers,
$\mathbb{C}^{n\times m}$	set of n by m matrices with elements in $\mathbb{C}$,
x^{T}	transpose of vector x,
A^{T}	transpose of matrix A,
A^*	conjugate transpose of matrix A,
$\lambda_i(A)$	i-th eigenvalue of matrix A,
$\sigma_i(A)$	i-th singular value of matrix A,
$\bar{\sigma}(A)$	largest singular value of matrix A,
$\rho(A)$	spectral radius of matrix A,
$tr(A)$	trace of matrix A.

For a vector $v \in \mathbb{C}^n$, we define the vector p-norm of v as

$$|v|_p := \left(\sum_{i=1}^{n} |v_i|^p \right)^{\frac{1}{p}},$$

for $1 \leq p \leq \infty$. The subscript p is dropped whenever clear from the context.

For a matrix $A \in \mathbb{C}^{m\times n}$, we define the matrix norm induced by a vector p-norm as

$$\|A\|_p := \sup_{x\neq 0} \frac{\|Ax\|_p}{\|x\|_p},$$

and the Frobenius norm as

$$\|A\|_F := \sqrt{tr(A^*A)}.$$

A square hermitian matrix $A = A^*$ is said to be *positive definite* (*semi-definite*), denoted by $A > 0$ ($A \geq 0$), if

$$\forall_{x \neq 0} : x^*Ax > 0 \quad (x^*Ax \geq 0).$$

$\ell_p[0, \infty)$, or simply ℓ_p, consists of all sequences $x = \big(x(0), x(1), \cdots\big)$, with $x(i) \in \mathbb{R}^n$ for all $i \in \{0, 1, \cdots\}$, and such that

$$\sum_{i=0}^{\infty} |x(i)|^p < \infty.$$

Furthermore, the associated norm is defined as

$$\|x\|_p := \left(\sum_{i=0}^{\infty} |x(i)|^p\right)^{\frac{1}{p}}.$$

For simplicity, the dimension of $\mathbb{R}^n$ in the definition of ℓ_p is not specified.

In particular, $\ell_2[0, \infty)$ denotes the usual space of square measurable sequences in $\mathbb{R}^n$, with norm $\| \cdot \|_2$, and $\ell_\infty[0, \infty)$ denotes the space of bounded sequences, with norm

$$\|x\|_\infty := \sup_i |x(i)|.$$

For simplicity, we define $\ell_2 := \ell_2[0, \infty)$.

The ℓ_2-norm induced gain of an operator $H : \ell_2 \to \ell_2$ is defined as

$$\sup_{0 < \|w\|_2 < \infty} \frac{\|Hw\|_2}{\|w\|_2}.$$

$\mathcal{L}_2[0, \infty)$, or simply $\mathcal{L}_2$ consists of all square integrable and Lebesgue measurable functions defined on $\mathbb{R}$, with the inner product defined as

$$\langle f, g \rangle := \int_0^\infty f(t)^* g(t) dt,$$

for $f, g \in \mathcal{L}_2[0, \infty)$. Similarly, if the function is vector- or matrix-valued, the inner product is defined as

$$\langle f, g \rangle := \int_I tr\big(f(t)^* g(t)\big) dt.$$

For simplicity, we define $\mathcal{L}_2 := \mathcal{L}_2[0, \infty)$.

The $\mathcal{L}_2$-norm induced gain, or $\mathcal{H}_\infty$ norm, of an operator $J : \mathcal{L}_2 \to \mathcal{L}_2$ is defined as

$$\|J\| := \sup_{0<\|w\|_2<\infty} \frac{\|Jw\|_2}{\|w\|_2}.$$

For a linear time-invariant operator G, the corresponding transfer function is denoted by $G(s)$.

For $M \in \mathbb{C}^{n \times n}$, the *structured singular value* $\mu_{\mathcal{B}}(M)$ is defined as

$$\mu_{\mathcal{B}}(M) := \frac{1}{\min\{\bar{\sigma}(\Delta) : \Delta \in \mathcal{B}, \det(I - M\Delta) = 0\}}.$$

If $\det(I - M\Delta) = 0$ for all $\Delta \in \mathcal{B}$, then we define $\mu_{\mathcal{B}}(M) = 0$. Whenever $\mathcal{B}$ can be inferred from the context, we write

$$\mu(M) := \mu_{\mathcal{B}}(M).$$

Chapter 2

The Control Subsystem

2.1 Introduction

In order to be successfully implemented, a feedback controller must be robust against model uncertainty. Without robustness guarantees, even the slightest difference between the true plant dynamics and the model used to design the controller can lead to an unstable closed-loop – see for instance [39, 77, 78]. In this view, a methodology to synthesize Linear Time-Invariant (LTI) controllers for time-varying Linear Parameter Varying (LPV) plants (see [62, 79, 80]), with uncertain real-valued parameters and complex uncertainties, using Bilinear Matrix Inequalities (BMIs), is proposed in this chapter. These disturbance-rejection controllers, referred to as LPV/BMI controllers, are able to provide performance-robustness guarantees against different types of uncertainties on the model of the plant, while also providing performance-robustness guarantees against time-variations of the unmeasured parameters.

As mentioned in [81], Linear Time-Varying (LTV) systems depend on their future dynamics, despite the fact that, in general, their time-varying dynamics may be measurable *on-line*, but may not be available beforehand. Thus, in this thesis, we adopted the LPV framework, introduced in [62], which assumes that the dynamics of the plant depend upon time-varying parameters, whose corresponding time variations are not known a priori.

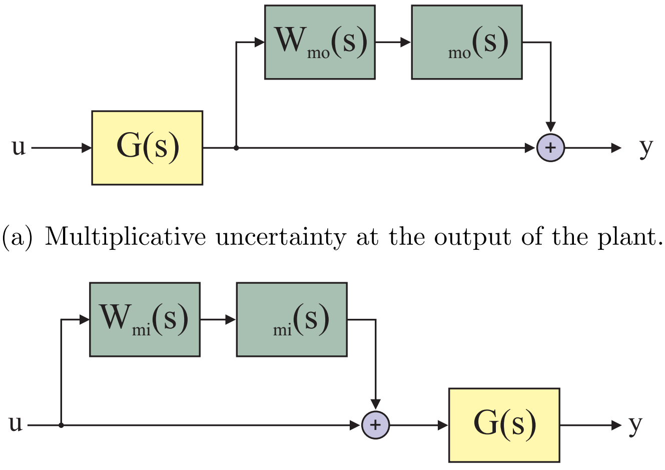
2.1.1 Types of Model Uncertainty

Several types of model uncertainty have been considered in the literature of dynamic systems. In particular, the so-called *complex-valued uncertainties*[1] have been used to model LTI uncertainties in the input and output of the nominal plant (*c.f.*, [77, 78]). These complex uncertainties are commonly divided into the following categories:

i) Multiplicative uncertainty – see Fig. 2.1;

ii) Additive uncertainty – see Fig. 2.2;

iii) Division uncertainty – see Fig. 2.3.



(a) Multiplicative uncertainty at the output of the plant.

(b) Multiplicative uncertainty at the input of the plant.

Figure 2.1: Multiplicative uncertainty.

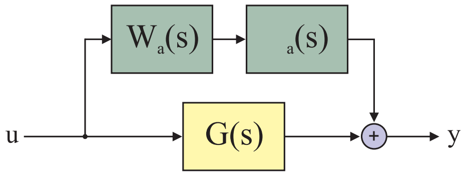
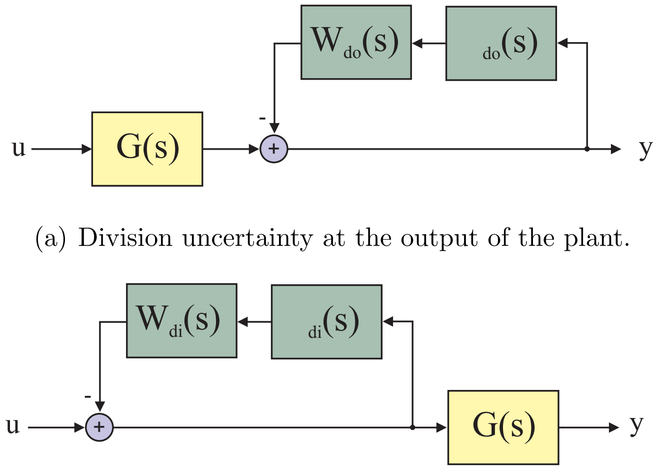


Figure 2.2: Additive uncertainty.

The multiplicative and division uncertainty can either be at the input or at the output of the nominal plant.

[1] Typically, representing unmodeled dynamics.



(a) Division uncertainty at the output of the plant.

(b) Division uncertainty at the input of the plant.

Figure 2.3: Division uncertainty.

The applicability of these types of uncertainties is considerably broad, since they can be used to model a wide range of dynamic phenomena. For instance, and as shown later on, the multiplicative model uncertainty in the input of the plant can be used to model unknown input time-delays, which should often be accounted for in realistic applications. As another example, changes in the input variable, u, which rapidly propagate to the output of the plant can be modeled by additive uncertainties. For further examples on this topic, the reader is referred to [77, 78] and references therein.

Despite of the many uses of the aforementioned types of uncertainties, they all can be represented by Linear Fractional Transformations (LTFs, [77]). These LFTs are suitable for robust-stability and -performance analysis and synthesis, as described in the sequel.

Another important class of model uncertainty is designated by *real-valued (or parametric) uncertainty*. This type of uncertainty also has an important physical meaning, since it can be used to model uncertainty in parameters of a plant. Typical examples of parametric uncertainty include uncertain spring stiffnesses, uncertain damping coefficients, uncertain gains of the sensors and actuators, among others.

A system with parametric uncertainty can also be represented in an LFT fashion, where the uncertainty block is now a diagonal block containing the uncertain vector

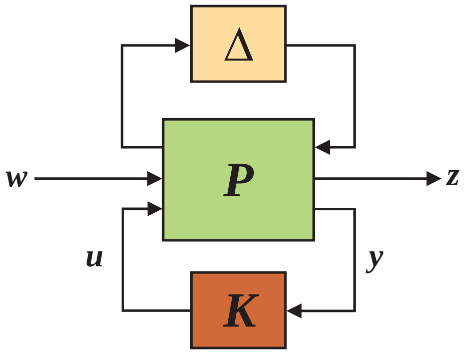


Figure 2.4: Mixed-μ synthesis block diagram.

of parameters. Nevertheless, it can also be modeled by an LPV system, which depends upon these uncertain parameters.

2.1.2 Controllers for Uncertain Dynamics

A well-know method to design LTI controllers, with disturbance-rejection objectives, for uncertain LTI plants, is called *μ-synthesis*, and is going to be used herein for comparison purposes. A mixed-μ controller is an approximation of the optimal controller in the $\mathcal{L}_2$-induced norm sense, from the exogenous inputs to the performance outputs – see [47, 77, 78, 82, 83]. The architecture for the synthesis of this type of controllers is depicted in Fig. 2.4. Despite the sub-optimality of the solution, these controllers are capable of handling different types of uncertainties, namely complex and parametric uncertainties, by using the so-called D,G-K iterations – see [44, 46, 48, 84] and references therein. *However, μ-controllers can only guarantee robust-performance and -stability for time-invariant systems.* This means that the controllers' guarantees of performance are valid for every "legal" uncertainty of the plant, as long as it does not vary with time.

This may be problematic, since in many applications (such as multiple-model adaptive control) it is desirable that these time-variations of the plant do not impact on the performance and, ultimately, the stability of the closed-loop plant. Nonetheless, if the time-varying parameters of the plant change *very slowly* with time, then the assumptions of the μ-controllers design are approximately met, and it has been shown in some examples that the results do not deteriorate significantly

– see [58–61].

A different approach to the design of robust controllers for LPV systems [62, 79, 80] has been developed that is able to handle time-varying uncertain parameters and that can be posed as an optimization problem, *e.g.*, minimize the $\mathcal{L}_2$-induced norm of the closed-loop – see [63–67, 85–89] and references therein. Furthermore, LPV systems can be time-varying and have arbitrary parameter dependence. As a caveat, complex uncertainties cannot be treated the same way as parametric ones, and some relaxations have to be introduced (for instance, using integral quadratic constraints) in order to take them into account during the controller design phase, thus adding conservatism to the solution – see [63]. Another option, which is adopted herein, resorts to the so-called *D-scales* to reduce the conservatism of the result obtained with the minimization of the $\mathcal{H}_\infty$-norm (see [77]).

When there are unknown parameters in the plant dynamics that cannot be measured nor estimated on-line with enough accuracy, this type of controller design problem can be cast as an optimization procedure subject to Bilinear Matrix Inequalities (BMIs). This is known to be an NP-hard problem, and hence cannot be used for very large design problems. Since new BMI software packages are now available that can solve this kind of optimization for medium size problems (plants with about 20 states), it is worthwhile to try to design and evaluate such controllers in the cases where the plant dynamics depend upon uncertain parameters. Moreover, the BMIs can also be solved by very simple methods, as shown in the sequel, that "eliminate" the bilinearity of the optimization procedure, and hence problems with larger numbers of states can be handled. We stress that both mixed-μ and LPV/BMI controllers are LTI systems and hence do not depend upon the uncertain parameters of the plant, but only on prior-known bounds.

From a physical point of view, uncertain plant parameters represent uncertain energy storage and/or uncertain gains. If the uncertain parameters do not change with time, then this initial "energy uncertainty" does not change; the robust μ-compensators do take into account this uncertain constant energy. Indeed the information about the upper and lower bound for each uncertain parameter is required to design the robust μ-compensator. We stress that we cannot guarantee stability robustness if the μ-compensators are used for a plant with time-varying parameters.

In the case of uncertain time-varying parameters, as in LPV systems, there is a time-varying exogenous energy added and/or subtracted to/from the dynamic system, in addition to the uncertain exogenous energy of the disturbances. This "time-varying uncertain energy" exists in addition to that provided by the plant disturbances and the control inputs. If the parameters of the LPV plant are measured (which is not the case in adaptive systems), then there is no uncertainty in the energy associated with the time-varying parameters. If, however, we do not measure the uncertain time-varying parameters, then the solution of the BMI problem presented in the sequel allows us to construct a time-invariant LPV/BMI compensator with guaranteed stability and performance of the closed-loop.

2.1.3 Main Contributions and Organization

The LPV/BMI compensator takes into account

a) the same performance requirements used to design the μ-compensator

b) the same upper and lower bounds for each real uncertain parameter as the μ-compensator

c) a bound on the maximum slope (time-rate-of-change) of each uncertain time-varying parameter (not required by the μ-compensator, *i.e.*, assumed zero)

Property (c), from a physical point of view, bounds the time-rate-of-change of the exogenous energy associated with each uncertain parameter. In this manner, the LPV/BMI compensator guarantees stability-robustness. Thus, property (c) allows us to use LPV/BMI compensators when the plant parameters are time-varying.

In the numerical example presented in the sequel, which uses a Mass-Spring-Dashpot (MSD) type of system with an uncertain spring, we will discuss with greater specificity the above "uncertain energy" issues.

The key contributions of this chapter are as follows:

1. The use of D-scales with LPV/BMI controllers and the development of the D-BMI algorithm;

2. The development of an algorithm to solve optimization problems subject to BMI constraints;

3. An insightful practical and theoretical comparison between LPV/BMI and mixed-μ controllers.

More precisely, the first main contribution of this chapter amounts for the use of D-scales with the LPV/BMI controllers, in a similar manner to what is done in the μ-synthesis. We also adopt the technique in [58, 59] to derive lower bounds in terms of performance for the closed-loop plant, while guaranteeing stability against model uncertainty. An iterative algorithm, referred to as D-BMI iterations, is proposed to synthesize this type of controllers.

The second main contribution of this chapter is concerned with the optimization of problems subject to BMI constraints. We claim that, for small problems (plants with less than 20 states), the use of general BMI solvers leads to a solution in a reasonable amount of time (a few hours). For plants with a higher number of states, we propose a faster algorithm that, by fixing the optimization variables one at a time, reduces the BMI problem to a Linear Matrix Inequalities (LMIs) problem.

The final main contribution of this chapter is the insightful comparison between LPV/BMI and mixed-μ controllers. We assess in detail the practical and theoretical differences between the two designs, when applied to LTI and Linear Time-Varying (LTV) plants, and illustrate, in simulation, the benefits of the proposed approach.

The remainder of this chapter is organized as follows: Section 2.2 introduces the mixed-μ controllers, which are going to be needed for comparison purposes; the synthesis of LPV/BMI controllers, as well as a detailed analysis of this type of controllers, is presented in Section 2.3; in Section 2.4, simulation results are shown that illustrate the applicability and the advantages of the LPV/BMI controllers; finally, in Section 2.5, we summarize and discuss the proposed methodologies.

2.2 Mixed-μ Controllers

The design of mixed-μ controllers (see, for instance, [46, 48, 77]) is done resorting to the so-called D,G-K iterations. This (local) optimization procedure is able to account for constant real parametric and complex-valued uncertainties on the model of the plant. As a caveat, the degree of conservatism of this method can become prohibitive if the plant depends upon more than a couple of uncertain parameters.

Figure 2.4 depicts the block diagram used to synthesize a mixed-μ controller. The Δ block represents the (possibly structured) uncertainties of the plant, while K is the controller that we are trying to synthesize. The resulting mixed-μ controller must have an $\mathcal{L}_2$-induced norm from the Δ block input to the corresponding output smaller than one, while minimizing the norm from the exogenous inputs, w, to the performance outputs, z, using the measurements y, that may be corrupted with noise. Hence, the mixed-μ design specifications can be stated as follows: synthesize an LTI controller which minimizes the $\mathcal{L}_2$-induced norm of the exogenous inputs to the performance outputs, while guaranteeing that the product of the $\mathcal{L}_2$-induced norm from w_Δ to z_Δ with the norm of the Δ block is smaller than 1.

The main advantage of mixed-μ controllers is that they can guarantee robust performance and stability for the aforementioned different types of uncertainties. However, it is assumed that the system to be controlled is linear and time-invariant, and therefore no guarantees can be provided for time-varying plants. This means that a given mixed-μ controller can guarantee robust performance for every *valid* realization of the plant dynamics, but cannot guarantee even stability if the plant dynamics are changing with time. Another practical problem with the mixed-μ approach is the deterioration of the performance when the parameter is time-varying. Since this type of controllers is not guaranteed to be robust to such variations, a significant loss in terms of performance can occur when that assumption is violated. Furthermore, the dynamics of the plant cannot depend arbitrarly upon the uncertain parameters. In fact, these uncertain parameters must be represented by means of an LFT, as previously described – see [77, 78].

The mixed-μ controller synthesis can be separated into two steps:

1. one for the computation of the D- and G-scales;

2. and another for the computation of an $\mathcal{H}_\infty$ controller.

In the proposed approach, we keep the computation of the D-scales unchanged, and eliminate the computation of the G-scales, by using the method in Section 2.3 to synthesize $\mathcal{H}_\infty$ controllers that are robust against real-valued parametric uncertainty. Therefore, the same structure of the mixed-μ controller design is going to be used to synthesize robust LTI controllers for LPV systems.

2.2.1 Robust Performance Specifications

The performance output, $z(\cdot)$, is often obtained by weighting certain variables of interest by a low-pass filter[2] of the form

$$W_p(s) = A_p \left(\frac{\alpha}{s+\alpha} \right)^q .$$

As previously mentioned, the design objective of the controller is to minimize the $\mathcal{L}_2$-induced norm from the exogenous inputs to this performance output, for all the legal realizations of Δ. In reference to Fig. 2.4, we say that *robust performance* is achieved by controller K if

$$\| F_u \left(F_\ell(P,K), \Delta \right) \| < 1, \underset{\Delta, \|\Delta\| \leq 1}{\forall} . \tag{2.1}$$

In (2.1), the upper and lower LFTs are denoted by F_u and F_ℓ, respectively – see [77].

As shown in [77], the condition in (2.1) can be used to design controllers yielding robust performance, by redefining the allowable realizations for the uncertainty block as

$$\Delta \in \left\{ \begin{bmatrix} \Delta_{\text{unc}} & 0 \\ 0 & \Delta_{\text{perf}} \end{bmatrix} : \Delta_{\text{unc}}, \Delta_{\text{perf}} \in \mathcal{RH}_\infty; \|\Delta_{\text{unc}}\| \leq 1, \|\Delta_{\text{perf}}\| \leq 1 \right\}, \tag{2.2}$$

where Δ_{unc} is used to describe complex-valued uncertainty (*e.g.*, unmodeled dynamics of the plant), and Δ_{perf} is a full complex-valued matrix of appropriate dimensions.

[2] It can also be used a band-pass filter in certain applications.

As stated in [34, 77], the robust performance condition in (2.1) is achieved if and only if

$$\mu(F_\ell(P, K)) < 1,$$

where the complex-μ (see [77]) is computed with respect to the structure of Δ in (2.2), and where Δ_{perf} is a full complex-valued uncertainty matrix with appropriate dimensions, reflecting the performance specifications.

The maximization of the performance from the exogenous inputs to the performance variables of interest, is done iteratively by using the following algorithm:

Algorithm 1: Algorithm for the maximization of the performance from the exogenous inputs to the performance outputs.

```
Input: $\eta$
Output: $K$, $A_p$
Initialization $\mu(0) = \infty$, $n = 1$, Synthesize mixed-$\mu$ controller and compute value of $\mu(1)$
while $|\mu(n) - \mu(n-1)| \leq \eta$ or $\mu(n) \geq 1$ do
    • $n = n + 1$;
    • Synthesize mixed-$\mu$ controller and compute value of $\mu(n)$;
    • if $\mu(n) < 1$ then
        Increase $A_p$
    else
        Decrease $A_p$
    end
end
```

In the algorithm, we denote by $\mu(k)$ the value of the complex-μ with respect to the structure of Δ in (2.2), at iteration k. The value of η defines the minimum variation of μ from one iteration to the other, such that it is worthwhile to keep the algorithm running.

In summary, we try to maximize the value of A_p, while guaranteeing the robust performance of the closed-loop system. This method has been extensively used in our previous work, as an evaluation tool to compare the performance of different controllers [58, 59].

2.3 LPV/BMI Controllers

As stress in the Introduction, LPV models represent nowadays a compromise between the global accuracy of nonlinear models and the straightforward controller synthesis techniques available for LTI representations. In *classical* LPV control, we assume that the matrices of the dynamics depend upon measurable exogenous time-varying parameters. Moreover, these parameters are usually confined to a given (convex) set, and the corresponding maximum time rate-of-variation may be assumed known.

2.3.1 Classical LPV Controllers

The type of controllers for disturbance rejection reviewed in the sequel (see [63, 86, 90, 91] and references therein for further details) can be applied to LPV systems that can be described by

$$\left\{\begin{array}{rcl} \dot{x}(t) & = & A\left(\rho(t)\right)x(t) + B\left(\rho(t)\right)u\left(t\right) + L\left(\rho(t)\right)d(t) \\ y(t) & = & C\left(\rho(t)\right)x(t) + N\left(\rho(t)\right)n(t) \end{array}\right. , \tag{2.3}$$

where $x(t)$ is the state of the system, $y(t)$ is the noisy measurements vector, $u(t)$ represents the control inputs, $\rho(t) = [\rho_1(t), \rho_2(t), \cdots, \rho_p(t)]^T$ is the vector of the (possibly time-varying) real-valued exogenous parameters of the plant, $d(t)$ are the exogenous disturbances, and $n(t)$ is the sensor noise. We further assume that $\rho(t) \in \Omega$ and $\dot{\rho}(t) \in \Lambda$, for all t, where Ω is a convex polytope of $\mathbb{R}^p$ and where

$$\Lambda := \left\{\lambda : |\lambda_i| \leq \nu_i, i = \{1, \cdots, p\}\right\}, \tag{2.4}$$

for given $\nu_i \geq 0$, $i = \{1, \cdots, p\}$. Notice that Λ is a hyper-parallelepiped. The dynamics of the controller to be designed are described by

$$\left\{\begin{array}{rcl} \dot{x}_c(t) & = & A_c(\rho(t))x(t) + B_c(\rho(t))u_c(t), \\ y_c(t) & = & C_c(\rho(t))x_c(t) + D_c(\rho(t))u_c(t). \end{array}\right. \tag{2.5}$$

The closed-loop plant with the controller in (2.5) is described by

$$\left\{\begin{array}{rcl} \dot{x}_{cl}(t) & = & A_{cl}\left(\rho(t)\right)x_{cl}(t) + B_{cl}\left(\rho(t)\right)w(t) \\ z(t) & = & C_{cl}\left(\rho(t)\right)x_{cl}(t) + D_{cl}\left(\rho(t)\right)w(t), \end{array}\right. \tag{2.6}$$

where $z(t)$ is the performance output and $w(t) = [d(t),\, n(t)]$ includes both exogenous disturbances and measurement noise.

The key principle in designing a controller for LPV systems is that we want to find a Parameter Dependent Lyapunov Function (PDLF), $V(x_{\text{cl}}, \rho)$, such that the stability of the closed-loop is guaranteed even for time-varying plants – see [63, 90, 92, 93] and references therein.

Instead of trying to find an arbitrary candidate PLDF for the closed-loop system, we are interested in finding quadratic PDLF, *i.e.*, functions of the form

$$\begin{cases} V : \mathbb{R}^n \times \Omega \to \mathbb{R}, \\ V(x_{\text{cl}}, \rho) = x_{\text{cl}}^T P(\rho) x_{\text{cl}}, \end{cases}$$

where

$$P(\rho) = \xi_1(\rho)P_1 + \xi_2(\rho)P_2 + \cdots + \xi_m(\rho)P_m, \tag{2.7}$$

(and where $P_1, P_2, \cdots, P_m$ are constant symmetric matrices) so that the problem becomes tractable, at the cost of some increased conservatism [94]. In the previous equation, $\xi_i : \Omega \to \mathbb{R}$, for each $i \in 1, \cdots, m$, is a differentiable function of ρ.

Remark 2.1: Although the candidate Lyapunov function may be parameter dependent, the matrices of the compensator are not. This is important given that, unlike "classical" LPV control, our approach aims to design LTI controllers that are non-parameter-dependent, and hence are robust to parametric uncertainty. $\diamond$

Remark 2.2: Describing the Lyapunov function by means of basis functions makes the problem tractable, but, at the same time, introduces conservatism in the solution. Nevertheless, we can have as many basis functions as we want, thus we can give arbitrary degrees of freedom to $P(\rho)$. $\diamond$

In this case, we get

$$\dot{V} = x_{\text{cl}}^T P A_{\text{cl}} x_{\text{cl}} + x_{\text{cl}}^T A_{\text{cl}}^T P x_{\text{cl}} + x_{\text{cl}}^T \dot{P} x_{\text{cl}},$$

where A_{cl} is the closed-loop system dynamics matrix, as in (2.6). We omitted the time-depence of the Lyapunov and plant matrices, for the sake of clarity.

It can be shown – see, for instance, [90, 92] – that the closed-loop system is stable if there exists $P(\rho) > 0$ such that

$$PA_{cl} + A_{cl}^T P + \dot{P} < 0, \tag{2.8}$$

for every *valid* value of ρ and $\dot{\rho}$. This idea is going to be used in the sequel in order to design robust controllers.

The same block diagram of Fig. 2.4 can be used to synthesize this type of controllers (for instance, with an $\mathcal{H}_\infty$ minimization objective), but in this case the plant is assumed to be time-varying and described by (2.3). Notice that $\rho(t)$ can only represent time-varying parameters, thus complex valued LTI uncertainties, such as unmodeled dynamics, are going to be handled using the small-gain theorem. The use of the D-scales is therefore required to reduce the conservatism of the solution, in the same way they are used in the D,G-K iterations. Integral quadratic constraints (IQCs) have also been successfully used to tackle this problem, adding some conservatism to the solution. The interested reader is referred to [63] and references therein. However, we do not examine IQCs in this thesis.

Since the controller synthesis problem involves testing an infinite number of Matrix Inequalities (MIs), several different structures for the LPV system have been proposed which reduce the problem to that of solving a finite number of MIs. For LPV systems that depend in an affine manner upon the parameters that, in turn, can only take values inside (or on the boundaries) of a convex polytope, a state-feedback solution may exist that guarantees stability for the (arbitrary or rate-bounded) time-varying plant, by checking a (finite) set of Linear Matrix Inequalities (LMIs) in the vertices of that polytope. With this approach, one can find robust state-feedback controllers that do not depend upon the vector of parameters, $\rho(t)$. In case the system depends arbitrarily upon the parameters, a griding on the parameters set is required – see [86, 90]. In that case, the resulting controller is an LPV system itself, as in (2.5), which schedules on the vector of exogenous parameters. Therefore, uncertainty in the measured parameters can result in a deterioration of the performance of the closed-loop and, ultimately, may lead to instability. Moreover, the dynamics matrices in (2.5) are typically obtained by controller interpolation, which does not guarantee, in general, even stability of the closed-loop.

2.3.2 Robust Controllers for LPV Systems

In what follows, it is assumed that the vector of exogenous parameters, $\rho(t)$, is uncertain or cannot be measured. Therefore, the controllers to be designed are assumed LTI and described by

$$\left\{\begin{array}{rcl} \dot{x}_c(t) & = & A_c x(t) + B_c u_c(t) \\ y_c(t) & = & C_c x_c(t) + D_c u_c(t) \end{array}\right. , \tag{2.9}$$

where $x_c(t)$, for each time-instant t, is a vector, with pre-specified length, containing the state of the controller. Notice that $u_c(\cdot)$ corresponds to the output measurements and $y_c(\cdot)$ is the control signal applied to the plant, *i.e.*, $u_c(t) = y(t)$ and $y_c(t) = u(t)$ – see (2.3).

The non-parameter-dependent robust output feedback can be cast as an optimization problem subject to a set of Bilinear Matrix Inequalities (BMIs), as described in [63–65, 86], where one can use the knowledge about the maximum rate of variation of the parameters to reduce the conservatism of the solution. This result is summarized in what follows.

Theorem 2.1 ([86]). *Let T_{zw} denote the closed-loop operator from w to z, and suppose $x(0) = 0$. Then, the $\mathcal{H}_\infty$-norm of T_{zw} is less than γ, that is, $\|T_{zw}\|_\infty < \gamma$, if a continuous symmetric map $P : \Omega \to \mathcal{S}^{n\times n}$ exists such that $P(\rho) > 0$, and*

$$\begin{bmatrix} W(\rho) & P(\rho)B_{cl}(\rho) & \gamma^{-1}C_{cl}^T(\rho) \\ B_{cl}(\rho)P(\rho) & -I & \gamma^{-1}D_{cl}^T(\rho) \\ \gamma^{-1}C_{cl}(\rho) & \gamma^{-1}D_{cl}(\rho) & -I \end{bmatrix} < 0, \tag{2.10}$$

where

$$W(\rho) = P(\rho)A_{cl}(\rho) + A_{cl}^T(\rho)P(\rho) + \sum_{i=1}^{p} \dot{\rho}_i \frac{\partial P}{\partial \rho_i}(\rho),$$

holds for every realization of the parameters vector, $\rho(t) \in \Omega$, and corresponding time-derivatives, $\dot{\rho}(t) \in \Lambda$.

Proof. For the sake of clarity, the proof of Theorem 2.1 is presented, along similar lines as in [86].

Consider the map $V : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}$ defined as

$$V(x,\rho) = x^T P(\rho) x. \tag{2.11}$$

We omit the time and parameter dependence of all maps, for the sake of simplicity. We also omit the subscript cl in maps $A_{cl}(\cdot)$, $B_{cl}(\cdot)$, $C_{cl}(\cdot)$ and $D_{cl}(\cdot)$, for the same reason. Then, for $\rho \in \Omega$, and $\dot{\rho} \in \Lambda$, we have

$$\begin{array}{rcl} \frac{dV}{dt} & = & \left(x^T A^T + w^T B^T\right) Px + x^T P\left(Ax + Bw\right) + x^T \frac{dP}{dt} x \\ & = & x^T \left(A^T P + PA + \frac{dP}{dt}\right) x + w^T B^T Px + xPBw. \end{array} \tag{2.12}$$

Using the Schur complement of the last 2×2-block of the matrix on the left-hand side of (2.10) leads to

$$I - \gamma^{-2} D^T D > 0. \tag{2.13}$$

Also by Schur complement arguments, we have that

$$\begin{array}{l} A^T P + PA + \frac{dP}{dt} + \\ +\gamma^{-2} C^T C + \left(PB + \gamma^{-2} C^T D\right) \left(I - \gamma^{-2} D^T D\right)^{-1} \left(PB + \gamma^{-2} C^T D\right)^T < 0. \end{array} \tag{2.14}$$

Hence, using inequality (2.14) in (2.12), we obtain

$$\begin{array}{l} \frac{dV}{dt} \leq -x^T \left[\left(PB + \gamma^{-2} C^T D\right) \left(I - \gamma^{-2} D^T D\right)^{-1} \left(PB + \gamma^{-2} C^T D\right)^T + \gamma^{-2} C^T C\right] x + \\ +w^T B^T Px + xPBw. \end{array} \tag{2.15}$$

We recall that $z = Cx + Dw$. Therefore,

$$\begin{array}{rcl} \frac{dV}{dt} & \leq & -\| \left(I - \gamma^{-2} D^T D\right)^{\frac{1}{2}} w - \left(I - \gamma^{-2} D^T D\right)^{-\frac{1}{2}} \left(PB\gamma^{-2} C^T D\right)^T x\|^2 - \\ & & -\gamma^{-2} z^T z + d^T d \\ & \leq & -\gamma^{-2} \|z\|^2 + \|d\|^2. \end{array} \tag{2.16}$$

Integrating both sides of (2.16) from $t = 0$ to $t = \infty$, for $x(0) = 0$, and using Lemma 3.3.2 of [90], we get

$$\|z\|^2 \leq \gamma^2 \|w\|^2. \tag{2.17}$$

□

Remark 2.3: If a constant Lyapunov matrix exists, $P(\cdot) \equiv P$, that satisfies (2.10), then the parameters $\rho(\cdot)$ can vary arbitrarily fast. ◇

Remark 2.4: Throughout the remainder of this chapter, we will search for Lyapunov functions with the structure in (2.7), satisfying (2.10). Moreover, as previously stated, the controller is assumed LTI and described by (2.9). Hence, for fixed values of ρ and $\dot{\rho}$, the unknown variables in (2.10) are the controller matrices (A_c, B_c, C_c, D_c), and the Lyapunov function matrices $(P_1, P_2, \cdots, P_m)$. These variables appear in a bilinear fashion, since we have the products between $P(\rho)$ and the closed-loop matrices. Thus, (2.10) is a bilinear matrix inequality (BMI). $\diamond$

Notice that, contrary to [86, 90], our intention is to design controllers that account for parametric (and complex) uncertainty on the plant model. Nevertheless, the same principles can be used in either case, as shown in the sequel. In fact, the key differences between the approach presented in this chapter and the previous related work can be summarized as follows:

- in [86, 90], the authors provide an *LPV controller* synthesis methodology for LPV systems, with output feedback, which means that the controller itself is also parameter-dependent – see Section 2.3.1. This requires the classical "gain scheduling" methodology [95]. Furthermore, it is assumed that these real-valued parameters can be measured on-line. In our approach, the parameters are assumed uncertain and the controller is a linear time-invariant system.
- in [63], the authors also provide a solution to robust full-state feedback control, while in this chapter we are interested in output feedback control.

It is important to stress that (2.10) depends affinely upon $\dot{\rho}(t)$, for each time, t. Suppose that $\dot{\rho}(t)$, for all t, can only take values in a convex polytope, Λ. Then, if (2.10) is satisfied in the vertices of Λ, it can be shown that (2.10) is satisfied for every $\dot{\rho}(\cdot) \in \Lambda$ – see, for instance, [90]. In particular, the following corollary proves that (2.10) need not be satisfied at every point in Λ, but only at the corresponding vertices, in case Λ is a parallelepiped:

Corollary 2.1. *Let T_{zw} denote the closed-loop operator from w to z, and suppose $x(0) = 0$. Further consider that $\dot{\rho} \in \Lambda$ and that Λ is a parallelepiped. Then, the $\mathcal{H}_\infty$-norm of T_{zw} is less than γ, that is, $\|T_{zw}\|_\infty < \gamma$, if a continuous symmetric* *map $P : \Omega \to \mathcal{S}^{n\times n}$ exists such that $P(\rho) > 0$, and*

$$\begin{bmatrix} W(\rho) & P(\rho)B_{cl}(\rho) & \gamma^{-1}C_{cl}^T(\rho) \\ B_{cl}(\rho)P(\rho) & I & \gamma^{-1}D_{cl}^T(\rho) \\ \gamma^{-1}C_{cl}(\rho) & \gamma^{-1}D_{cl}(\rho) & I \end{bmatrix} < 0, \tag{2.18}$$

where

$$W(\rho) = P(\rho)A_{cl}(\rho) + A_{cl}^T(\rho)P(\rho) + \sum_{i=1}^{p} \pm\nu_i \frac{\partial P}{\partial \rho_i}(\rho), \tag{2.19}$$

holds for every realization of the parameters vector, $\rho \in \Omega$.

Remark 2.5: The notation ± in (2.19) indicates that (2.18) has to be satisfied at every vertex of set Λ. ◇

Proof. Since (2.10) is affine in the derivatives of the parameters, $\dot{\rho}$, and given that Λ is a convex polytope, we have that (2.10) is satisfied everywhere everywhere in $\Omega \times \Lambda$ if it satisfied in the vertices of Λ for all $\rho \in \Omega$. Therefore, using Theorem 2.1, we have that $\|T_{zw}\|_\infty < \gamma$. □

Remark 2.6: As previously stated, our main goal is to design LTI controllers for LPV plants. Therefore, imposing such a structure to the controllers guarantees that they are time-invariant, even if the Lyapunov function depends upon the uncertain parameters of the plant. ◇

The optimization problem at hand can be stated as follows: *Find the minimum value of γ such that* (2.10) *holds for every realization of the parameters vector, $\rho \in \Omega$, and corresponding time-derivatives, $\dot{\rho} \in \Lambda$.*

Given that we cannot evaluate (2.18) for every point in Ω (since this is a set that, in general, contains an infinite number of elements), a finite number of representative points in Ω has to be selected. The method adopted in this thesis is to uniformly grid the space of parameters, and then to constraint the optimization program (*i.e.*, the minimization of the $\mathcal{L}_2$-induced norm from the exogenous inputs to the performance outputs) by the BMIs in (2.18) evaluated at the points of that grid. In Section 2.3.6, we derive conditions for the density of this grid that allow us to generalize the results for all the points in Ω. In other words, the conditions in Section 2.3.6

amount to the selection of the density of the grid such that, if (2.18) is satisfied at the points of the aforementioned grid, then it is also satisfied at any other point in Ω.

This type of controllers is thus going to be referred to as LPV/BMI controllers, since they can be used on LPV systems, and are designed using optimization programs subject to BMIs. Notice that, using small gain arguments [77], one can readily guarantee the stability of the closed-loop against complex-valued uncertainties[3], as with the μ-controllers.

2.3.3 BMI Optimization Using General BMI Solvers

As previously stated, in classical LPV control, we assume that we can measure *exactly* the time-varying parameters. Therefore, there are some methods that can be used to transform the controller synthesis problem into an optimization program subject to Linear Matrix Inequalities (LMIs) – see [63, 86, 89]. However, in designing LPV/BMI controllers, we assume that we cannot measure the time-varying parameters. Indeed, it is only required that the *maximum rate-of-change* and magnitude bounds of these parameters are known a priori. Hence, there does not exist a transformation, in the best of our knowledge, that convexifies this problem, for the general case, and thus an optimization program subject to BMIs has to be solved. Still, BMIs have been shown to be non-convex NP-hard optimization problems – see [96]. This obviously implies that large scale problems are most likely to be impossible to solve in a reasonable amount of time. We stress that, nevertheless, these are off-line computations.

However, some recent work on finding suboptimal solutions of BMIs rendered promising results. For instance, [66] and [67] introduce an algorithm that can find a solution better than the one achieved by the D-K iterations, for a specific example. However, a recently updated MATLAB Toolbox referred to as PENBMI (see [97] and [98]) yields even better results for the same example, and requiring reduced

[3] Note that complex-valued uncertainties are very important when the plant has unmodeled dynamics (a situation that always occurs).

computational effort. This suggests that the aforementioned strategy may lead to reasonable results for medium size problems (about 20 states) with an affordable computational burden. However, for plants with a larger number of states, this solution may not be implementable, and hence strategies such as the D,G-K iterations are more suitable [44–48].

2.3.4 BMI Optimization Using LMI Iterations

Although general BMI optimization toolboxes, such as PENBMI, can be used to tackle the problem at hand, similar results can be obtained, at least in the numerical example presented in the sequel, much faster by alternating the optimization process between the Lyapunov function matrices and the controller matrices. This means that, given an initial μ-controller (that can be obtained using the D,G-K iterations), we try to find a Lyapunov function that minimizes some optimality criterium (in the present case, we try to minimize the closed-loop $\mathcal{H}_\infty$-norm). We do this by keeping the controller matrices constant, which means that the problem to be solved can be posed as an optimization procedure subject to LMI constraints. After that, the matrices of the candidate PDLF $(P_1, P_2, \cdots, P_m)$ are fixed, and we try to find the controller that optimizes that same criterium. We repeat this process until no further significant improvements are attained with the iterations, or until a prespecified value of γ is achieved. This algorithm is illustrated in Fig. 2.5.

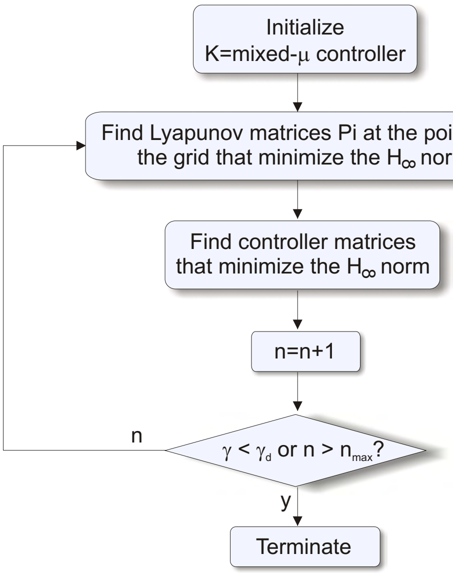

Initialize
K=mixed-μ controller
Find Lyapunov matrices Pi at the points of the grid that minimize the H∞ norm
Find controller matrices that minimize the H∞ norm
n=n+1
γ < γd or n > nmax?
n
y
Terminate

Figure 2.5: LMI-based algorithm to solve optimization problems subject to BMI constraints. The algorithm stops if the value of γ is smaller than a given prespecified value, γ_d, or if the maximum number of iterations is exceeded.

For the sake of completeness, the pseudo-code of this algorithm is presented next:

Algorithm 2: BMI optimization algorithm using LMI iterations.

```
Input: Initial controller ($K_i$), closed-loop desired $\mathcal{H}_\infty$-norm ($\gamma_d$), maximum
       rate of variation of the parameters ($\dot{\rho}_{max}$), number of maximum
       iterations ($n_{max}$)
Output: $K$, $\gamma$
Initialization $K = K_i$, $n = 1$
while ($\gamma > \gamma_d$) && ($n < n_{max}$) do
    • Find the Lyapunov matrices $P_i$ that minimize the closed-loop $\mathcal{H}_\infty$-norm,
    $\gamma$, for LFT of plant with controller $K$, and such that (2.18) is satisfied at
    the points of the grid;
    • Find the matrices of the controller that minimizes the $\mathcal{H}_\infty$-norm of
    closed-loop, $\gamma$;
    • $n = n + 1$
end
```

Although this solution may not achieve the global (or even a local) optimum, it ensures convergence to (at least) a local optimum with respect to P_i alone and with respect to K alone.[4] Furthermore, the controller obtained after each iteration can never be *worse* (in the $\mathcal{L}_2$-induced norm sense) than that obtained in the previous iteration.

2.3.5 D-BMI Iterations

As previously mentioned, the D-scales, obtained with μ-analysis, are very important in reducing the conservatism of the LPV/BMI controllers. Therefore, if we initialize the BMI optimization procedure with the mixed-μ solution, obtained with the D,G-K iterations, it is reasonable to use the D-scales obtained from the synthesis of that controller to reduce the convervatism of the results. However, the resulting LPV/BMI controller may be different from the mixed-μ controller, and hence the D-scales may not be optimal for that controller. Thus, the strategy adopted herein is to iterate between the BMI optimization and the D-scales computation, as described in Fig. 2.6, similarly to what is done in the D-K or the D,G-K iterations.

Once again, for the sake of completeness, the pseudo-code describing the algo-

[4] The D,G-K iterations used in the μ-synthesis are also only guaranteed to converge to a (at least) local optimum, with respect to K alone and with respect to D alone – see [99].

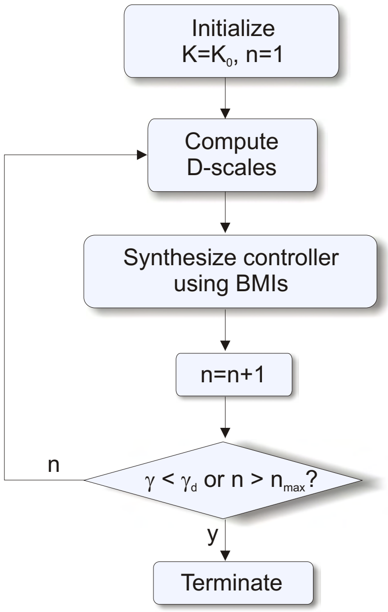


Figure 2.6: D-BMI iterations for designing robust controllers. The algorithm stops if the value of γ is smaller than a given pre-specified value, γ_d, or if the maximum number of iterations is exceeded.

rithm is also included as follows:

Algorithm 3: D-BMI iterations for robust controllers design.

```
Input: Initial controller ($K_i$), closed-loop desired $\mathcal{H}_\infty$-norm ($\gamma_d$), maximum
       rate of variation of the parameters ($\dot{\rho}_{max}$), number of maximum
       iterations ($n_{max}$)
Output: $K$, $\gamma$, D-scales
Initialization $K = K_i$, $n = 1$
while $(\gamma > \gamma_d)$ && $(n < n_{max})$ do
    • Get D-scales, $D$, and closed-loop $\mathcal{H}_\infty$-norm, $\gamma$, for LFT of plant with
    controller $K$;
    • Use $K$ and $\gamma$ to find initial Lyapunov function;
    • Synthesize LPV/BMI controller, $K$, for plant with D-scales and
    compute $\mathcal{H}_\infty$-norm of closed-loop, $\gamma$;
    • $n = n + 1$
end
```

Remark 2.7: As mentioned above, the D-BMI algorithm iterates between the synthesis of D-scales and the synthesis of an $\mathcal{H}_\infty$ controller. Since in both steps of the iteration, the $\mathcal{L}_2$-induced norm of the closed-loop is guaranteed to decrease (or, at least, not to increase), and given that this norm is bounded by below, it is straightforward to conclude that the algorithm converges to a (local) optimum, with respect to D alone and to K alone. $\diamond$

In comparison with the D,G-K iterations, used to synthesize mixed-μ controllers, the main difference is that we eliminate the computation of the G-scales, by designing controllers that are robust against real-valued parametric uncertainty. Moreover, at each iteration, the method used to synthesize the LPV/BMI controller is the optimization program described in the previous subsections, that aims to minimize the $\mathcal{H}_\infty$ norm of the closed-loop, while guaranteeing that (2.18) is satisfied at every point in $\Omega \times \Lambda$.

2.3.6 Gridding the Space of Parameters

One way of avoiding conservative solutions, that usually arise from simplifications of the model of the plant (for instance, approximating the model by one that depends affinely on the parameters), is to grid the parameters set, Ω, and evaluate (2.10) at the points of that grid – see Fig. 2.7. This approach is also used in "classical" LPV control, as described in [90]. Using this method reduces the number of BMI constraints that have to be satisfied, since now we only need to compute (2.10) for a finite set of points.

However, if (2.10) is satisfied in the points of the grid, it does not necessarily mean that it is satisfied in every point in Ω. In this section, we present sufficient conditions that guarantee that (2.10) is satisfied for every legal realization of the pair $(\rho(\cdot),\ \dot{\rho}(\cdot))$ if it is satisfied for a finite number of points in $\Omega \times \Lambda$. In summary, the conditions in the sequel amount for the selection of the density of the grid in Ω.

Theorem 2.2. *Suppose that we grid Ω by $L_1 \times \cdots \times L_p$ points, uniformly spaced in each dimension (as depicted in Figure 2.7). Denote the set of points of the grid by*

$$\Upsilon = \left\{(\rho_1, \cdots, \rho_p) : \rho_1 \in \{\rho_{1,1}, \cdots, \rho_{1,L_1}\}, \cdots, \rho_p \in \{\rho_{p,1}, \cdots, \rho_{p,L_p}\}\right\}.$$

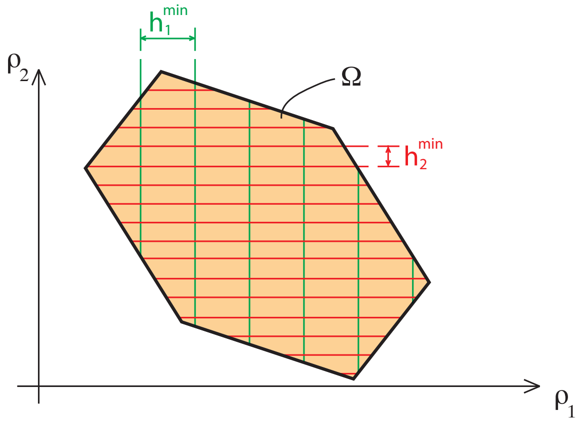


Figure 2.7: Grid for a polytope, Ω, in $\mathbb{R}^2$.

Let h_j^{min} be defined such that, for any two points $\bar{\rho}, \tilde{\rho} \in \Upsilon$, we have $\|\bar{\rho}_j - \tilde{\rho}_j\| < h_j^{min}$, for every $j = \{1, \cdots, p\}$. Further suppose that, for some $\delta < 0$ and $T \geq \max_i \|P_i\|$,

$$\begin{aligned} h_j^{min} &< \tfrac{|\delta|}{m}\left[2T\sum_{i=1}^{m}\max_{\rho\in\Omega}\left\|\frac{\partial\left(\xi_i A_{cl}^T\right)}{\partial\rho_j}\right\|_F + \nu_j T\sum_{t=1}^{p}\sum_{i=1}^{m}\max_{\rho\in\Omega}\left|\frac{\partial^2\xi_i}{\partial\rho_j\rho_t}\right| + \right. \\ &+ \left. 2T\sum_{i=1}^{m}\max_{\rho\in\Omega}\left\|\frac{\partial\left(\xi_i B_{cl}\right)}{\partial\rho_j}\right\|_F + 2\gamma^{-1}\max_{\rho\in\Omega}\left\|\frac{\partial C_{cl}}{\partial\rho_j}\right\|_F + 2\gamma^{-1}\max_{\rho\in\Omega}\left\|\frac{\partial D_{cl}}{\partial\rho_j}\right\|_F\right]^{-1}, \end{aligned} \tag{2.20}$$

for every $j = \{1, \cdots, p\}$, and that (2.18) *is satisfied for all $\rho \in \Upsilon$. Then* (2.18) *is satisfied for all $\rho \in \Omega$.*

Proof. We follow closely the derivations in [90]. Define

$$F(\rho) = \begin{bmatrix} W(\rho) & P(\rho)B_{\text{cl}}(\rho) & \gamma^{-1}C_{\text{cl}}^T(\rho) \\ B_{\text{cl}}(\rho)P(\rho) & -I & \gamma^{-1}D_{\text{cl}}^T(\rho) \\ \gamma^{-1}C_{\text{cl}}(\rho) & \gamma^{-1}D_{\text{cl}}(\rho) & -I \end{bmatrix},$$

where $W(\rho)$ is as in (2.19). In what follows, we omit the *cl* subscript, for the sake of simplicity. Notice that, if (2.18) is satisfied for $\rho = \tilde{\rho}$, then there exists $\delta < 0$ such that

$$F(\tilde{\rho}) < \delta.$$

We are interested in deriving conditions that guarantee that if $\|\bar{\rho}_i - \tilde{\rho}_i\| < h_i^{\text{min}}$, for every $i = \{1, \cdots, p\}$ and where $\bar{\rho}, \tilde{\rho} \in \Omega$ are two arbitrary points in Ω, then

$F(\tilde{\rho}) < \delta$ implies $F(\bar{\rho}) < 0$. Therefore, consider that indeed $\|\bar{\rho}_i - \tilde{\rho}_i\| < h_i^{\min}$ and that $\max_{t\in\mathbb{R}} |\dot{\rho}_i(t)| = \nu_i$. Then, using (2.7) and (2.18),

$$\begin{aligned}
&\|F(\bar{\rho}) - F(\tilde{\rho})\|_F \leq 2\sum_{i=1}^{m} \left\|P_i\left(\xi_i A^T(\bar{\rho}) - \xi_i A^T(\tilde{\rho})\right)\right\|_F + \\
&+\sum_{j=1}^{p}\left[\nu_j \sum_{i=1}^{m}\left\|P_i\left(\frac{\partial \xi_i}{\partial \rho_j}(\bar{\rho}) - \frac{\partial \xi_i}{\partial \rho_j}(\tilde{\rho})\right)\right\|_F\right] + 2\sum_{i=1}^{m}\|P_i\left(\xi_i B(\bar{\rho}) - \xi_i B(\tilde{\rho})\right)\|_F + \\
&+2\gamma^{-1}\|C(\bar{\rho}) - C(\tilde{\rho})\|_F + 2\gamma^{-1}\|D(\bar{\rho}) - D(\tilde{\rho})\|_F \\
&\leq \sum_{i=1}^{m}|\rho_j - \rho_{j,k_j}|\left[2\sum_{i=1}^{m}\left\|P_i \frac{\partial\left(\xi_i A^T\right)}{\partial \rho_j}(\xi^{1ij})\right\|_F + \nu_j\sum_{t=1}^{p}\sum_{i=1}^{m}\left\|P_i\frac{\partial^2 \xi_i}{\partial \rho_j \rho_t}(\xi^{2ijt})\right\|_F + \right. \\
&\left. + 2\sum_{i=1}^{m}\left\|P_i\frac{\partial\left(\xi_i B\right)}{\partial \rho_j}(\xi^{3ij})\right\|_F + 2\gamma^{-1}\left\|\frac{\partial C}{\partial \rho_j}(\xi^{4j})\right\|_F + 2\gamma^{-1}\left\|\frac{\partial D}{\partial \rho_j}(\xi^{5j})\right\|_F\right],
\end{aligned}$$

for some ξ^{1ij}, ξ^{2ijt}, ξ^{3ij}, ξ^{4j}, $\xi^{5j} \in [\rho_{1,k_1},\, \rho_{1,k_1+1}] \times \cdots \times [\rho_{p,k_p},\, \rho_{p,k_p+p}]$

$$\begin{aligned}
&\leq \sum_{i=1}^{m} h_j^{\min}\left[2T\sum_{i=1}^{m}\max_{\rho\in\Omega}\left\|\frac{\partial\left(\xi_i A^T\right)}{\partial \rho_j}\right\|_F + \nu_j T\sum_{t=1}^{p}\sum_{i=1}^{m}\max_{\rho\in\Omega}\left|\frac{\partial^2 \xi_i}{\partial \rho_j \rho_t}\right| + \right. \\
&\left. + 2T\sum_{i=1}^{m}\max_{\rho\in\Omega}\left\|\frac{\partial\left(\xi_i B\right)}{\partial \rho_j}\right\|_F + 2\gamma^{-1}\max_{\rho\in\Omega}\left\|\frac{\partial C}{\partial \rho_j}\right\|_F + 2\gamma^{-1}\max_{\rho\in\Omega}\left\|\frac{\partial D}{\partial \rho_j}\right\|_F\right]
\end{aligned}$$

Thus, if

$$\begin{aligned}
&h_j^{\min} < \tfrac{|\delta|}{m}\left[2T\sum_{i=1}^{m}\max_{\rho\in\Omega}\left\|\frac{\partial\left(\xi_i A_{\mathrm{cl}}^T\right)}{\partial \rho_j}\right\|_F + \nu_j T\sum_{t=1}^{p}\sum_{i=1}^{m}\max_{\rho\in\Omega}\left|\frac{\partial^2 \xi_i}{\partial \rho_j \rho_t}\right| + \right. \\
&\left. + 2T\sum_{i=1}^{m}\max_{\rho\in\Omega}\left\|\frac{\partial\left(\xi_i B_{\mathrm{cl}}\right)}{\partial \rho_j}\right\|_F + 2\gamma^{-1}\max_{\rho\in\Omega}\left\|\frac{\partial C_{\mathrm{cl}}}{\partial \rho_j}\right\|_F + 2\gamma^{-1}\max_{\rho\in\Omega}\left\|\frac{\partial D_{\mathrm{cl}}}{\partial \rho_j}\right\|_F\right]^{-1},
\end{aligned} \tag{2.21}$$

and (2.18) is satisfied at the points of the grid, then (2.18) is satisfied everywhere in Ω. □

Remark 2.8: The relevance of this result is twofold. On the one hand, it is important from a theoretical point of view, since it allows us to provide stability and performance specifications in a region, based upon the evaluation of an optimization procedure constrained by a finite set of BMIs. On the other hand, it is helpful, from a practical point of view, to evaluate whether or not the density of a given grid is enough. ◇

Notice that Theorem 2.2 should be used to test if a given grid is sufficiently dense, after (2.18) has been checked at every point of the grid. This means that the steps to be taken are as follows:

Algorithm 4: Gridding the space of parameters

```
Initialization Create an arbitrary grid for $\Omega$, and use Theorem 2.2 to check
    if the grid is sufficiently dense, for a given value of $\gamma$
while Density of the grid not enough do
    Increase the density of the grid, and use Theorem 2.2 to check if the grid
    is sufficiently dense for the given value of $\gamma$
end
```

As an example, consider a plant that depends affinely upon one uncertain parameter, ρ, i.e.,

$$\begin{cases} A(\rho) &= A_0 + A_1\rho \\ B(\rho) &= B_0 + B_1\rho \\ L(\rho) &= L_0 + L_1\rho \\ C(\rho) &= C_0 + C_1\rho \\ C_z(\rho) &= C_{z0} + C_{z1}\rho \\ D(\rho) &= D_0 + D_1\rho \\ N(\rho) &= N_0 + N_1\rho. \end{cases}$$

In this case, the closed-loop system can be written as

$$\begin{cases} \begin{bmatrix} \dot{x} \\ \dot{x}_c \end{bmatrix} = \begin{bmatrix} A + BD_cC & BC_c \\ B_cC & A_c \end{bmatrix} \begin{bmatrix} x \\ x_c \end{bmatrix} + \begin{bmatrix} L & BD_cN \\ 0 & B_cN \end{bmatrix} \begin{bmatrix} d \\ n \end{bmatrix} \\ z = \begin{bmatrix} C_z & 0 \end{bmatrix} \begin{bmatrix} x \\ x_c \end{bmatrix} \end{cases},$$

where

$$\begin{cases} \dot{x}_c &= A_cx_c + B_cy \\ u &= C_cx_c + D_cu_c \end{cases},$$

describes the controller dynamics. For the sake of simplicity, assume $D_c = 0$. Further suppose that the PDLF can be written as

$$V = x^T P(\rho)x,$$

where $P(\rho) = P_0 + \rho P_1$. Then, one would get

$$\xi_0 = 1,\ \xi_1 = \rho.$$

Thus, the derivatives in (2.20) are given by

$$\begin{aligned}
\tfrac{\partial}{\partial\rho}A_{\mathrm{cl}} &= \begin{bmatrix} A_1 & B_1C_c \\ B_cC_1 & 0 \end{bmatrix}, & \tfrac{\partial^2}{\partial\rho^2}1 &= 0, \\
\tfrac{\partial}{\partial\rho}B_{\mathrm{cl}} &= \begin{bmatrix} L_1 & 0 \\ 0 & N_1 \end{bmatrix}, & \tfrac{\partial}{\partial\rho}C_{\mathrm{cl}} &= \begin{bmatrix} C_{z1} & 0 \end{bmatrix}, \\
\tfrac{\partial}{\partial\rho}\left(\rho A_{\mathrm{cl}}\right) &= \begin{bmatrix} A_0 + 2\rho A_1 & B_0C_c + 2\rho B_1C_c \\ B_cC_0 + 2\rho B_cC_1 & A_c \end{bmatrix}, & \tfrac{\partial}{\partial\rho}D_{\mathrm{cl}} &= 0, \\
\tfrac{\partial}{\partial\rho}\left(\rho B_{\mathrm{cl}}\right) &= \begin{bmatrix} L_0 + 2\rho L_1 & 0 \\ 0 & N_0 + 2\rho N_1 \end{bmatrix}, & \tfrac{\partial^2}{\partial\rho^2}\rho &= 0.
\end{aligned}$$

Using these derivatives and given that the parameter range and maximum rate of variation are known, an upper bound for $h^{\min}$ can be computed using Theorem 2.2. For instance, let us suppose that we used a grid with just two points, *i.e.*, the grid is simply $\Upsilon = \{\rho_{\min},\ \rho_{\max}\} := \{0,\ 1\}$. Then, the candidate value for $h^{\min}$ would be $\hat{h}^{\min} = \rho_{\max} - \rho_{\min} = 1$. Further suppose that we solve the optimization problem of minimizing the $\mathcal{L}_2$-induced norm from the exogenous inputs to the performance outputs, while (2.18) is satisfied at the points of the grid, υ, and that this results in $\delta = 1$, $T = 0.1$ and $\gamma = 1$. If we consider that

$$\begin{aligned}
\left\|\tfrac{\partial}{\partial\rho}A_{\mathrm{cl}}\right\|_F &= 0.1, & \left\|\tfrac{\partial}{\partial\rho}B_{\mathrm{cl}}\right\|_F &= 0.1, \\
\max_{\rho\in[0,\,1]}\left\|\frac{\partial}{\partial\rho}\left(\rho A_{\mathrm{cl}}\right)\right\|_F &= 0.1, & \max_{\rho\in[0,\,1]}\left\|\frac{\partial}{\partial\rho}\left(\rho B_{\mathrm{cl}}\right)\right\|_F &= 0.1, \\
\left\|\tfrac{\partial}{\partial\rho}C_{\mathrm{cl}}\right\|_F &= 0.1,
\end{aligned}$$

then the result obtained is

$$h^{\min} = 1.56 > \hat{h}^{\min}.$$

Thus, the density of the grid for this example is sufficient, which means that the resulting controller guarantees a level of performance of $\gamma = 1$ for every realization of the plant with $\rho \in [0,\, 1]$. As a final remark regarding this example, bounds for the terms like

$$\max_{\rho \in [0,\, 1]} \left\| \frac{\partial}{\partial \rho} \left(\rho A_{\mathrm{cl}} \right) \right\|_F$$

can be obtained by noting that

$$\begin{aligned}
\max_{\rho \in [0,\, 1]} \left\| \frac{\partial}{\partial \rho} \left(\rho A_{\mathrm{cl}} \right) \right\|_F &\leq \left\| \begin{bmatrix} A_0 & B_0 C_c \\ B_c C_0 & A_c \end{bmatrix} \right\|_F + \max_{\rho \in [0,\, 1]} \left\| \rho \begin{bmatrix} 2A_1 & 2B_1 C_c \\ 2B_c C_1 & 0 \end{bmatrix} \right\|_F \\
&= \left\| \begin{bmatrix} A_0 & B_0 C_c \\ B_c C_0 & A_c \end{bmatrix} \right\|_F + \left\| \begin{bmatrix} 2A_1 & 2B_1 C_c \\ 2B_c C_1 & 0 \end{bmatrix} \right\|_F .
\end{aligned}$$

Remark 2.9: Notice that solving this optimization problem for a general time-varying LPV plant, only provides sufficient conditions for stability (and performance), since a quadratic Lyapunov function may not exist for that specific plant, although it might be stable – see [94]. ◇

Remark 2.10: The main advantage of the LPV/BMI approach, is that, if properly designed, these LTI controllers can be less sensitive to the variations of the parameters, and, therefore, the solution obtained is more robust in that sense. It should be noticed that, contrary to what happens with the mixed-μ controllers, these controllers can guarantee stability and performance of the closed-loop for time-varying parameters. We stress that this property is instrumental, for instance, in adaptive control, as previously mentioned. Furthermore, this method can fulfill other design requirements, such as closed-loop pole placement [63, 92] and mixed $\mathcal{H}_2/\mathcal{H}_\infty$ objectives [63]. ◇

Table 2.1 summarizes the design procedures and the bounds on $\dot{\rho}$ for the different parameters dependence of the plant and structure of the quadratic Lyapunov function.

Remark 2.11: Consider that the plant dynamics depend affinely on the vector of parameters, ρ. Further consider that the Lyapunov function can be written as $V(x) = x^T P x$, with P a constant matrix. Then, inequality (2.10) also becomes affine in the vector of parameters, ρ. Therefore, if Ω is a convex polytope, then (2.10) is satisfied everywhere in Ω if it is satisfied in the vertices of this polytope -

	Constant Lyapunov Function $V(x) = x^T P x$	Parameter-Dependent Lyapunov Function $V(x, \rho) = x^T P(\rho) x$
Plant depends affinely upon the parameters	$\dot{\rho}$ may be unbounded. BMI (2.10) needs only to be checked at the vertices of the polytope, Ω (see Remark 2.11).	Bounded $\dot{\rho}$. Gridding may be required.
Plant depends arbitrarily upon the parameters	$\dot{\rho}$ may be unbounded. Gridding may be required.	Bounded $\dot{\rho}$. Gridding may be required.

Table 2.1: Design procedures and bounds on $\dot{\rho}(\cdot)$ for the different parametric dependence of the plant and structure of the quadratic Lyapunov function

see [63] for further details. Nevertheless, considering the same Lyapunov function for every point in Ω is, in general, a restrictive assumption. $\diamond$

The comparison between mixed-μ and LPV/BMI controllers is summarized in Table 2.2.

It is important to stress that the LPV/BMI solution provides stability and performance guarantees for time-varying and time-invariant plants. Moreover, we show next that, at least in the case presented herein, the LPV/BMI solution can also provide better performance guarantees for time-invariant plants than those provided by μ-controllers.

In summary, the conservatism of the LPV/BMI controllers design method is due to:

a) the (possibly) sub-optimal solution of the BMI solver;

b) the use of basis functions to represent the Lyapunov matrix (although this in practice is not an issue, since we can use as many basis function as needed);

c) the use of a quadratic PDLF (although this in practice may also not be a shortcoming, since we can have state-dependent parameters, and hence have higher

	mixed-μ	LPV/BMI
Stability and performance guarantees for LTI plants	Yes	Yes, and at least as good those provided by μ
Stability and performance guarantees for uncertain LTI plants	Yes	Yes, and at least as good those provided by the μ
Stability and performance guarantees for time-varying plants	No	Yes
Stability and performance guarantees for uncertain TV plants	No	Yes
Plant parameters dependence	LFT representable	LPV representable
Computational complexity	Low	High

Table 2.2: Mixed-μ vs LPV/BMI Controllers

order dependence of the Lyapunov function upon the states).

2.4 Mixed-μ vs LPV/BMI Controllers

The applicability of the LPV/BMI controllers is going to be illustrated using a Mass-Spring-Dashpot plant (MSD) with two real parametric uncertainties and a complex-valued one. Consider the MSD plant depicted in Fig. 2.8.

The unknown spring constant is bounded by

$$k_1 \in [0.25,\ 1.75] \text{ N/m},$$

the unkown mass is bounded by

$$m \in [0.2,\ 3.8] \text{ kg},$$

and the unknown input time-delay is bounded by

$$0 < \tau < 0.05 \text{ s},$$

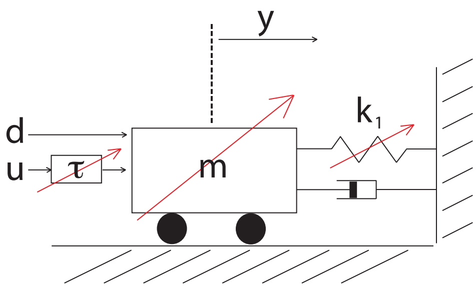


Figure 2.8: MSD system with uncertain spring constant, k_1, and mass, m. The disturbances are denoted by $d(t)$. $u(t)$ is the control input and $y(t)$ is the system output. τ is an uncertain time-delay bounded by $0 < \tau < 0.05$ s.

	LFB Design	HFB Design
α	0.1 rad/s	3 rad/s
b	0.1 N/(m/s)	0.05 N/(m/s)

Table 2.3: Specifications for the LFB and HFB designs

acting as a surrogate for unmodeled dynamics. The friction coefficient is given by

$$b = 0.1 \text{ Ns}^2/\text{m}.$$

Figure 2.9 shows the block diagram used in the design process. The disturbances, $d(t)$, are generated by driving a low-pass filter with transfer function (2.22) with continuous-time white noise $\xi(t)$ with zero mean and unit intensity.

$$W_d(s) = \frac{\alpha}{s+\alpha}, \quad \alpha = 0.1 \text{ rad/s} \tag{2.22}$$

Two design scenarios are considered, namely one with a low-frequency bandwidth (LFB) specification for the disturbances, and another with a high-frequency bandwidth (HFB). For the LFB design, we consider $\alpha = 0.1$ rad/s, and for the HFB, we define $\alpha = 3$ rad/s, as summarized in Table 2.3.

We assume that the sensor noise can be modeled by continuous-time white noise $\theta(t)$, with zero mean and intensity $W_n = 0.001$. The performance filter, W_p, is described by

$$W_p(s) = A_p \frac{\alpha}{s+\alpha}, \tag{2.23}$$

where $\alpha = 0.1$ rad/s and A_p is called the performance gain and is determined using Algorithm 1. The unknown input time-delay, τ, is modeled by a complex-valued uncertainty bounded by the magnitude of

$$W_{un}(s) = \frac{2.1s}{s+40}$$

in the frequency domain. Finally, the control input, $u(t)$, is penalized by

$$W_u^{\mathrm{LFB}}(s) = \frac{10(s+10)}{s+10^3},$$

for the LFB design, and by

$$W_u^{\mathrm{HFB}}(s) = \frac{10(s+30)}{s+3\times 10^3},$$

for the HFB design.

In the MSD system, the uncertain spring has a potential energy of

$$(1/2)k_1(\Delta x)^2.$$

If the spring parameter k_1 is constant and bounded, we have an upper and a lower bound on the potential energy uncertainty stored in the spring. If, however, the parameter k_1 is time-varying, then the spring potential energy changes exogenously. When the spring parameter k_1 is increasing, there is an exogenous addition to the potential energy; when k_1 is decreasing, there is an exogenous decrease in the stored spring potential energy. A similar line of thought can be applied to the uncertain mass, m, whose kinetic energy is $(1/2)m(\dot{x})^2$.

2.4.1 Design of the Mixed-μ Controller

Assume that the spring stiffness, k_1, and that the mass m are constant but unknown, subject to their magnitude constraints. The framework used to synthesize the mixed-μ controller for maximum disturbance-rejection is depicted in Fig. 2.9, which can be re-arranged as in Fig. 2.4 – see [77, 78] for further details on designing mixed-μ controllers. The transfer function of the plant is defined such that the position of the mass, $x(t)$, satisfies

$$\ddot{x}(t) = \frac{1}{m(t)}\left[d(t) + u(t) - k_1(t)x(t) - b\dot{x}(t)\right], \tag{2.24}$$

where b is the friction coefficient. For the LFB design, we consider $b = 0.1$ N/(m/s), while for the HFB design we assume $b = 0.05$ N/(m/s) – see Table 2.3.

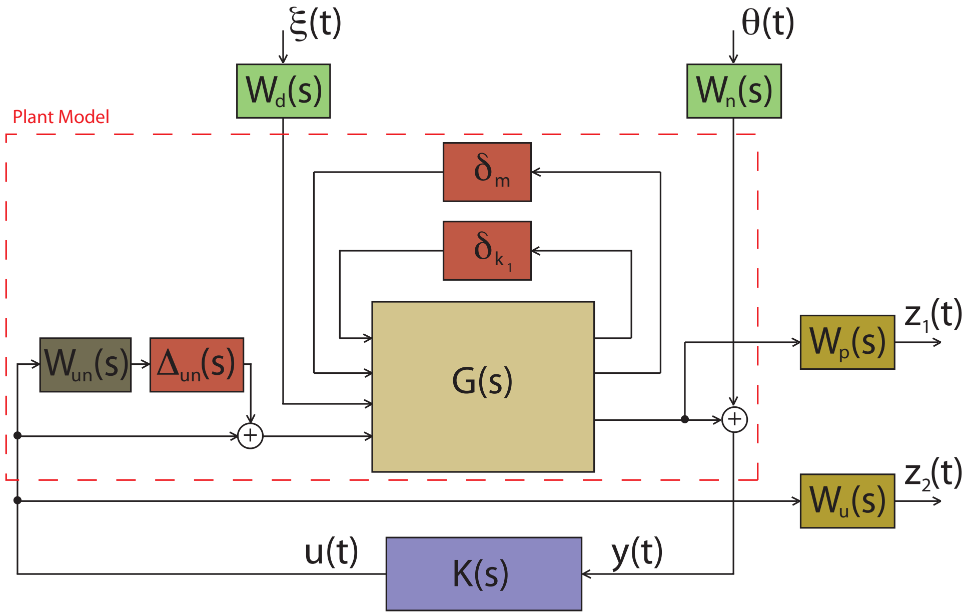


Figure 2.9: Setup used to synthesize the mixed-μ controller. δ_{k_1} and δ_m represent the uncertain spring constant and uncertain mass, respectively. $\Delta_{un}(s)$ is used to model the input time-delay. $z_1(t)$ and $z_2(t)$ are the performance outputs. The goal is maximum disturbance rejection.

We use Algorithm 1 to obtain the *maximum value of* A_p *that guarantees robust performance* as $A_p = A_p^{\text{mixed}-\mu} := 750$. This means that, for values of A_p larger than 750, there may exist a structured Δ-block in Fig. 2.4, with $\|\Delta\| \leq 1$, that destabilizes the closed-loop system, if we use the mixed-μ controller. Therefore, $A_p^{\text{mixed}-\mu}$ can be interpreted as a measure of the guaranteed $\mathcal{H}_\infty$ performance of the mixed-μ controller for the constant parameter case.

Remark 2.12: It should be noticed that the same technique as in [58, 59] is used to maximize the performance of the closed-loop system. By maximizing the value of A_p, we are able to find the controller that provides the best performance while guaranteeing the stability of the uncertain closed-loop plant. This is, indeed, the appropriate method to compare the robust performance levels achieved by two different controllers. $\diamond$

The D,G-K iterations treat the (real-valued) parametric uncertainties as complex-valued uncertainties. Therefore, the resulting controller guarantees stability and performance not only for all the possible realizations of the plant, but also for realizations that are not described by the model of the plant in (2.24) with a real-valued k_1.

2.4.2 LFB Design: Analysis for LTI Plants

The mixed-μ synthesis assumes that the plant to be controlled is time-invariant. Therefore, suppose that indeed $\dot{k}_1 = 0$ and $\dot{m} = 0$. We synthesize an LPV/BMI controller using the method described in Section 2.3. The BMI optimization was carried out using the aforementioned weights, except for the performance gain A_p in (2.23). We define $A_p^{\mathrm{LPV/BMI}}$ as the maximum value of A_p for which the LPV/BMI controller guarantees stability (and performance) for any structured Δ-block in Fig. 2.4, with $\|\Delta\| \leq 1$. Once again, this value is obtained using Algorithm 1.

Figure 2.10 depicts the poles of the open-loop system for several values of k_1 and m, and for $b = 0.1$ N/(m/s). The poles near the imaginary axis are, in general, and from a physical point of view, the ones that pose sharper constraints on the achievable performance. Representing k_1 by means of a complex-valued uncertainty, as in the D,G-K iterations, adds admissible realizations of the plant with poles that are closer to the imaginary axis. In fact, it may even happen that, for some complex values of k_1, the poles of the plant are located in the right half-plane of $\mathbb{C}$. Hence, the performance obtained with the mixed-μ controller can be severely deteriorated, since this type of controller accounts for models of the plant which are not realizable in practice.

The D-BMI iterations were initialized with the matrices of the dynamics of the μ-compensator. Notice that the initialization of the algorithm is very important, since we are dealing with local optimization procedures. Each of these optimization problems subject to BMI constraints was solved using the method presented in Section 2.3.4.

The Bode plots of the controllers transfer functions for the LTI case are depicted

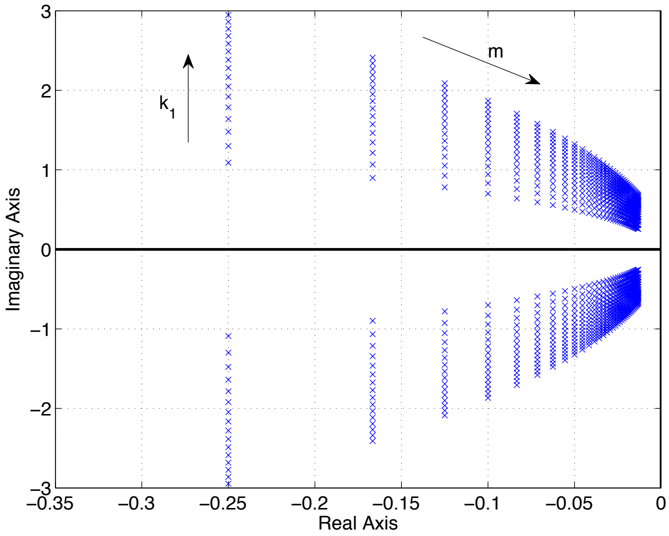


Figure 2.10: Poles of the open-loop plant with $b = 0.1$ N/(m/s).

in Fig. 2.11. Although the controllers are similar, the μ-controller has smaller low-frequency gain, which explains the benefits in terms of performance of the LPV/BMI controller for the present case. For higher frequencies, the Bode plots of the two controllers are very similar. This can be explained by the fact that both controllers use similar models for those frequencies. In particular, the unmodeled high frequency dynamics are handled in the same manner (*i.e.*, using D-scales) by the two designs. Notice that, at those frequencies, these (uncertain) dynamics are predominant.

In terms of guaranteed $\mathcal{H}_\infty$ performance, we obtained

$$A_p^{\text{mixed}-\mu} = 750,$$

$$A_p^{\text{LPV/BMI}} = 960,$$

using the so-called μ-analysis [46, 48, 82].

Figure 2.12 depicts the sensitivity gain for the two controllers, using a realization of the plant with $k_1 = 0.25$ N/m and $m = 0.5$ kg. We can observe that the sensitivity for the LPV/BMI controller is smaller than that obtained with the mixed-μ controller, for the low-frequencies, which is in accordance with the performance improvements observed with the μ-analysis.

The behaviors of both controllers are assessed in simulation, using $k_1 = 0.25$ N/m and $m = 0.5$ kg, and considering no input time-delay. Consider that the

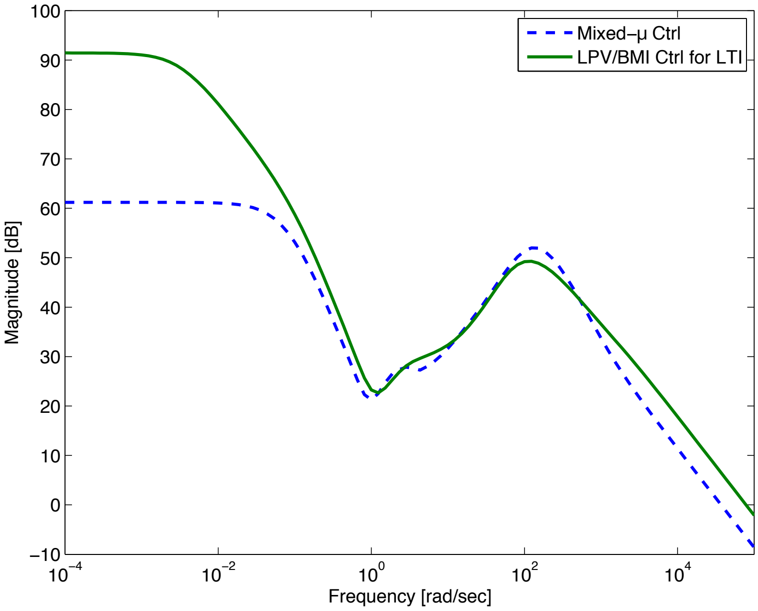


Figure 2.11: LFB Design: Bode plots for the mixed-μ and LPV/BMI controllers, for $\dot{k}_1 = 0$ and $\dot{m} = 0$.

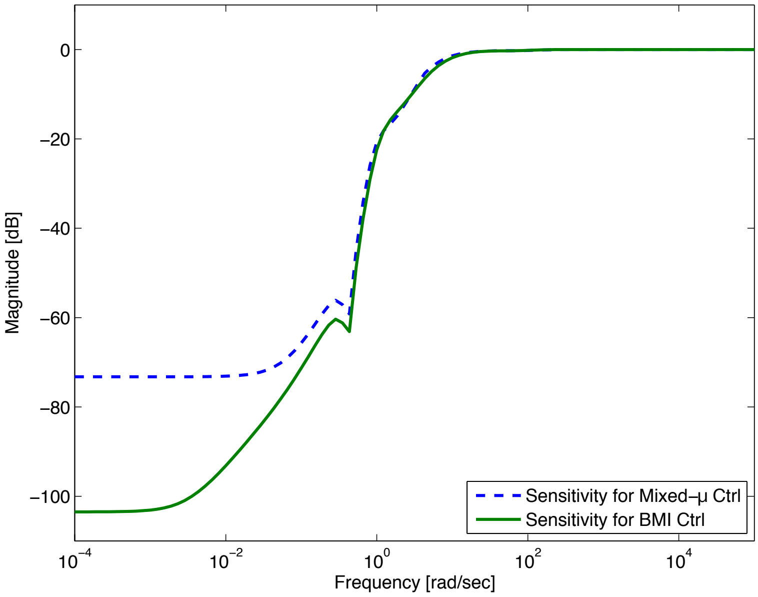


Figure 2.12: LFB Design: Bode plots of the sensitivity gain for the LPV/BMI and mixed-μ controllers, for $\dot{k}_1 = 0$, $\dot{m} = 0$, $k_1 = 0.25$ N/m and $m = 0.5$ kg.

disturbances are as shown in Fig. 2.13. These low-frequency disturbances clearly illustrate the differences between the controllers, as shown in Fig. 2.14. Consider the RMS of the performance output, $z_1(\cdot)$, of the closed-loop, using each controller. For this case, we get

$$\begin{aligned} \text{RMS for LPV/BMI Controller} &= 0.0233 \text{ m}, \\ \text{RMS for Mixed-}\mu \text{ Controller} &= 0.0495 \text{ m}. \end{aligned}$$

Therefore, the RMS of the performance output, $z_1(\cdot)$, of the closed-loop with the LPV/BMI controller is nearly 47% of that obtained with the mixed-μ controller.

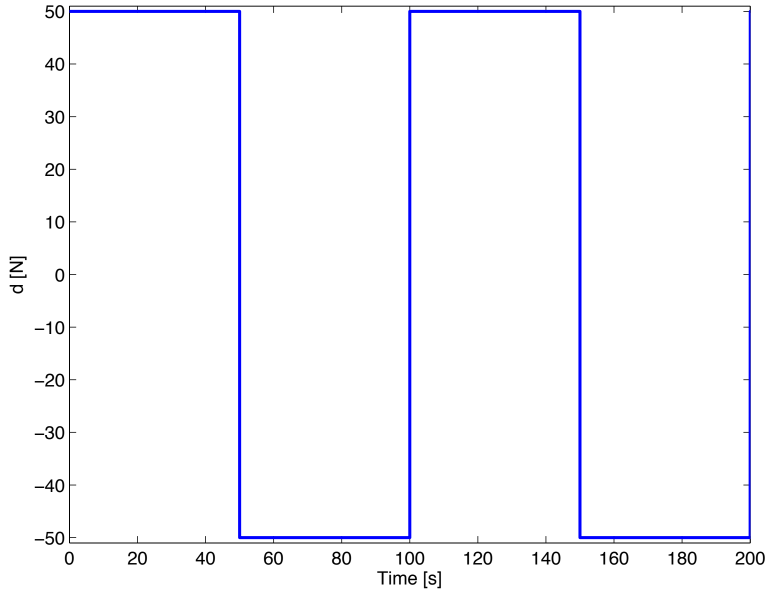


Figure 2.13: Low-frequency exogenous disturbances, $d(t)$.

We stress that the design goal of both the mixed-μ and the LPV/BMI controllers is to guarantee robust $\mathcal{H}_\infty$ performance of the closed-loop. Nevertheless, for time-simulations, the RMS is significantly easier to compute, and provides some insight regarding the behavior of the controllers. This explains why we make such a comparison, despite of the fact that these values should only be taken into account as a superficial evaluation of the controllers.

Remark 2.13: Suppose that we use the performance gain $A_p = 960$ in (2.23). (We recall that this is the value of A_p used to design the LPV/BMI controller. For the mixed-μ design, the maximum value of the performance gain yielding $\mu < 1$ was

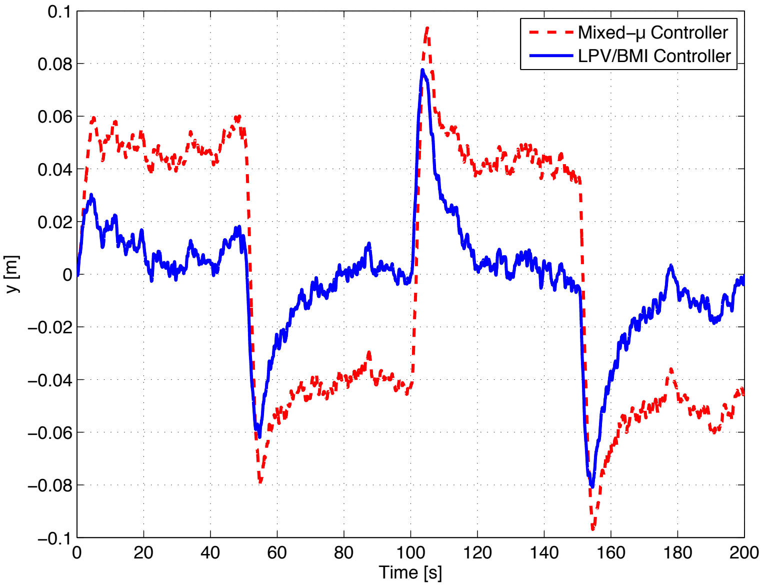


Figure 2.14: LFB Design: Output of the closed-loop for the LPV/BMI and mixed-μ controllers, for $\dot{k}_1 = 0$, $\dot{m} = 0$, $k_1 = 0.25$ N/m and $m = 0.5$ kg.

$A_p^{\text{mixed-}\mu} = 750$.) If one computes the value of μ for the closed-loop of the plant with the mixed-μ controller and with $A_p = 960$, one would get

$$\mu_{\text{mixed-}\mu} \approx 1.27$$

If we use the LPV/BMI controller, we get

$$\mu_{\text{LPV/BMI}} \approx 0.99,$$

which is in accordance to the fact that the closed-loop system is stable for any legal realization of the plant. $\diamond$

Therefore, for time-invariant plants, the level of $\mathcal{H}_\infty$ closed-loop performance guaranteed by the mixed-μ controller is only about 77% of that guaranteed by the LPV/BMI controller, in the LFB design. Furthermore, contrary to what happens with the mixed-μ controller, when we use the LPV/BMI controller, *the stability and performance of the closed-loop is guaranteed even for time-varying plants.*

Remark 2.14: As a technical detail, the PDLF used for the LPV/BMI controller was $V = x^T P\,(k_1, m)\,x$, with

$$P\,(k_1, m) = P_1 + k_1 P_2 + m P_3 + k_1 m P_4 + k_1^2 P_5 + m^2 P_6,$$

although higher order terms could also have been used. $\diamond$

Although, in this case, both controllers (mixed-μ and LPV/BMI) are pursuing the same disturbance-rejection objectives, the results are clearly different. As previously stated, there is a conservative step in the D,G-K iterations that can be avoided with the LPV/BMI design: parametric uncertainties are treated in a non-conservative manner, *i.e.*, the (possibly) time-varying real-valued parameters enter the dynamics of an LPV plant in a natural way, and hence do not have to be approximated by complex-valued uncertainties. Therefore, in spite of also being solved using local optimization procedures, the LPV/BMI synthesis potentially leads to results that are less conservative than the mixed-μ design.

As a caveat, the amount of time required by the (off-line) computations of the LPV/BMI design is almost 100 times larger than the amount of time required to design the corresponding mixed-μ controller.

2.4.3 LFB Design: Analysis for Time-Varying Plants

The BMI optimization was carried out using the same weights, and for several values of the bounds on the rate of variation of the parameters. The D-scales obtained from the mixed-μ design were also used in order to reduce the conservatism of the solution. The μ-controller matrices were used to initialize the algorithm. Figure 2.22 depicts the value of the $\mathcal{H}_\infty$-norm of the closed-loop from the exogenous inputs to the performance output, $z_1(\cdot)$, using LPV/BMI controllers, γ, versus the maximum rate of variation of the mass, m, and of the spring stiffness, k_1. As expected, γ increases when we allow m and k_1 to change faster. Intuitively, this means that increasing the rate by which the uncertain energy of the system can change, we reduce the performance guarantees. The decrease of the performance guarantees with the increase of the maximum rate of variation of the uncertain parameters can also be explained by the fact that we are providing robust-performance guarantees for a much wider family of plant models.

Remark 2.15: The mixed-μ controller *is always a valid controller for* $\dot{\rho}(\cdot) \equiv 0$ – see [63]. Therefore, for $\dot{\rho}(\cdot) \equiv 0$, this controller can be used to initialize the algorithm

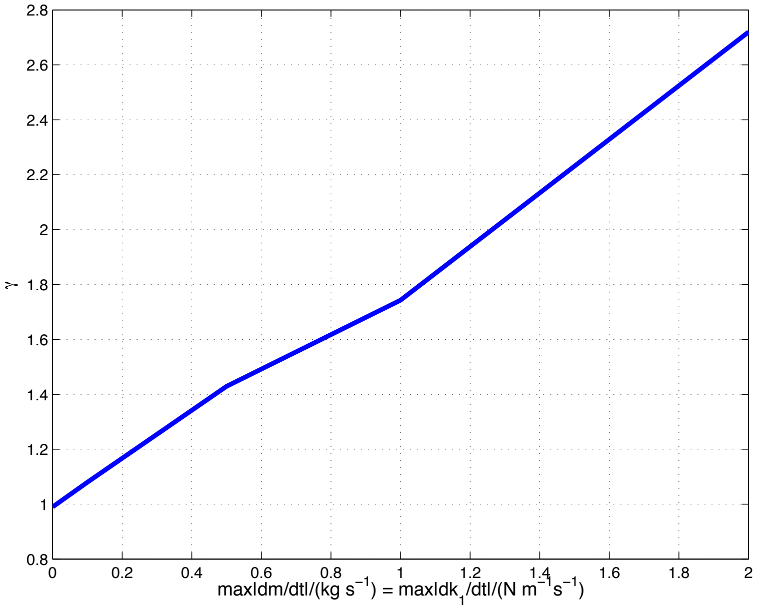


Figure 2.15: LFB Design: $\mathcal{H}_\infty$-norm of the closed-loop, γ, versus the maximum rate of variation of the mass, m, and of the spring stiffness, k_1.

depicted in Fig. 2.5. For values of $\dot{\rho} > 0$, one can start with the mixed-μ controller and try to find a valid initialization for the algorithm by reducing the performance gain, A_p. $\diamond$

The results presented next were obtained assuming the bounds on the rate of variation of $|\dot{k}_1| \leq 0.1$ (N/m)/s and $|\dot{m}| \leq 0.1$ kg/s. For this configuration, we get a value of γ of approximately 0.99 using $A_p = 950$. We recall that, for the D,G-K iterations, the performance gain was $A_p^{\text{mixed-}\mu} = 750$. Therefore, *for this particular example*, the LPV/BMI controller provides better performance guarantees than the mixed-μ controller, for a worst-case situation, even when assuming time-variations of the uncertain parameters.

Figure 2.16 depicts the Bode plots of the three controllers considered so far, *i.e.*, the mixed-μ controller, the LPV/BMI controller for the LTI plant, and the LPV/BMI controller for the LPV plant with $|\dot{k}_1| \leq 0.1$ (N/m)/s and $|\dot{m}| \leq 0.1$ kg/s. The LPV/BMI controllers have larger low-frequency gains, which is intuitively in accordance to the fact that they lead to higher levels of performance than the ones obtained with the μ-controller, for the present example. By comparing the two LPV/BMI controllers, it can also be observed that increasing the maximum allowable rate of variation of the parameters results in a decrease in terms of the controller gain, as expected.

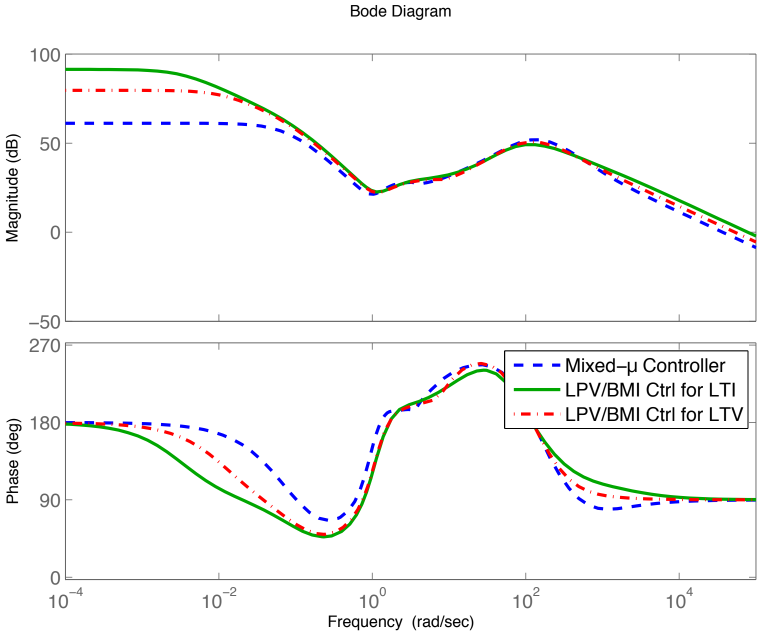


Figure 2.16: LFB Design: Bode plots for the mixed-μ and LPV/BMI controllers for the LTI plant and for the LPV plant with $|\dot{k}_1| \leq 0.1$ Nm^{-1}s^{-1} and $|\dot{m}| \leq 0.1$ kg/s.

A time-simulation of the plant with both controllers was performed, for the time-varying spring stiffness, k_1, and mass, m, shown in Fig. 2.17. We used the low-frequency exogenous disturbances depicted in Fig. 2.13. Figure 2.18 shows the results obtained. The LPV/BMI controller provides a significant increase in terms of disturbance-rejection, when compared to the μ-controller. In particular, regarding the RMS performance, the results obtained are as follows:

$$\begin{aligned}\text{RMS for LPV/BMI Controller} &= 0.0275 \text{ m},\\ \text{RMS for Mixed-}\mu\text{ Controller} &= 0.0509 \text{ m}.\end{aligned}$$

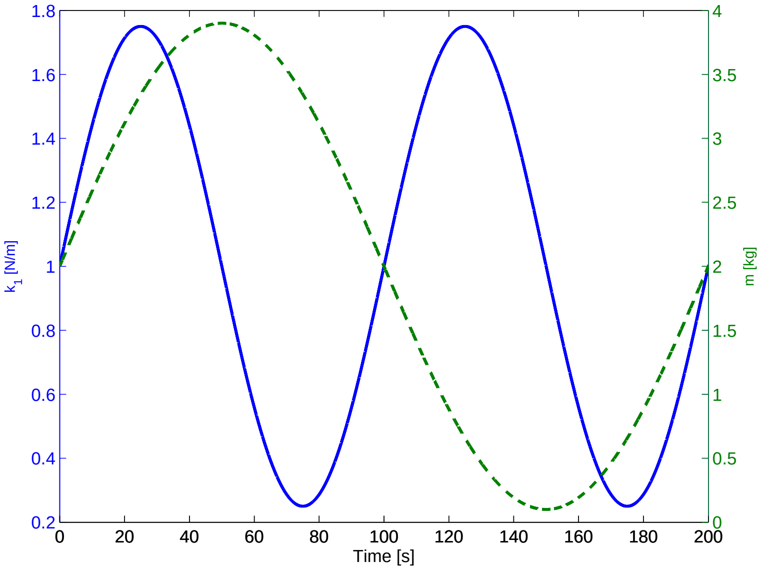


Figure 2.17: Time-varying spring stiffness, k_1, and mass, m, with $|\dot{k}_1| \leq 0.1$ Nm^{-1}s^{-1} and $|\dot{m}| \leq 0.1$ kg/s.

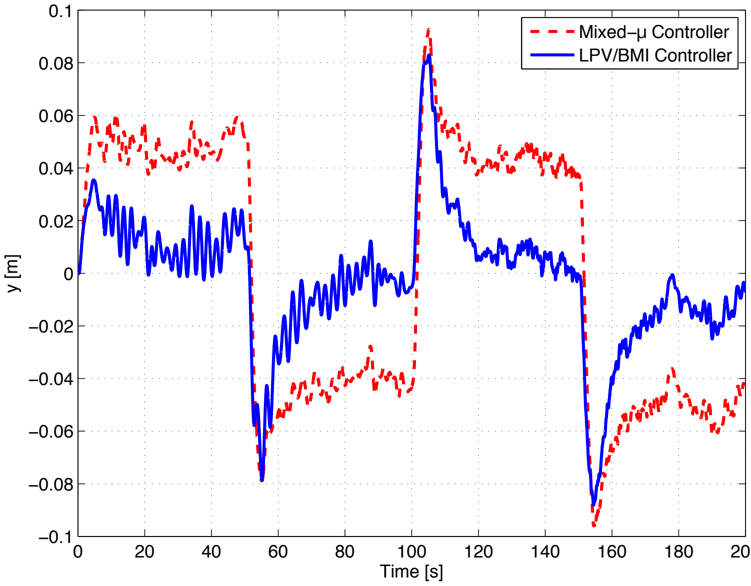


Figure 2.18: LFB Design: Output of the closed-loop using the LPV/BMI and mixed-μ controllers, for the time-varying spring stiffness, k_1, and mass, m, with $|\dot{k}_1| \leq 0.1$ Nm^{-1}s^{-1} and $|\dot{m}| \leq 0.1$ kg/s, depicted in Fig. 2.17.

2.4.4 HFB Design: Analysis for LTI Plants

In the following simulations, we use the MSD model in (2.24), but with a smaller value of b, namely

$$b = 0.05 \text{ N/(m/s)}.$$

The poles of the open-loop system for different values of k_1 and m are illustrated in Fig. 2.19. In comparison with Fig. 2.10, the poles can now take place closer to the imaginary axis. This immediately leads to the following conclusions:

- the achievable performance of the closed-loop system is decreased;
- the D,G-K iterations, due to the modeling of real-valued uncertainties by complex-valued uncertainties, can become even more conservative, since it will guarantee stability and performance for realizations of the plant that potentially include unstable poles.

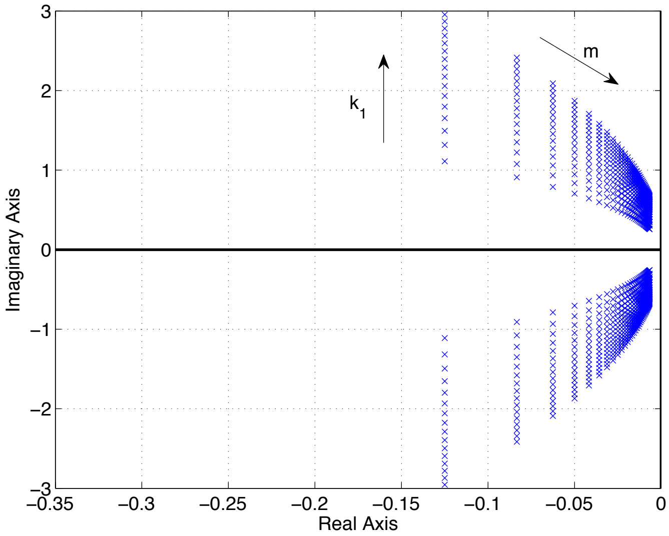


Figure 2.19: Poles of the open-loop plant with $b = 0.05$ N/(m/s).

In this subsection, we suppose that $\dot{k}_1 = 0$ and $\dot{m} = 0$. Similarly to the LFB design, we synthesize an LPV/BMI controller using the method described in Section 2.3.

The Bode plots of the controllers transfer functions for the LTI case are depicted in Fig. 2.20. The conclusions are analogous to those of the LFB design, *i.e.*, although the controllers are similar, the μ-controller has smaller low-frequency gain, which explains the benefits in terms of performance of the LPV/BMI controller.

In terms of guaranteed $\mathcal{H}_\infty$ performance, we obtained

$$A_p^{\text{mixed}-\mu} = 19,$$
$$A_p^{\text{LPV/BMI}} = 81.$$

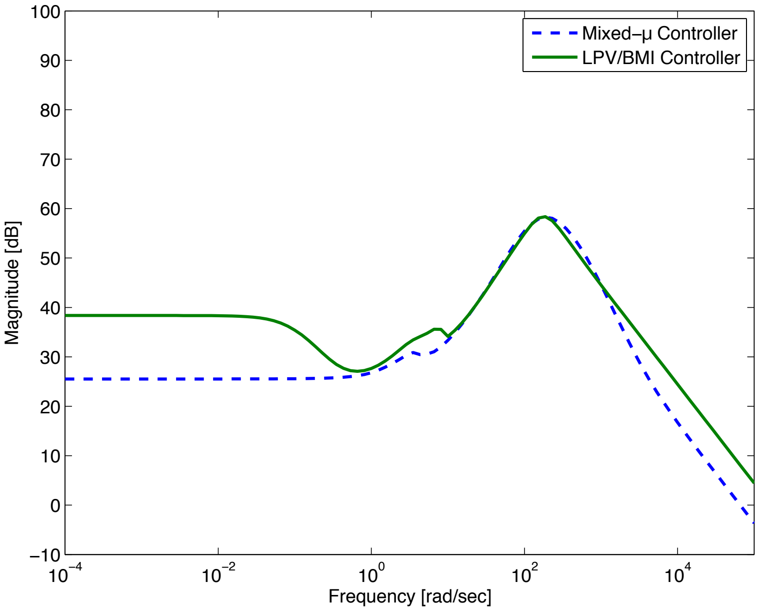


Figure 2.20: HFB Design: Bode plots for the mixed-μ and LPV/BMI controllers, for $\dot{k}_1 = 0$ and $\dot{m} = 0$.

The behaviors of both controllers are assessed in simulation, using $k_1 = 1.75$ N/m and $m = 0.2$ kg. The disturbances, $d(t)$, were generated by driving a low-pass filter with transfer function (2.22) with continuous-time white noise with zero mean and intensity 10. The output of the closed-loop system using each controller and averaging 10 Monte-Carlo runs is depicted in Fig. 2.21. Consider the RMS of the performance output, $z_1(\cdot)$, of the closed-loop, using each controller. For this case, we get

$$\begin{aligned} \text{RMS for LPV/BMI Controller} &= 0.0325 \text{ m}, \\ \text{RMS for Mixed-}\mu\text{ Controller} &= 0.0418 \text{ m}. \end{aligned}$$

Therefore, the RMS of the performance output, $z_1(\cdot)$, of the closed-loop with the LPV/BMI controller is nearly 80% of that obtained with the mixed-μ controller.

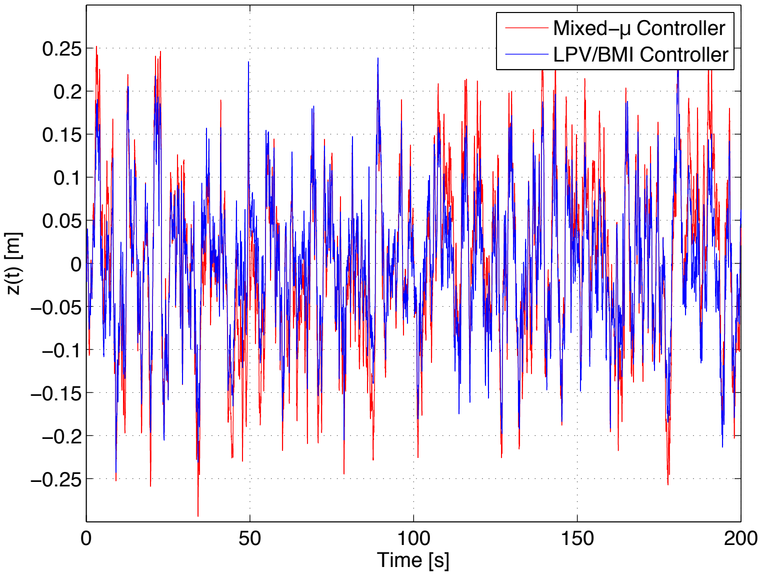


Figure 2.21: HFB Design: Output of the closed-loop for the LPV/BMI and mixed-μ controllers, for $\dot{k}_1 = 0$, $\dot{m} = 0$, $k_1 = 1.75$ N/m and $m = 0.2$ kg.

Therefore, for time-invariant plants, the level of $\mathcal{H}_\infty$ closed-loop performance guaranteed by the mixed-μ controller is only about 23% of that guaranteed by the LPV/BMI controller.

2.4.5 HFB Design: Analysis for Time-Varying Plants

The methodology used to synthesize the LPV/BMI controller for the LFB design with time-varying parameters was also adopted in this subsection. Figure 2.22 depicts the value of the $\mathcal{H}_\infty$-norm of the closed-loop from the exogenous inputs to the performance output, $z_1(\cdot)$, using LPV/BMI controllers, γ, versus the maximum rate of variation of the mass, m, and of the spring stiffness, k_1. As in the LFB case, the value of γ increases when we allow m and k_1 to change faster with time.

The results presented next were obtained assuming that the bounds on the rate of variation of $|\dot{k}_1| \leq 0.1$ (N/m)/s and $|\dot{m}| \leq 0.1$ kg/s. For this configuration, we get a value of γ of approximately 0.99 using $A_p = 75$. We recall that, for the D,G-K iterations, the performance gain was $A_p^{\text{mixed-}\mu} = 19$. Therefore, *for this particular example*, the LPV/BMI controller provides better performance guarantees than the

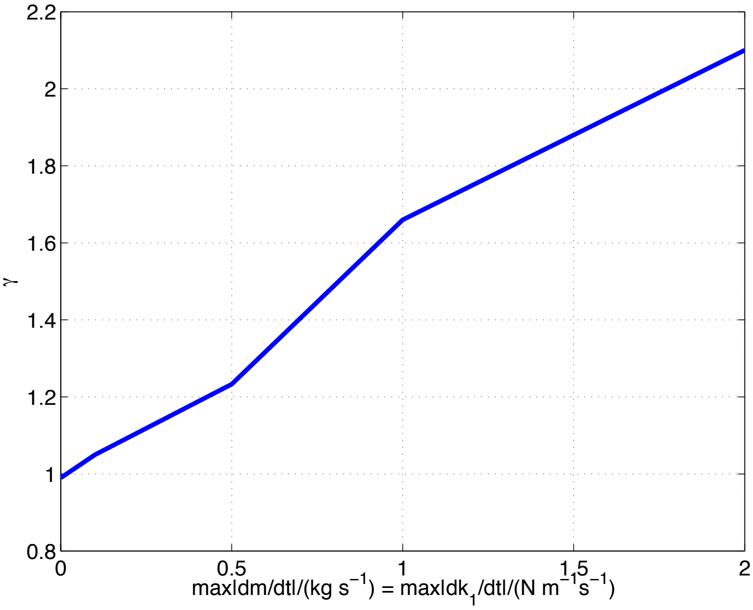


Figure 2.22: HFB Design: $\mathcal{H}_\infty$-norm of the closed-loop, γ, versus the maximum rate of variation of the mass, m, and of the spring stiffness, k_1.

mixed-μ controller, for a worst-case situation, even when assuming time-variations of the uncertain parameters.

Figure 2.23 depicts the Bode plots of the three controllers. As in the LFB design, by comparing the two LPV/BMI controllers, it can also be observed that increasing the maximum allowable rate of variation of the parameters results in a decrease in terms of the controller gain, as expected.

Two time-simulations of the plant with both controllers were performed:

1. for the first simulation, the disturbances were generated by driving a low-pass filter with transfer function (2.22) with continuous-time white noise with zero mean and intensity 10;

2. for the second simulation, we considered the low-frequency disturbances depicted in Fig. 2.24.

In both cases, we used the time-varying spring stiffness, k_1, and mass, m, shown in Fig. 2.17. Figure 2.25 shows the results obtained by averaging 10 Monte-Carlo runs. Once again, the LPV/BMI controller provides a significant increase in terms of

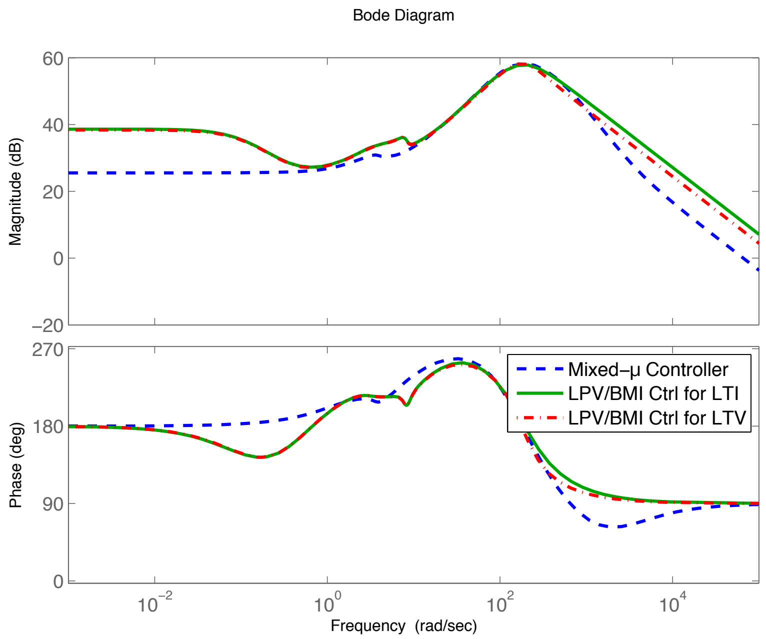


Figure 2.23: HFB Design: Bode plots for the mixed-μ and LPV/BMI controllers for the LTI plant and for the LPV plant with $|\dot{k}_1| \leq 0.1$ Nm^{-1}s^{-1} and $|\dot{m}| \leq 0.1$ kg/s.

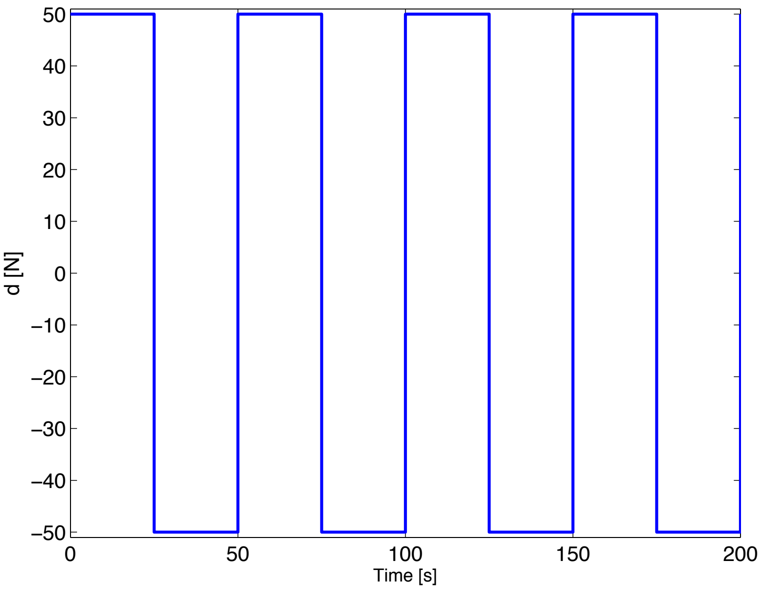


Figure 2.24: HFB Design: Low-frequency exogenous disturbances, $d(t)$.

disturbance-rejection, when compared to the μ-controller. In particular, regarding the RMS performance, the results obtained are as follows:

$$\begin{aligned}\text{RMS for LPV/BMI Controller} &= 0.0381 \text{ m},\\ \text{RMS for Mixed-}\mu\text{ Controller} &= 0.0480 \text{ m}.\end{aligned}$$

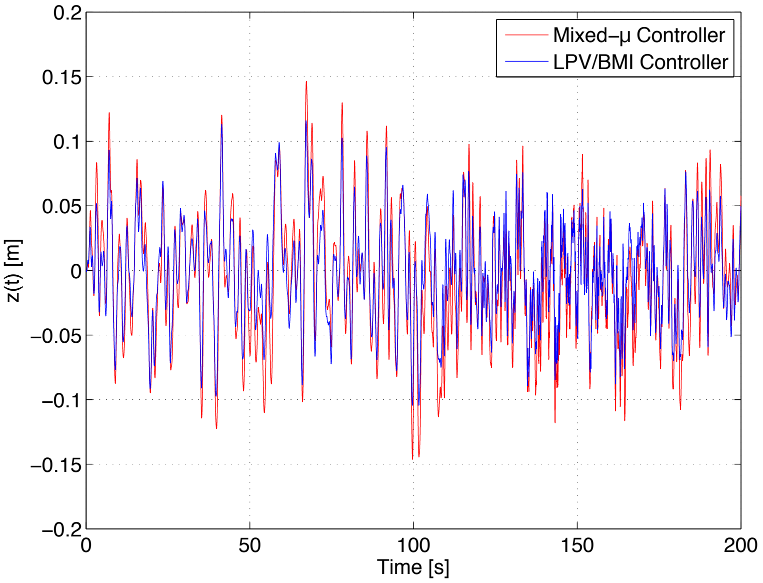


Figure 2.25: HFB Design: Output of the closed-loop using the LPV/BMI and mixed-μ controllers, for the time-varying spring stiffness, k_1, and mass, m, with $|\dot{k}_1| \leq 0.1$ Nm^{-1}s^{-1} and $|\dot{m}| \leq 0.1$ kg/s, depicted in Fig. 2.17, and with disturbances generated by driving a low-pass filter with transfer function (2.22) with continuous-time white noise with zero mean and intensity 10.

Remark 2.16: For the LTI case, the RMS of the output of the closed-loop system with the mixed-μ controller was approximately 0.0418 m. For the time-varying case, and also using the mixed-μ controller, the RMS of the output is increased up to 0.0480 m, which is equivalent to nearly 15%. Thus, this example illustrates the shortcomings of the mixed-μ controllers when the plant to be controlled is time-varying. $\diamond$

To highlight the differences in terms of performance provided by the two types of controllers, we now consider a simulation obtained by using the low-frequency exogenous disturbances depicted in Fig. 2.24. Figure 2.26 shows the results obtained. The benefits in terms of disturbance-rejection of the LPV/BMI controller, when compared to the μ-controller, are even clearer in this case. The RMS of the output of the closed-loop system, using both types of controllers are as follows:

$$\begin{aligned} \text{RMS for LPV/BMI Controller} &= 1.162 \text{ m}, \\ \text{RMS for Mixed-}\mu \text{ Controller} &= 2.441 \text{ m}. \end{aligned}$$

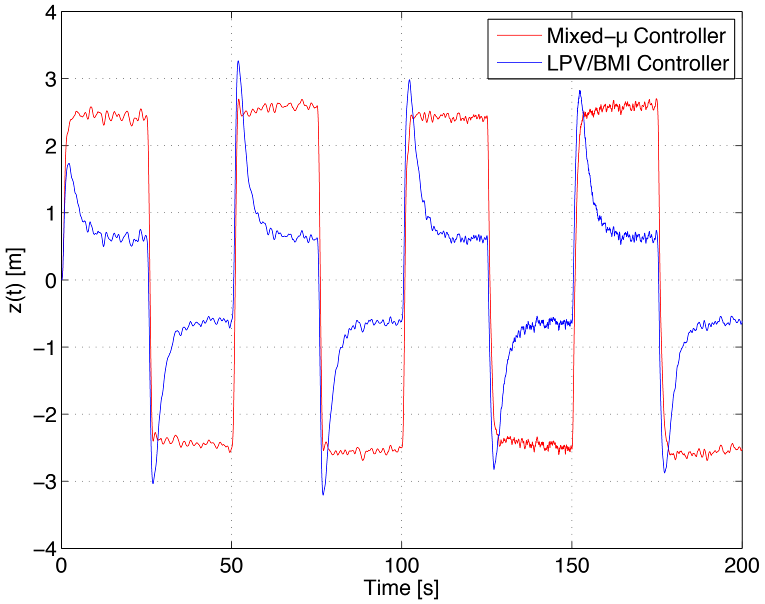


Figure 2.26: HFB Design: Output of the closed-loop using the LPV/BMI and mixed-μ controllers, for the time-varying spring stiffness, k_1, and mass, m, with $|\dot{k}_1| \leq 0.1$ Nm^{-1}s^{-1} and $|\dot{m}| \leq 0.1$ kg/s, depicted in Fig. 2.17, and using the exogenous disturbances illustrated in Fig. 2.24.

The RMS of the output of the closed-loop system, using the LPV/BMI controller is less than half of the value obtained with the mixed-μ controller, for this type of disturbances.

We stress that the rate of variation of the time-varying parameters is within the bounds used to synthesize the LPV/BMI controllers. Therefore, we can guarantee robust-stability and -performance of the closed-loop if the LPV/BMI controller is used. Moreover, we also have a smaller upper bound on the $\mathcal{H}_\infty$-norm of the closed-loop system, when compared to the mixed-μ controller. Finally, we also get significant improvements in terms of RMS performance for this example, as illustrated in the simulations.

2.5 Conclusions

It was shown that LPV/BMI controllers can become significantly less conservative than μ-controllers when there is more than one uncertain parameter affecting the plant dynamics. We also demonstrated that LPV/BMI controllers not only guarantee stability and performance for time-varying plants, but also provide higher performance guarantees than (or at least as good as) mixed-μ controllers.

We also developed a new computational algorithm, called the D-BMI algorithm, which incorporates the use of D-scales in the design of LPV/BMI controllers, by alternating between the D-scales optimization, and the BMI optimization. The optimization problem subject to BMI constraints can be solved either by general BMI optimization procedures, or by alternating the optimization process between the two sets of variables that generate the bilinearity. In the latter algorithm, which was developed in this thesis, the BMIs are locally solved using LMIs, which corresponds to much faster optimization problems. The shortcoming with such an approach is that we may not converge to the optimal solution. The technique introduced in [58, 59] to derive lower bounds in terms of performance for the closed-loop plant, while guaranteeing stability against model uncertainty, was also used in the design of the LPV/BMI controllers.

Therefore, LPV/BMI controllers may be used in a wide spectrum of applications, providing

a) potentially higher performance guarantees than (or at least as good performance guarantees as those provided by) mixed-μ controllers for LTI plants;

b) stability and performance guarantees for closed-loop time-varying plants that depend upon time-varying (real-valued) parameters;

c) potentially less conservative results than those obtained with the D,G-K iterations, when there are parametric uncertainties in the plant model.

The key differences between LPV/BMI and mixed-μ controllers were illustrated in a simulation, using LTI and LTV plants. We were able to show that, at least for the present examples, a significant increase in terms of performance can be achieved due to the use of the LPV/BMI controllers.

The proposed method to synthesize these controllers, however, does not void the use of the D,G-K iterations[5], since they are still required to compute an initial guess for the BMI optimization algorithm and to obtain the D-scales required to handle the unmodeled dynamics. Moreover, contrary to μ-compensators, the synthesis of LPV/BMI controllers usually require a heavier, off-line, computational burden. Thus, further research on solving optimization problems subject to BMI constraints is needed in order to apply this methodology to plants with a larger numbers of states.

In closing, we remark that the design of LPV/BMI controllers can be used in Robust Multiple-Model Adaptive Control (RMMAC) architectures – see [58, 59] –, as shown in the following chapter. The overwhelming reason for using adaptive control is the possible time-variations of some real-valued plant parameters. As shown in [49] and in the following chapter, the bank of controllers in the RMMAC architecture can be the LPV/BMI controllers designed according to the principles and algorithms described in this chapter. However, the guarantees in terms of performance and stability are jeopardized if the decision subsystem of the RMMAC does not guarantee the convergence for the *correct* model, as explained in the Chapter 4.

[5] Any improvement in the D,G-K-type of iterations will also help the BMI optimization.

Chapter 3

Standard RMMAC vs RMMAC/BMI

3.1 Introduction

In the previous chapter, a new methodology to design (non-adaptive) controllers that are robust against time-varying uncertainties in the model of the plant was proposed. As previously stressed, LPV/BMI controllers can become significantly less conservative than μ-controllers when there are more than one uncertain parameter affecting the plant dynamics. Unlike mixed-μ controllers, LPV/BMI controllers not only guarantee stability and performance for Linear Time-Invariant (LTI) plants, but also for Linear Parameter-Varying (LPV) plants. Moreover, the performance guarantees provided by an LPV/BMI controller are at least as good as those provided by the corresponding mixed-μ controller. This motivated the integration of the LPV/BMI controllers with the Robust Multiple-Model Adaptive Control (RMMAC) architecture – see Fig. 3.1.

As stressed in Chapter 1, the RMMAC is a multiple model approach that computes and uses the posterior probabilities of the uncertain parameters of the process model being in a specific region to switch or blend the outputs of a set of controllers, each of which designed for a given uncertainty region. The estimation part is done by a bank of Kalman filters (KFs), while for the control part a set of mixed-μ

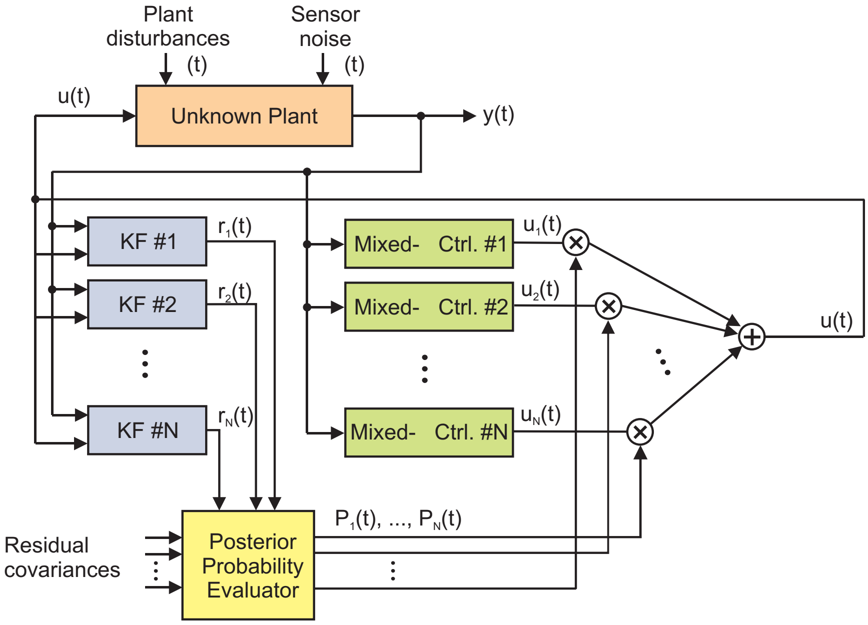


Figure 3.1: RMMAC architecture.

controllers is used. For further details, the interested reader is referred to [57–59].

To obtain what we are going to refer to as an *RMMAC/BMI control architecture*, from a standard RMMAC design, we simply have to replace the mixed-μ controllers by the corresponding LPV/BMI controllers. The design specifications of the LPV/BMI controllers are similar to those of the mixed-μ controllers, but assuming nonzero bounds on the rate of variation of the uncertain parameters of the plant, as detailed in the previous chapter. Since the design assumptions are very similar in both cases, except for the time-variation of the parameter, little effort is needed in order to "upgrade" the standard RMMAC method to this novel architecture. Figure 3.2 depicts the RMMAC/BMI architecture. This chapter is, therefore, devoted to the comparison between the standard RMMAC and the RMMAC/BMI, through an example.

3.1.1 Main Contributions and Organization

The key contributions of this chapter are as follows:

1. The integration of LPV/BMI controllers in the RMMAC architecture;

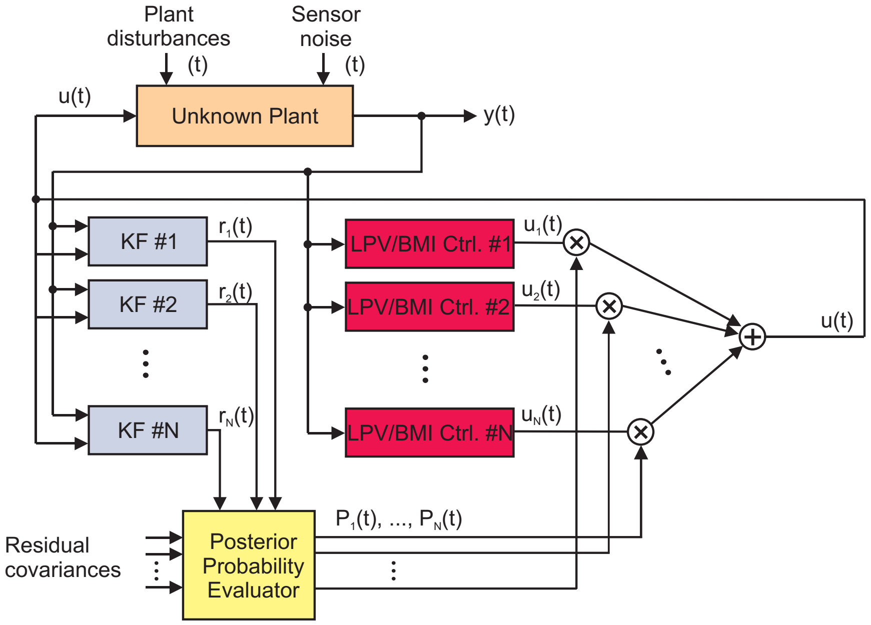


Figure 3.2: RMMAC/BMI architecture.

2. An insightful practical comparison between the standard RMMAC and the RMMAC/BMI architectures.

The remainder of this chapter is organized as follows: in Section 3.2, the double Mass-Spring-Dashpot (MSD) plant is introduced; Section 3.3 presents several Monte-Carlo simulation results illustrating the advantages of the use of the RMMAC/BMI architecture over the *classical one*; finally, in Section 3.4, the advantages and shortcomings of each of the methodologies are discussed.

3.2 The Double Mass-Spring-Dashpot Plant

We are going to resort to the (double) Mass-Spring-Dashpot (MSD) testbed example of Fig. 3.3 to illustrate the benefits of using LPV/BMI controllers within the RMMAC architecture. We stress that this testbed has recently been used with the so-called multiple model adaptive control with mixing (MMACwM) architecture – see [41, 100] – with uncertain time-invariant parameters. Notice that the MSD testbed represents a challenging control problem, because the control is non-colocated with the performance variable, the position of mass m_2 – see [101] and references therein for a list of robust control studies using the MSD framework – and that the use of a non-adaptive controller deteriorates significantly the performance of the overall system. Further discussion on this subject is presented in the next chapters. Moreover, there are several real life applications that share the dynamics of the MSD plant, for instance in seismic and vibration models [1–5], automotive suspension systems [6–9], flexible space structures [10–15], among others, as stressed in [100, 102].

The state-space description of the dynamics of the MSD system, excluding the dynamics of the disturbances, is given by

$$\begin{cases} \dot{x}(t) &= A(k_1)x(t) + Bu(t) + L\xi(t), \\ y(t) &= Cx(t) + \theta(t), \end{cases}$$

where

$$x^{\mathrm{T}}(t) = \begin{bmatrix} x_1(t) & x_2(t) & x_3(t) & x_4(t) \end{bmatrix}$$

is the state of the plant, and

$$A(k_1) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{k_1}{m_1} & \frac{k_1}{m_1} & -\frac{b_1}{m_1} & \frac{b_1}{m_1} \\ \frac{k_1}{m_2} & -\frac{k_1+k_2}{m_2} & \frac{b_1}{m_2} & -\frac{b_1+b_2}{m_2} \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{m_1} \\ 0 \end{bmatrix}, \quad L = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix},$$

$$C = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}.$$

where

$$m_1 = m_2 = 1 \text{ kg}, \quad k_2 = 0.15 \text{ N/m}, \quad b_1 = b_2 = 0.1 \text{ N/(m/s)}.$$

In what follows, the unknown parameters are the time-varying spring stiffness

$$k_1 \in K := [0.25 \quad 1.75] \text{ N/m}$$

and the constant but unknown input time-delay, bounded by

$$0 < \tau < 0.05 \text{ s}.$$

The disturbance force, $d(\cdot)$, shown in Fig. 3.3 is a stationary first-order (colored) stochastic process generated by driving a low-pass filter, with transfer function

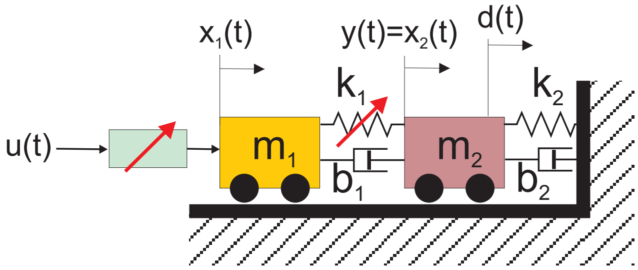


Figure 3.3: MSD system with uncertain spring constant, k_1, and disturbances denoted by $d(t)$. $u(t)$ is the control input and $z(t)$ is the system output.

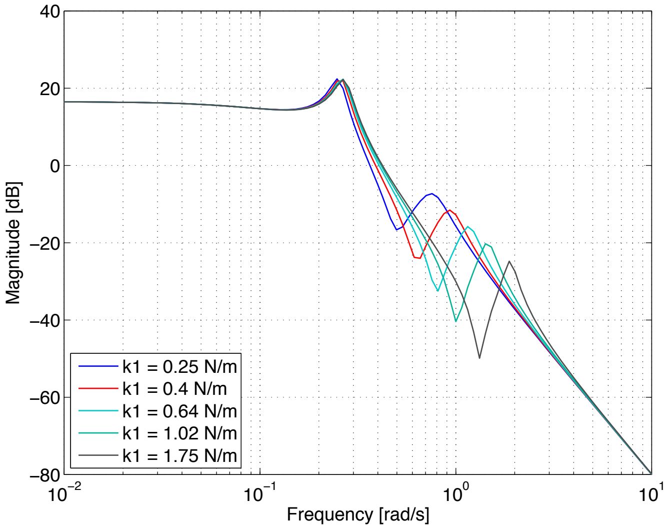


Figure 3.4: Magnitude of the transfer function from $\xi(\cdot)$ to the output $y(\cdot)$, for several values of the spring stiffness, k_1.

$W_d(s)$, with continuous-time white noise $\xi(\cdot)$, with zero mean and intensity $\Xi = 1$, according to

$$d(s) = \frac{\alpha}{s+\alpha}\xi(s) = W_d(s)\xi(s).$$

The sensor noise considered is continuous-time white noise $\theta(\cdot)$, with zero mean and intensity $\Theta = 0.001$.

The magnitude of the transfer function from $\xi(\cdot)$ to $y(\cdot)$, for different values of the spring stiffness, k_1. Notice that the spring stiffness impacts significantly more the second natural frequency of the MSD system than the first one.

Following the RMMAC synthesis methodology and using the same design choices as the ones described in [58, 59], we obtain $N = 4$ Local Non-Adaptive Robust Controllers (LNARCs) – which are mixed-μ controllers in the original RMMAC design – in order to achieve at least 70% of the performance we would have obtained had we known the value of the uncertain parameter, k_1. The boundaries for the spring constant for each model are summed up in Table 3.1.

Remark 3.1: The so-called problem of *model coverage* is tackled, in this thesis, mainly by resorting to the results in [58, 59]. However, different approaches are available in the literature to address this topic, in particular by developing (stability-based) proximity measures between a given dynamic system and a set of models generated by considering bounded parametric uncertainty – see [103, 104] and references therein.

Moreover, in Chapter 5, we introduce the concept of *absolute input distinguishability*, which can be used to discard models (and corresponding controllers) that cannot be distinguished from each other, based upon a given realization of the plant. ◇

Model	Spring Stiffness Uncertainty Interval
#1	$\Omega_1 = [1.02\ \ 1.75]$
#2	$\Omega_2 = [0.64\ \ 1.02]$
#3	$\Omega_3 = [0.4\ \ 0.64]$
#4	$\Omega_4 = [0.25\ \ 0.4]$

Table 3.1: RMMAC Model Definitions

The mixed-μ controllers are thereafter substituted by LPV/BMI controllers with similar specifications, but assuming nonzero bounds on the time-rate of variation of parameter k_1. In this design, we assume a bound of 0.001 (N/m)/s for the slope (time-variation) of parameter k_1. The state of the plant is augmented by the D-scales obtained from the μ-analysis, increasing the state up to 16 variables. For the LPV realization, we have adopted a polytopic description for the system, since, for each region, the plant dynamics can be naturally described as the convex combination of the dynamics on the boundaries of that region. We used the mixed-μ controller designed for the whole region ($k_1 \in [0.25, 1.75]\, N/m$) to initialize the algorithm. Each D-BMI iteration takes about 6 hours running in a Pentium IV at 2.6 GHz and each controller takes up to 10 iterations to achieve an $\mathcal{L}_2$-induced norm from the inputs to the outputs of the closed-loop smaller than 1. The μ-synthesis only requires a few minutes to obtain a solution which is, however, in general worse than that of the D-BMI iterations.

A *Global Non-Adaptive Robust Controller*, denoted by GNARC, was also designed, for comparison purposes. The GNARC determines the lower bound in terms of robust performance in the absence of adaptive control, and consists of a mixed-μ controller designed for the whole uncertainty region, K.

3.3 Simulation Results

We considered three different cases for the comparison between the standard RMMAC and the RMMAC/BMI. For each case, 10 Monte-Carlo runs were performed. In the first scenario, we consider k_1 constant, while in the second and third ones, we assume that k_1 is time-varying.

Each of the aforementioned situations is going to be analyzed in the following subsections. The RMS results presented are obtained by averaging the 10 Monte-Carlo runs of each case. The time-domain results illustrated in the figures are representative simulations.

3.3.1 Case #1

In the first case considered, we assumed that the spring stiffness is constant. In particular, let

$$k_1 = 0.9 \text{ N/m}.$$

The output of the plant, $x_2(t)$, is depicted in Fig. 3.5, for the different controllers used. The improvements in terms of performance due to the use of adaptive control are clear from the figure. The outputs of the closed-loop systems using the classical RMMAC and the RMMAC/BMI are similar for this case. The RMS value of x_2 is

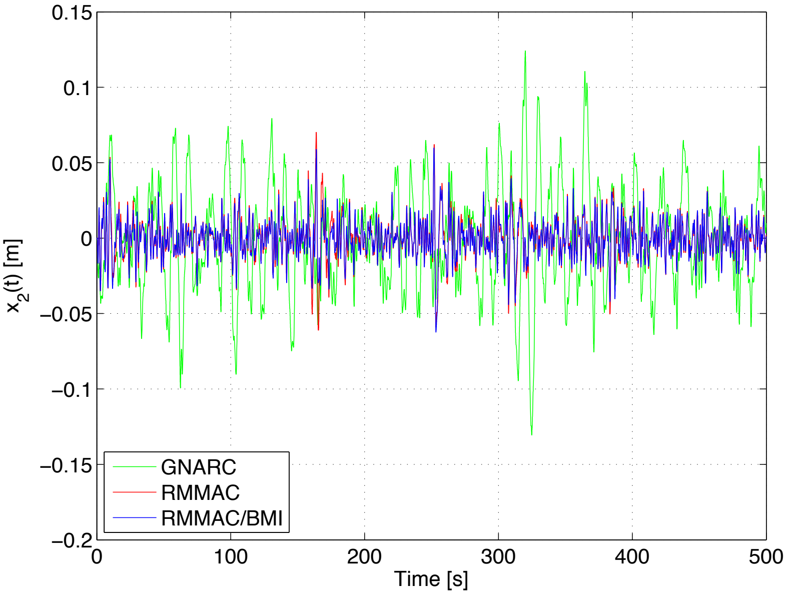


Figure 3.5: Case #1: output of the double MSD plant.

0.0123 m for the RMMAC/BMI controller, and 0.0128 m for the standard RMMAC. For the sake of completeness, the RMS value of x_2 using the GNARC is 0.0327 m.

Let $\%E$ denote the ratio between the RMS value of the output of the closed-loop using the RMMAC/BMI and the RMS value of the output obtained using the classical RMMAC, *i.e.*,

$$\%E = \frac{\text{RMS RMMAC/BMI}}{\text{RMS RMMAC}} \times 100\%. \tag{3.1}$$

Analogously, let $\%F$ relate the RMS of the output obtained by using the RMMAC/BMI with that obtained with the GNARC, *i.e.*,

$$\%F = \frac{\text{RMS RMMAC/BMI}}{\text{RMS GNARC}} \times 100\%. \tag{3.2}$$

Remark 3.2: The quantities $\%E$ and $\%F$ can be used to assess the improvements due to the use of the RMMAC/BMI architecture, when compared to the classical RMMAC and to the GNARC, respectively. ◇

For the current scenario, we have that $\%E = 96.9\%$ and $\%F = 37.6\%$. Therefore, we are clearly obtaining a significant improvement in terms of performance by using adaptive control. The improvements due to the use of LPV/BMI controllers, however, are not significant in this example.

The posterior probabilities of each model, computed using the residuals of the KFs, are depicted in Fig. 3.6. Since the outputs of the closed-loop systems, using both adaptive control strategies, are similar, the differences between the corresponding posterior probabilities are also not significant. Therefore, the decision subsystem of the RMMAC does not impact severely on the performance of the controllers, in the present case.

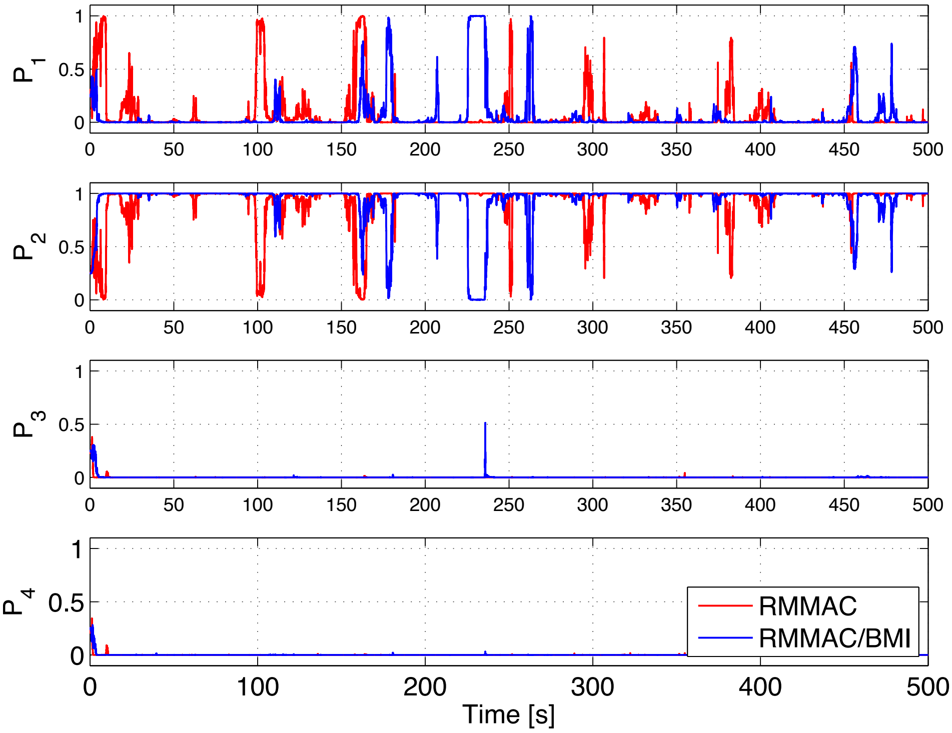


Figure 3.6: Case #1: posterior probabilities of each model.

3.3.2 Case #2

In the following simulation, we consider that the spring stiffness, $k_1(\cdot)$, is time-varying and with the time-evolution depicted in Fig. 3.7. The time-variations of the uncertain parameter, k_1, impact not only the control part, but also the decision subsystem.

Figure 3.8 depicts the results obtained. Once again, the performance of the RMMAC/BMI controller is slightly higher than that of the classical RMMAC. Moreover, it is apparent from Fig. 3.8 that both adaptive schemes performed much better than

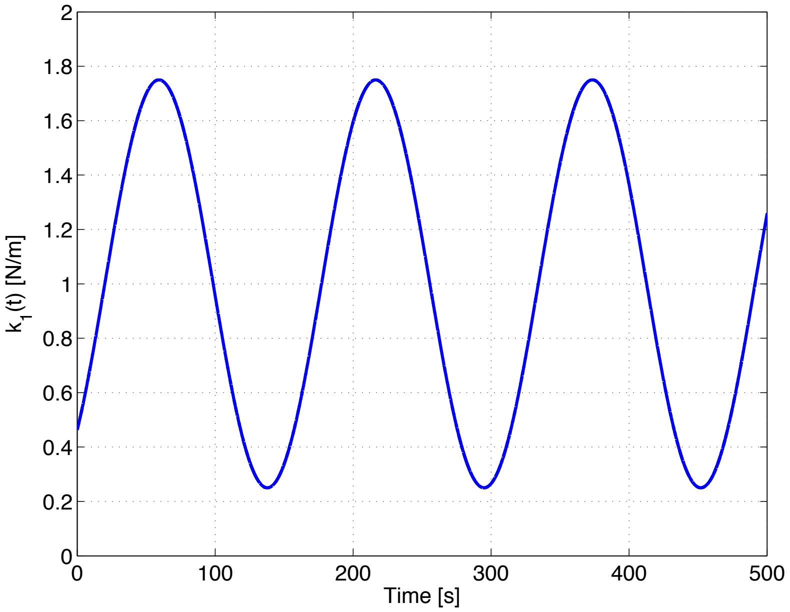


Figure 3.7: Case #2: time-varying spring stiffness, $k_1(\cdot)$.

the GNARC. Indeed, the RMS of the output is 0.0159 m for the RMMAC/BMI, 0.0165 m for the standard RMMAC, and 0.0432 m for the non-adaptive controller.

Once again, the improvements in terms of RMS performance can be assessed by evaluating $\%E$ and $\%F$ in (3.1) and (3.2), respectively. In the present case, we have $\%E = 96.3\%$ and $\%F = 36.8\%$.

For the sake of completeness, the posterior probabilities of each model are illustrated in Fig. 3.9. The behavior of the aforementioned posterior probabilities are similar in both cases, as the results obtained with the RMMAC/BMI and with the classical RMMAC do not differ significantly. However, the transients observed in the identification subsystem slightly deteriorate the performance of the overall system.

3.3.3 Case #3

In the following example, we also consider that the spring stiffness is time-varying, but now with the time-evolution depicted in Fig. 3.10. Notice that, in comparison with the previous example, the variations of the spring stiffness are much faster and, therefore, the decision problem is expected to be harder. Moreover, the design

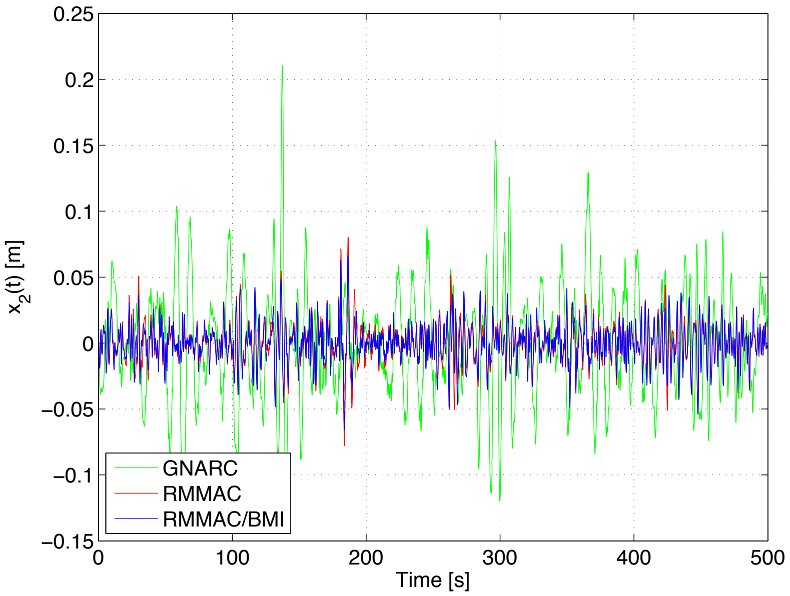


Figure 3.8: Case #2: output of the double MSD plant.

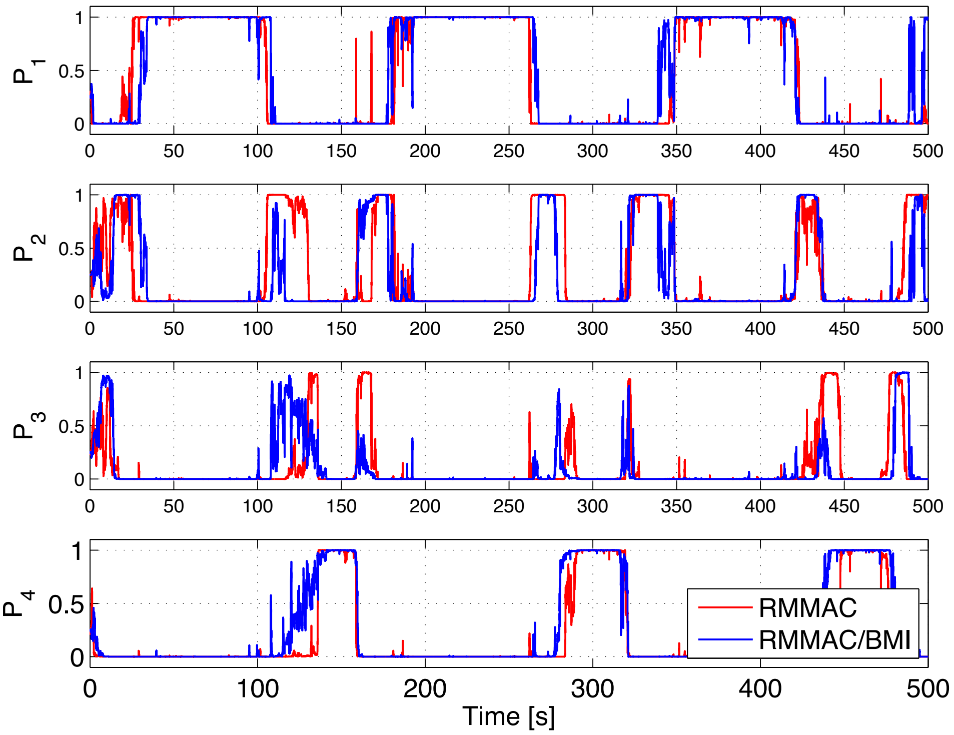


Figure 3.9: Case #2: posterior probabilities of each model.

assumptions of the mixed-μ are clearly violated, which may lead to the deterioration of the performance of the closed-loop system.

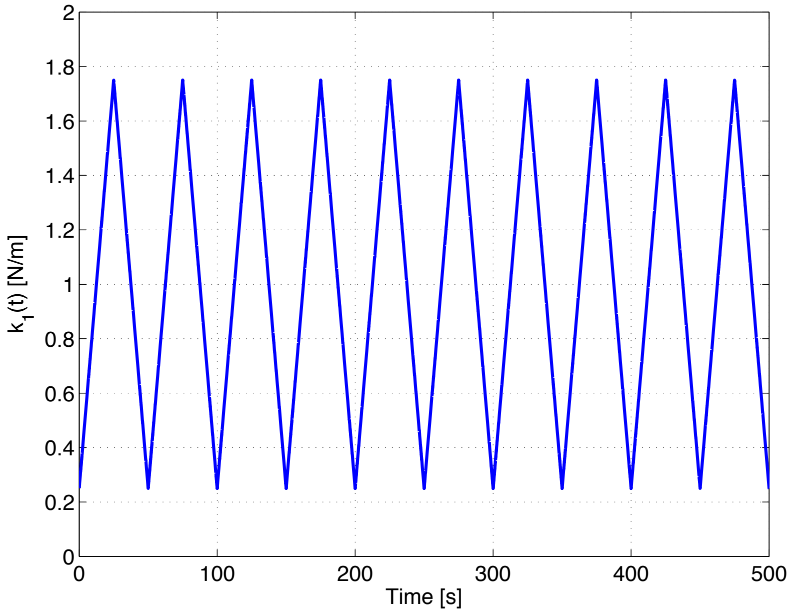


Figure 3.10: Case #3: time-varying spring stiffness, $k_1(\cdot)$.

The outputs of the closed-loop for each controller are depicted in Fig. 3.11. In this case, the RMS of the output using the RMMAC/BMI is 0.0277 m, while that of the classical RMMAC is 0.0330 m. Therefore, we have that $\%E = 83.9\%$, indicating that the value of the RMS of the output diminishes significantly due to the use of LPV/BMI controllers. This is in accordance with what is stated in the previous chapter, since, unlike mixed-μ compensators, the LPV/BMI controllers have performance guarantees against time-variations of the plant. The RMS of the output using the GNARC is 0.0400 m, and thus $\%F = 69.2\%$.

Hence, in the present case, the improvements in terms of RMS performance due to the use of adaptive control are not as significant as in the previous examples. This suggests that the decision subsystem may not be weighting properly the control signals. Indeed, Fig. 3.12 shows that the identification procedure is not sufficiently fast to adapt itself to the changes in the dynamics of the plant.

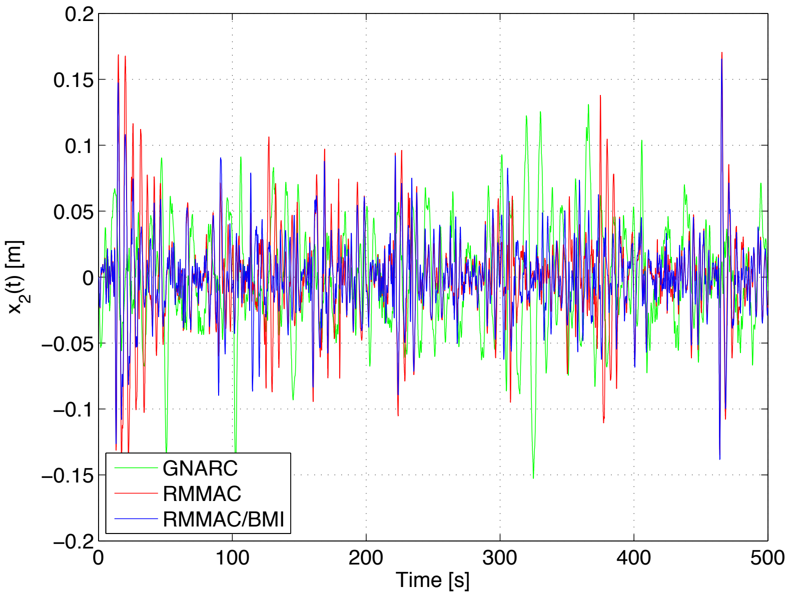


Figure 3.11: Case #3: output of the double MSD plant.

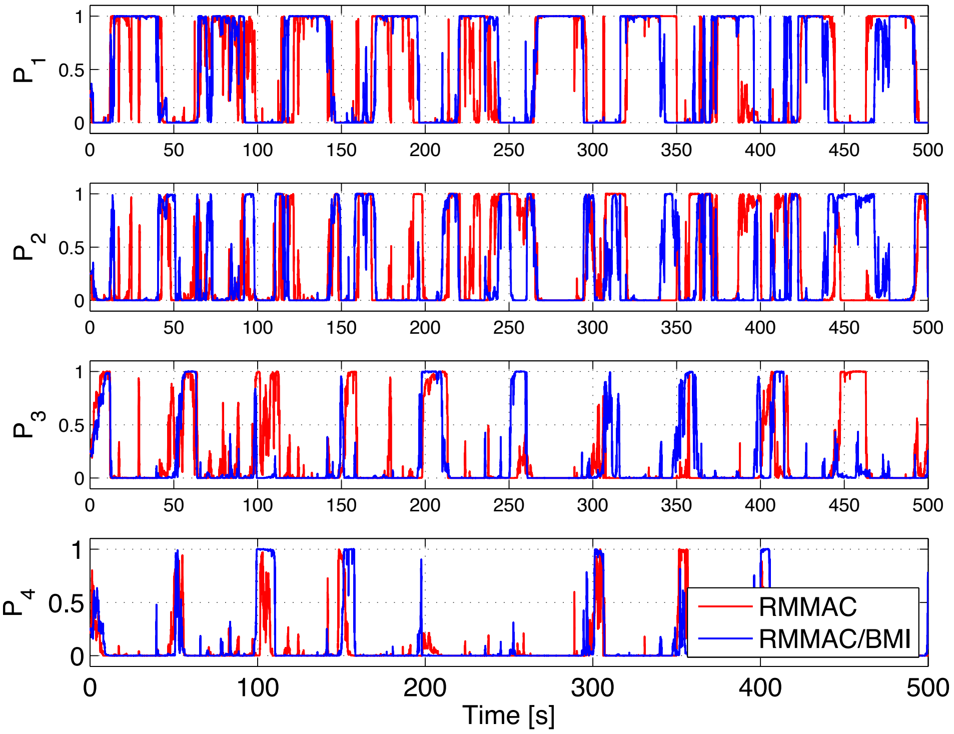


Figure 3.12: Case #3: posterior probabilities of each model.

Case #	k_1	RMS [m]			$\%E$	$\%F$
		RMMAC/BMI	RMMAC	GNARC		
1	0.9 N/m	0.0123	0.0128	0.0327	96.1%	37.6%
2	Fig. 3.7	0.0159	0.0165	0.0432	96.3%	36.8%
3	Fig. 3.10	0.0277	0.0330	0.0401	83.9%	69.2%

Table 3.2: Summary of the comparisons, in terms of RMS of the output, of the RMMAC/BMI versus the standard RMMAC

3.4 Conclusions

In this chapter, a comparison between the classical RMMAC and the RMMAC/BMI architectures was presented, by means of an example. We used the (double) mass-spring-dashpot testbed example to illustrate the benefits of using LPV/BMI controllers within the RMMAC architecture, since it represents a challenging control problem, due to the noncolocation of the control input with the performance variable.

The results obtained after several Monte-Carlo simulations are summarized in Table 3.2. In all the analyzed cases, the RMMAC/BMI performed better, in terms of minimization of the RMS of the output, than the standard RMMAC. The improvements due to the use of the LPV/BMI controllers are particularly apparent when the time-variations of the plant are very fast – see Case #3. As stressed in the previous chapter, the performance of the mixed-μ controllers can deteriorate significantly when the plant is not time-invariant. This is avoided by using LPV/BMI controllers, which can take into account the variations of the dynamics of the plant.

It is also important to notice that, in general, the use of adaptive control, in the present case, clearly increases the performance of the closed-loop system. Nevertheless, the decision subsystem may be responsible for the deterioration of the performance of the closed-loop – see [60, 61]. Indeed, this problem is going to be tackled in the following chapters. Nevertheless, in Case #3, due to the erroneous behavior of the decision subsystem, the use of a non-adaptive controller does not deteriorate the performance of the closed-loop as much as in the previous cases. However, this issue comes from the violation of the assumptions used to design the KFs of the decision subsystem, rather than from the control part itself. Indeed, this problem is going to be tackled in the following chapters.

As a final remark, this chapter presented no theoretical results. Therefore, no guarantees in terms of robust stability/performance are provided at this point, since the decision subsystem may not be able to *select* the appropriate controller. This fact motivated the development of the decision subsystems presented in the following chapters.

Chapter 4

The Stability Overlay

4.1 Introduction

Some adaptive control laws can lead to unstable closed-loop systems when connected to a plant with even the slightest discrepancies from the family of admissible plant models. This issue was first described in [39], in the so-called Rohrs *et al.* counterexample. Very small disturbances can be responsible for destabilizing the closed-loop because of the unavoidable unmodeled high frequency dynamics, present in every physical system.

The previous chapters stressed that, even when the control subsystem is robust against model uncertainty, multiple-model adaptive control strategies may not be able to stabilize the plant, if the decision subsystem is not guaranteed to converge to the *correct* model of the plant.

This chapter thus proposes a solution to the stability problem common to many closed-loop linear and nonlinear, time-invariant and time-varying, systems with performance-based adaptive control laws. The strategy developed herein, referred to as Stability Overlay (SO), takes into account both stability objectives – often robust to a very wide class of disturbances and model uncertainty – and performance requirements – that, in general, assume a stronger knowledge about the plant to be controlled. The algorithms presented in the sequel are based upon [68, 105] and were firstly introduced in [69, 70, 106]. They assess the "rewards" received by each controller after its most recent utilization, without any prior information on the bounds of the exogenous disturbances and sensors noise. A control law is then disqualified or not, based upon its rewards, in a similar way to what is done in [50, 51, 53, 54, 107–114] and in the references therein. However, in our approach, we suggest that the SO should only be responsible for the stability of the plant, and thus another algorithm should run in parallel in order to satisfy the posed performance requirements. Although the SO can be used as an adaptive control method *per se*, such an approach can lead to low levels in terms of performance, as explained in the sequel. Therefore, our methodology differs from the aforementioned ones, in the sense that the controllers rewards are not used to decide which controller leads to the highest closed-loop performance, but rather to guarantee that a control law which is not able to stabilize the plant is not persistently selected. Other solutions, such as the Lyapunov-based approach presented in [54], rely on the model of the plant and hence require stronger assumptions than the ones presented in the sequel.

For the proposed SO methodology, it is not required to know the plant model to be controlled nor the disturbance properties – see [115] for a model-free adaptive controller for linear time-invariant systems. Still, as shown in [113], it is clear that the performance of the closed-loop can be severely affected if no knowledge is available about the plant. Nonetheless, the model-free characteristic of the present method ensures robustness to several types of model uncertainty. In a sense, if the actual plant is *close* to a plant model in the family used to design the adaptive control law, then the adaptation runs as usual, without (or with minor) intervention of the SO. If, however, the actual plant or disturbances properties do not match the ones used during the design, the closed-loop system may become unstable. Therefore, instead of blindly continuing to use the adaptation law, we assess the norm of the inputs and outputs of the system, and eventually switch to a controller that is able to stabilize the plant, as long as such controller belongs to the set of *legal* controllers that the SO is allowed to use.

Control strategies such as the Robust Multiple-Model Adaptive Control (RM-MAC – see Chapter 1), the Multiple Model Adaptive Control with Mixing (MMACwM

– see [41, 100]), and the unfalsified control (see [50, 53, 107, 108, 110]), among others [51, 54], have successfully shown high levels of performance against complex and real uncertainties in the plant, although they can lead the closed-loop to instability if the uncertain parameters of the plant vary with time. This chapter, thus, proposes the use of the SO to ensure input/output stability of this kind of methodologies when applied to time-varying plants. We emphasize that adaptive control strategies should be able to handle time-varying plants, since these methodologies are usually applied to plants with drifting parameters, and hence guaranteeing stability of plants with time-frozen parameters is usually not enough.

Therefore, the SO can be seen as a *safety device* that can be used with many adaptive algorithms, achieving high levels of performance while providing robust stability guarantees for several different types of modeling errors. We show that the applicability of the SO is very wide, in the sense that it can be used in parallel with several adaptive control laws, as long as a few set of natural assumptions is satisfied, in particular that at least one controller in the set of *eligible* controllers is able to stabilize the plant.

The use of the SO is illustrated by a set of simulations, for both linear and nonlinear (possibly) time-varying plants. We show how to solve the instability issues in the well-known Rohrs *et. al.* counterexample [39] using the SO and how to augment the RMMAC architecture in order to obtain robust stability guarentees. However, we stress that the main result presented in the sequel is able to provide stability guarantees for a much wider family of performance-based adaptive laws, with a few changes in the original algorithms.

The problem of smoothly switching between the controllers is not addressed in this thesis. Nevertheless, this is an important topic in any practical application, as the transients generated during the switching of the controllers can severely deteriorate the performance of the closed-loop, even though stability can typically still be achieved – see, for instance, [116]. Therefore, several approaches to the problem of deriving the so-called bumpless transfer functions for switching controllers are available in the literature – for further details, the reader is referred to [116–120].

4.1.1 Main Contributions and Organization

To better understand the SO concept, we start by considering that the uncertain plant to be controlled is linear and time-invariant. We also assume that the (non-adaptive) local controllers are LTI. This allows us to have a deeper understanding on the difficulties that can arise in the selection of the appropriate controller, even though both the plant and the controllers are linear and time-invariant.

We also present an extension of the Stability Overlay, so that it can handle nonlinear and time-varying plants. Although the nonlinearities can obviously impact on the decision system, our main concern is the time-variations of the plant. In particular, it may happen that the *correct* controller at a given time instant, is a destabilizing controller at another time instant, due to the drifting of some parameters of the plant.

The main contributions of this chapter are as follows:

1. The derivation of a set of properties regarding the boundedness of LTI systems;

2. The development of the Stability Overlay algorithms for LTI and time-varying plants, presenting, for the first time, a theoretical proof that one can, at least in some cases, detect and correct instability in adaptive control schemes, for TV plants, without prior assumptions other than feasibility;

3. The application of the SO to the Rohrs' *et al.* counterexample;

4. The integration of the SO into the RMMAC/BMI control architecture.

The remainder of this chapter is organized as follows. We start by posing the problem in Subsection 4.1.2. In Section 4.2, some properties of LTI systems are derived. The main result for LTI plants is presented in Section 4.3. In Section 4.4, simulation results of the Rohrs *et al.* counterexample integrated with the SO are shown. In Section 4.5, the result of Section 4.3 is extended to a class of nonlinear time-varying plants. Simulation results of the integration of the SO with the RMMAC are presented in Section 4.6. Finally, in Section 4.7, some conclusions about the SO are discussed.

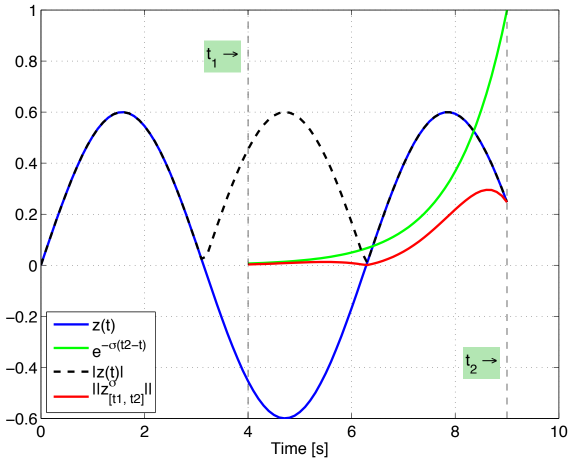


Figure 4.1: Discounted norm of signal $z(\cdot)$, between time instants t_1 and t_2.

4.1.2 Preliminaries

We define the discounted norm of signal $z(\cdot)$, for any $\sigma > 0$, as

$$\left\| z|^{\sigma}_{[t_1,t_2]} \right\| = \sup_{\tau \in [t_1,\, t_2]} e^{-\sigma(t_2-\tau)} |z(\tau)|,$$

and

$$\left\| z|_{[t_1,t_2]} \right\| = \sup_{\tau \in [t_1,\, t_2]} |z(\tau)|.$$

Figure 4.1 illustrates the discounted norm of a sinusoidal signal.

Although in the sequel we address a much more general family of plant models, at this point, we assume that the plant can be modeled by an LTI system described by

$$\dot{x} = Ax + Bu + F\xi, \quad x(0) = x_0 \tag{4.1a}$$

$$y = Cx + G\theta \tag{4.1b}$$

$$z = \begin{pmatrix} y \\ u \end{pmatrix} = \begin{pmatrix} Cx + G\theta \\ u \end{pmatrix} \tag{4.1c}$$

$$u = K_{\alpha(t)} y = \begin{pmatrix} K_{\alpha(t)} & 0 \end{pmatrix} z \tag{4.1d}$$

$$K_{\alpha(t)} \in S_o := \{K_1, K_2, \ldots, K_{N_c}\}\,. \tag{4.1e}$$

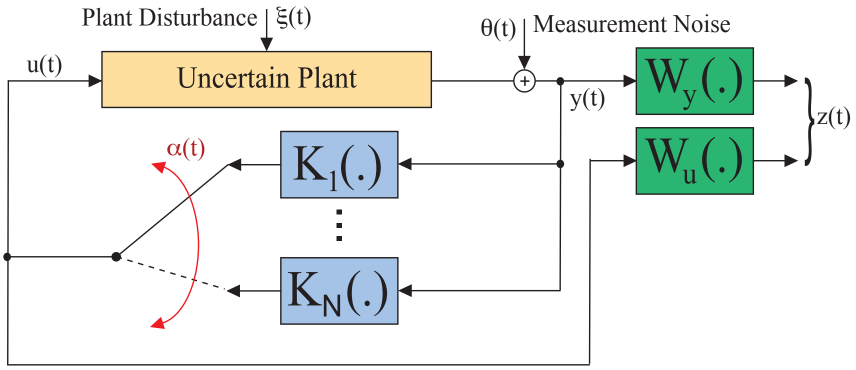


Figure 4.2: Feedback interconnection between the plant (4.1) and the controllers K_i, selected through signal $\alpha(t)$.

The output variables $z(\cdot)$ and $y(\cdot)$ can include performance outputs such as the ones obtained by filtering the plant output and the control input with the weights $W_y(s)$ and $W_u(s)$, respectively – see [121] and the example in Chapter 3 for more details on using performance weights. Furthermore, $x_0 \in \mathbb{R}^n$ is a fixed (but unknown) initial condition, $\xi(\cdot) \in \mathcal{L}_\infty$ is a bounded (but unknown) exogenous disturbance, $\theta(\cdot) \in \mathcal{L}_\infty$ is the bounded (but unknown) measurement noise and $u(\cdot)$ is the control input. S_o is the set of *eligible* controllers which are considered to be constant matrix gains, without loss of generality, as explained in the sequel. N is the number of legal control laws (and thus the number of elements in S_o), and K_i, for $i \in \{1, 2, ..., N\}$, represents a controller. We argue that the control laws of any adaptive or non-adaptive system should be robust against model uncertainty.

Define a *finitely switching* control input as

$$u_{\mathrm{fs}}(t) = \begin{cases} K_{\alpha(t)}(y(t)), & 0 \le t < t_o; \\ K_*(y(t)), & t \ge t_o. \end{cases} \tag{4.2}$$

Figure 4.2 depicts the output feedback interconnection between the plant and the controllers K_i, selected through signal $\alpha(t)$.

4.2 Properties of LTI Closed-Loop Systems

In this section, we derive some properties of LTI closed-loop systems that are going to be useful in the sequel. Consider the LTI plant described by (4.1). Without loss of generality, we assume that the N controllers in S_o are static output feedback controllers, *i.e.*, each K_i, for $i \in \{1, \ldots, N\}$, is a constant matrix. Notice that, as shown in [68, 105] if the controllers are dynamic and described by

$$\dot{x}_c = A_c x_c + B_c y,$$
$$u = C_c x_c + D_c y,$$

then, for each controller, one can rewrite the closed-loop system as

$$\begin{pmatrix} \dot{x} \\ \dot{x}_c \end{pmatrix} = \begin{pmatrix} A & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x \\ x_c \end{pmatrix} + \begin{pmatrix} B & 0 \\ 0 & I \end{pmatrix} u_{\text{aug}} + \begin{pmatrix} F \\ 0 \end{pmatrix} \xi,$$
$$z_{\text{aug}} = \begin{pmatrix} C & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} x \\ x_c \end{pmatrix} + \begin{pmatrix} G \\ 0 \end{pmatrix} \theta, u_{\text{aug}} = \begin{pmatrix} D_c & C_c \\ B_c & A_c \end{pmatrix} z_{\text{aug}}.$$

In this form, switching between dynamic controllers means switching the “static output feedback” matrix $\begin{pmatrix} D_c & C_c \\ B_c & A_c \end{pmatrix}$ which only depends on controller matrices. The only restriction is that all dynamic controllers in this setup must be of the same order, which in practice does not represent a shortcoming, since we can solve that issue by having some controllers that use only a subset of the available states, while forcing the remaining ones to go to zero.

We follow closely the steps in [68, 105] to show that, under mild assumptions, the linear system (4.1) has the following properties:

Property 4.1. *For any finitely switching input* (4.2) *and any* t_o, ΔT, $\sigma > 0$,

$$\left\| z|_{[0,t_o]}^{\sigma} \right\| < \infty \Rightarrow \left\| z|_{[0,t_o+\Delta T]}^{\sigma} \right\| < \infty,$$

i.e.*, the closed-loop does not have a finite escape time.*

Property 4.2 (Input/Output Stability)**.** *There exist a control law,* K_**, and positive constants* σ *and* l^**, such that for any* $0 < \gamma < 1$*, there exists a* $\Delta T^* \geq 0$ *that satisfies* *the following condition. For any finitely switching control input* (4.2)*,*

$$\left\| z|^{\sigma}_{[0,t_o+\Delta T]} \right\| \leq \gamma \left\| z|^{\sigma}_{[0,t_o]} \right\| + l^*,$$

for all $\Delta T \geq \Delta T^*$ *and* $t_o \geq 0$*. The closed-loop system with control law* K_* *is said* input/output stable.

Property 4.1 ensures there are no controllers in the *legal* set (referred to as *eligible* controllers) that lead the output of the plant to infinity in finite time. Finally, Property 4.2 states that there is at least one eligible controller that satisfies a desired stabilization condition. Parameter l^* accounts for the exogenous disturbances and the initial conditions.

Consider the following assumption:

Assumption 4.1: The switched linear system (4.1) satisfies:

- $\Re\left\{\lambda_j(A+BK_iC)\right\} < 0, \underset{j}{\forall}$ for some $i \in \{1,\ldots,N\}$.
- The pair $[A,\, C]$ is observable.
- The exogenous disturbances and the measurement noise are bounded by some (possibly unknown) constants ξ_0 and θ_0, respectively, *i.e.*, $|\xi(\cdot)| \leq \xi_0$ and $|\theta(\cdot)| \leq \theta_0$.

□

Proposition 4.1. *Under Assumption 4.1, the linear system* (4.1) *has the Properties 4.1–4.2.*

Next, the sketch of the proof of Proposition 4.1 is presented.

The pair $[A,\, C]$ is observable. Therefore, we can build a state observer of the form

$$\dot{\hat{x}} = (A-LC)\hat{x} + \begin{pmatrix} L & B \end{pmatrix} z, \quad \hat{x}(0) = 0.$$

The observer gain L is such that, for some m_0, $\lambda_0 > 0$,

$$\left\| e^{(A-LC)t} \right\| \leq m_0 e^{-\lambda_0 t}.$$

Remark 4.1: We can relax the assumption on the observability of the pair $[A, C]$ by considering that it is only detectable. In that case, however, the value of λ_0 (and, thus, of σ) is constrained by the slowest non-observable mode of the system. $\diamond$

Let $\sigma < \lambda_0$. Then, we can bound the size of the observer state, $|\hat{x}(t)|$, by

$$\begin{aligned} |\hat{x}(t)| &\leq \int_o^t m_0 e^{-\lambda_0(t-\tau)} \left\| \begin{pmatrix} L & B \end{pmatrix} \right\| |z(\tau)| \, d\tau \\ &= \int_o^t m_0 e^{-(\lambda_0-\sigma)(t-\tau)} \left\| \begin{pmatrix} L & B \end{pmatrix} \right\| e^{-\sigma(t-\tau)} |z(\tau)| \, d\tau \\ &\leq \frac{m_0 \left\| \begin{pmatrix} L & B \end{pmatrix} \right\|}{\lambda_0 - \sigma} \left\| z|^{\sigma}_{[0,\,t]} \right\| . \end{aligned} \tag{4.3}$$

Furthermore, we can write

$$\dot{\hat{x}}(t) - \dot{x}(t) = (A - LC)(\hat{x}(t) - x(t)) + LG\theta(t) - F\xi(t).$$

Hence,

$$|\hat{x}(t) - x(t)| \leq m_0 e^{-\lambda_0 t} |x(0)| + \frac{m_0}{\lambda_0} \|F\| \left\| \xi|_{[0,\,t]} \right\| + \frac{m_0}{\lambda_0} \|L\| \|G\| \left\| \theta|_{[0,\,t]} \right\| . \tag{4.4}$$

Combining (4.3) with (4.4) results in

$$\begin{aligned} |x(t)| \leq & \frac{m_0 \left\| \begin{pmatrix} L & B \end{pmatrix} \right\|}{\lambda_0 - \sigma} \left\| z|^{\sigma}_{[0,\,t]} \right\| + m_0 e^{-\lambda_0 t} |x(0)| \\ & + \frac{m_0}{\lambda_0} \|F\| \left\| \xi|_{[0,\,t]} \right\| + \frac{m_0}{\lambda_0} \|L\| \|G\| \left\| \theta|_{[0,\,t]} \right\| \\ \leq & \frac{m_0 \left\| \begin{pmatrix} L & B \end{pmatrix} \right\|}{\lambda_0 - \sigma} \left\| z|^{\sigma}_{[0,\,t]} \right\| + m_0 e^{-\lambda_0 t} |x(0)| + \frac{m_0}{\lambda_0} \left(\|F\| \xi_0 + \|L\| \|G\| \theta_0 \right) . \end{aligned} \tag{4.5}$$

Let $u_{\text{fs}}(t)$ be a finitely switching control input as in (4.2). Let t_o denote the time of the last switch, and let K_i be the final gain matrix. There exist positive constants m_{fb} and λ_{fb} such that, for any K_i,

a) $\|C\| \leq m_{\text{fb}}$,

b) $\|K_i\| \leq m_{\text{fb}}$,

c) $\left\| e^{(A+BK_iC)t} \right\| \leq m_{\text{fb}} e^{\lambda_{\text{fb}} t}$.

From these bounds, for any $t \geq t_o$, we obtain,

$$\begin{aligned}|z(t)| &= \left|\begin{pmatrix} Cx(t) + G\theta(t) \\ K_{\alpha(t)}Cx(t) + K_{\alpha(t)}G\theta(t)\end{pmatrix}\right| \\ &\leq (1+m_{\text{fb}})\Bigg[m_{\text{fb}}^2 e^{\lambda_{\text{fb}}(t-t_o)}\,|x(t_o)| \\ &+ \int_{t_o}^{t} m_{\text{fb}}^2 e^{\lambda_{\text{fb}}(t-\tau)}\|F\|\,|\xi(\tau)|\,d\tau + \|G\|\,|\theta(t)|\Bigg] \end{aligned} \tag{4.6}$$

Then,

$$\begin{aligned}\left\|z|^{\sigma}_{[0,\,t_o+\Delta T]}\right\| &= \sup_{\tau\in[0,\,t_o+\Delta T]} e^{-\sigma(t_o+\Delta T-\tau)}\,|z(\tau)| \\ &\leq \sup_{\tau\in[0,\,t_o+\Delta T]} \big[m_{\text{fb}}^2 e^{-\sigma(t_o+\Delta T-\tau)}e^{\lambda_{\text{fb}}(\tau-t_o)}|x(t_o)| \\ &+m_{\text{fb}}^2\int_{t_o}^{\tau} e^{\lambda_{\text{fb}}(\tau-\mu)}e^{-\sigma(t_o+\Delta T-\tau)}\|F\||\xi(\mu)|d\mu + \|G\||\theta(\tau)|e^{-\sigma(t_o+\Delta T-\tau)}\Big](1+m_{\text{fb}}) \\ &\leq \sup_{\tau\in[0,\,t_o+\Delta T]} \big[m_{\text{fb}}^2 e^{(\lambda_{\text{fb}}+\sigma)(\tau-t_o)}e^{-\sigma\Delta T}|x(t_o)| \\ &+m_{\text{fb}}^2\int_{t_o}^{\tau} e^{(\lambda_{\text{fb}}+\sigma)\tau}e^{-\sigma(t_o+\Delta T)}e^{-\lambda_{\text{fb}}\mu}\|F\||\xi(\mu)|d\mu + \|G\|\theta_0\Big](1+m_{\text{fb}}) \\ &\leq \left[m_{\text{fb}}^2 e^{\lambda_{\text{fb}}\Delta T}|x(t_o)| + m_{\text{fb}}^2\frac{e^{\lambda_{\text{fb}}\Delta T}}{\lambda_{\text{fb}}}\|F\|\xi_0 + \|G\|\theta_0\right](1+m_{\text{fb}})\,. \end{aligned} \tag{4.7}$$

Using (4.5) evaluated at $t = t_o$, we obtain

$$\begin{aligned}\left\|z|^{\sigma}_{[0,\,t_o+\Delta T]}\right\| \leq \Bigg\{ & m_{\text{fb}}^2 e^{\lambda_{\text{fb}}\Delta T}\Bigg[\frac{m_0\left\|\begin{pmatrix} L & B\end{pmatrix}\right\|}{\lambda_0-\sigma}\left\|z|^{\sigma}_{[0,\,t_o]}\right\| \\ & +m_0 e^{-\lambda_0 t_o}\,|x(0)| + \frac{m_0}{\lambda_0}\left(\|F\|\xi_0 + \|L\|\|G\|\theta_0\right)\Bigg] \\ & + m_{\text{fb}}^2\frac{e^{\lambda_{\text{fb}}\Delta T}}{\lambda_{\text{fb}}}\|F\|\xi + \|G\|\theta_0\Bigg\}(1+m_{\text{fb}}).\end{aligned}$$

Therefore, defining $\sigma_1 : \mathbb{R}^+ \to \mathbb{R}^+$ and $\sigma_2 : \mathbb{R}^+ \to \mathbb{R}^+$ as

$$\begin{aligned}\sigma_1(\Delta T) &= m_{\text{fb}}^2 e^{\lambda_{\text{fb}}\Delta T}\frac{m_0\left\|\begin{pmatrix} L & B\end{pmatrix}\right\|}{\lambda_0-\sigma}(1+m_{\text{fb}}), \\ \sigma_2(\Delta T) &= \Bigg\{m_{\text{fb}}^2 e^{\lambda_{\text{fb}}\Delta T}\Bigg[m_0 e^{-\lambda_0 t_o}|x(0)| \\ &+\frac{m_0}{\lambda_0}\left(\|F\|\xi_0 + \|L\|\|G\|\theta_0\right) + \frac{1}{\lambda_{\text{fb}}}\|F\|\xi_0\Bigg] + \|G\|\theta_0\Bigg\}(1+m_{\text{fb}}),\end{aligned}$$

one concludes that

$$\left\| z|^{\sigma}_{[0,\, t_o+\Delta T]} \right\| \le \sigma_1(\Delta T) \left\| z|^{\sigma}_{[0,\, t_o]} \right\| + \sigma_2(\Delta T),$$

i.e., the closed-loop does not have a finite escape time and thus Property 4.1 is satisfied.

Notice that the only restriction on σ is that is satisfies $\sigma < \lambda_0$. The observability assumption implies that the analysis can be carried out with any λ_0 by appropriately constructing L. Accordingly, σ can be arbitrarily chosen.

To show that Property 4.2 is also satisfied, let us assume that $A + BK_*C$ is a stability matrix, *i.e.*, $\Re\{\lambda_j(A + BK_*C)\} < 0, \underset{j}{\forall}$, satisfying

$$\left\| e^{(A+BK_*C)t} \right\| \le m_s e^{-\lambda_s t} \tag{4.8}$$

for some strictly positive constants m_s and λ_s. As in (4.6), we can bound the output magnitude (but this time using (4.8)) by

$$|z(t)| \le (1 + m_{\text{fb}}) \left(m_{\text{fb}} m_s e^{-\lambda_s(t-t_o)} |x(t_o)| + \frac{m_{\text{fb}} m_s}{\lambda_s} \|F\| \left\| \xi|_{[t_o,\, t]} \right\| + \|G\| |\theta(t)| \right).$$

Hence,

$$\begin{aligned}
\left\| z|^{\sigma}_{[t_o,\, t_o+\Delta T]} \right\| \le & \sup_{\tau \in [t_o,\, t_o+\Delta T]} e^{-\sigma(t_o+\Delta T-\tau)} (1 + m_{\text{fb}}) \\
\times & \left[m_{\text{fb}} m_s e^{-\lambda_s(\tau-t_o)} |x(t_o)| + \frac{m_{\text{fb}} m_s}{\lambda_s} \|F\| \xi_0 + \|G\| \theta_0 \right] \\
= & \sup_{\tau \in [0,\, \Delta T]} e^{-\sigma(\Delta T-\tau)} (1 + m_{\text{fb}}) \\
\times & \left[m_{\text{fb}} m_s e^{-\lambda_s \tau} |x(t_o)| + \frac{m_{\text{fb}} m_s}{\lambda_s} \|F\| \xi_0 + \|G\| \theta_0 \right] \\
= & \sup_{\tau \in [0,\, \Delta T]} (1 + m_{\text{fb}}) \\
\times & \left[e^{-\sigma \Delta T} m_{\text{fb}} m_s e^{(\sigma-\lambda_s)\tau} |x(t_o)| + e^{-\sigma(\Delta T-\tau)} \left(\frac{m_{\text{fb}} m_s}{\lambda_s} \|F\| \xi_0 + \|G\| \theta_0 \right) \right].
\end{aligned}$$

Let $\zeta = \min\{\sigma, \lambda_s\}$. Then, the previous inequality results in

$$\left\| z|^{\sigma}_{[t_o,\, t_o+\Delta T]} \right\| \le (1 + m_{\text{fb}}) \left[e^{-\zeta \Delta T} m_{\text{fb}} m_s |x(t_o)| + \frac{m_{\text{fb}} m_s}{\lambda_s} \|F\| \xi_0 + \|G\| \theta_0 \right]. \tag{4.9}$$

Moreover,

$$\begin{aligned}\left\|z|^{\sigma}_{[0,\,t_o+\Delta T]}\right\| &= \max\left(e^{-\sigma\Delta T}\left\|z|^{\sigma}_{[0,\,t_o]}\right\|,\ \left\|z|^{\sigma}_{[t_o,\,t_o+\Delta T]}\right\|\right)\\ &\le e^{-\sigma\Delta T}\left\|z|^{\sigma}_{[0,\,t_o]}\right\| + \left\|z|^{\sigma}_{[t_o,\,t_o+\Delta T]}\right\|. \end{aligned} \tag{4.10}$$

Therefore, we conclude that

$$\begin{aligned}\left\|z|^{\sigma}_{[0,\,t_o+\Delta T]}\right\| &\le e^{-\sigma\Delta T}\left\|z|^{\sigma}_{[0,\,t_o]}\right\| + (1+m_{\text{fb}})\\ &\times\left(e^{-\zeta\Delta T}m_{\text{fb}}m_s\,|x(t_o)| + \frac{m_{\text{fb}}m_s}{\lambda_s}\|F\|\xi_0 + \|G\|\theta_0\right).\end{aligned}$$

The value of $|x(t_o)|$ can be bounded using (4.5), *i.e.*,

$$|x(t_o)| \le \frac{m_0\left\|\begin{pmatrix}L & B\end{pmatrix}\right\|}{\lambda_0-\sigma}\left\|z|^{\sigma}_{[0,\,t_o]}\right\| + m_0e^{-\lambda_0t_o}\,|x(0)| + \frac{m_0}{\lambda_0}\left(\|F\|\xi_0 + \|L\|\|G\|\theta_0\right).$$

Thus,

$$\left\|z|^{\sigma}_{[0,\,t_o+\Delta T]}\right\| \le \gamma\left\|z|^{\sigma}_{[0,\,t_o]}\right\| + l^*,$$

where

$$\gamma = e^{-\sigma\Delta T} + (1+m_{\text{fb}})\,m_{\text{fb}}m_s\frac{m_0\left\|\begin{pmatrix}L & B\end{pmatrix}\right\|}{\lambda_0-\sigma}e^{-\zeta\Delta T} \tag{4.11a}$$

$$\begin{aligned} l^* &= (1+m_{\text{fb}})\Bigg\{e^{-\zeta\Delta T}m_{\text{fb}}m_sm_0\Bigg[e^{-\lambda_0\Delta T}|x(0)|\\ &+ \frac{\|F\|\xi_0+\|L\|\|G\|\theta_0}{\lambda_0}\Bigg] + \frac{m_{\text{fb}}m_s}{\lambda_s}\|F\|\xi_0 + \|G\|\theta_0\Bigg\}.\end{aligned} \tag{4.11b}$$

We conclude that Property 4.2 is satisfied for sufficiently large ΔT.

Remark 4.2: The discounted norm $\left\|z|^{\sigma}_{[0,\,t_o+\Delta T]}\right\|$ can be interpreted as a state-norm estimator (cf. [122, 123]). Although in a different perspective, these norm estimators are first order systems where the inputs are the discounted norms of the inputs and outputs of the plant. Therefore, our decisions of *disqualifying* or not a controller, according to this interpretation, will be based upon the estimate of the norm of the actual state of the closed-loop system. ◇

4.3 Stabilty Overlay Algorithm #1 for LTI Plants

The reward, $r(n)$, after using controller $K_p(\cdot)$ during the time interval $t_{n-1} \leq t < t_n$ is defined as

$$r(n) = \begin{cases} 1, & \left\| z|^{\sigma}_{[0,\,t_n]} \right\| \leq \gamma \left\| z|^{\sigma}_{[0,t_{n-1}]} \right\| + l(n) \\ 0, & \text{otherwise,} \end{cases} \tag{4.12}$$

where γ is a fixed scalar with $0 < \gamma < 1$ and $l(n)$ is going to be specified next.

Remark 4.3: We recall that $z(\cdot)$ may be an augmented and/or filtered output. In particular, during the design of the non-adaptive controllers, an output performance variable is typically defined as a function of a weighted combination of states and/or control input channels. This output performance variable can indeed be used as $z(\cdot)$, *i.e*, the rewards of the controllers can be computed using this output performance variable, rather than the actual output of the plant. $\diamond$

Figure 4.3 describes the Stability Overlay (SO) Algorithm #1, for LTI plants. The notation $S = S \backslash K(n)$ means "the exclusion of element $K(n)$ from set S". The initial set of eligible control laws is denoted S_o, while K_o is the first control law selected, $\Delta T(n)$ is the time-interval while the control law $K(n)$ is used, l_o is the initial value of $l(n)$ in (4.12), and l_{inc} and ΔT_{inc} are the increments for $l(n)$ and $\Delta T(n)$, respectively, whenever all the control laws have *failed* in their most recent utilization. The reasoning for the sequence of increasing dwell-times $\Delta T(n)$ – also used in [113] to provide stability guarantees for the closed-loop system – is going to be explained in what follows (see Remark 4.6).

As further stressed in the sequel, the selection of the control law among the set of eligible ones, S, can be done according to any other (probably performance-based) adaptive control algorithm, which may take into account the model of the plant. Regarding the initializations of the SO Algorithm #1, we can summarize the calculations suggested, based upon Section 4.2:

- choose arbitrary $\sigma > 0$;
- choose arbitrary positive $\gamma < 1$;
- choose values for ΔT_o, l_o.

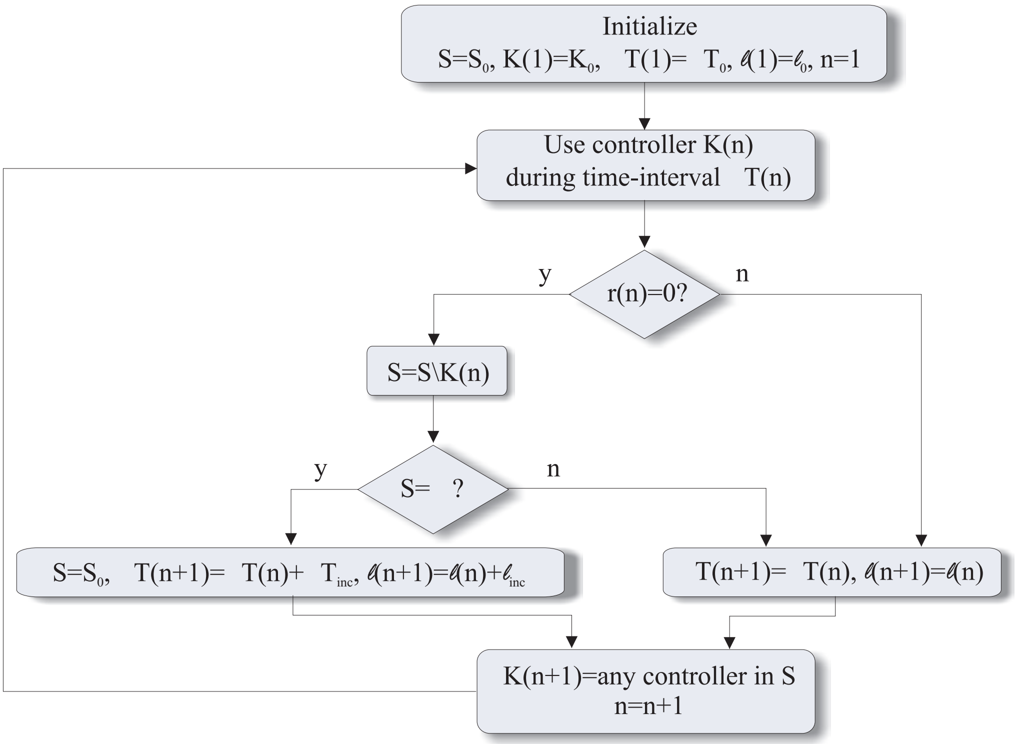

Initialize
S=S$_0$, K(1)=K$_0$, T(1)= T$_0$, ℓ(1)=ℓ$_0$, n=1
Use controller K(n)
during time-interval T(n)
y
r(n)=0?
n
S=S\K(n)
y
S= ?
n
S=S$_0$, T(n+1)= T(n)+ T$_{inc}$, ℓ(n+1)=ℓ(n)+ℓ$_{inc}$
T(n+1)= T(n), ℓ(n+1)=ℓ(n)
K(n+1)=any controller in S
n=n+1

Figure 4.3: Stability Overlay (SO) Algorithm #1, for LTI plants.

Nevertheless, the values for ΔT_o and l_o can be arbitrarily small, without endangering the stability of the closed-loop.

Remark 4.4: The number of controllers, N, is not relevant in terms of stability, as long as the initial set of control laws, S_o, contains a stabilizing controller. However, larger values of N may require longer searching periods, before a stabilizing controller is found. This obviously may deteriorate substantially the performance of the closed-loop. $\diamond$

Theorem 4.1. *If Assumption* 1 *is satisfied, the Stability Overlay Algorithm* #1, *for LTI plants, results in* $\left\| z|^{\sigma}_{[0,t]} \right\|$ *bounded.*

Proof of Theorem 4.1

We start by considering Claim 1 in [68, 105]:

Claim 4.1. *The parameters $\Delta T(n)$ and $l(n)$ are uniformly bounded,* i.e.,

$$\begin{aligned} \Delta T_{max} &:= \lim_{n\to\infty} \Delta T(n) < \infty \\ l_{max} &:= \lim_{n\to\infty} l(n) < \infty \end{aligned} \tag{4.13}$$

Proof. The parameters $\Delta T(n)$ and $l(n)$ are increased whenever every control law, K_i, in its most recent utilization resulted in a zero reward, *i.e.*,

$$\left\| z|^{\sigma}_{[0,\,t_n]} \right\| > \gamma \left\| z|^{\sigma}_{[0,t_{n-1}]} \right\| + l(n). \tag{4.14}$$

However, by Property 4.2, there exists a $\Delta T^* \geq 0$, a positive constant l^*, and at least one control law which satisfies the condition

$$\left\| z|^{\sigma}_{[0,t_o+\Delta T]} \right\| \leq \gamma \left\| z|^{\sigma}_{[0,t_o]} \right\| + l^*, \tag{4.15}$$

provided that $\Delta T(n) \geq \Delta T^*$. This implies that the condition (4.14) cannot be satisfied infinitely often with $\Delta T(n)$ and $l(n)$ increasing without bound. □

According to Claim 1, there is at least one control law that is going to be used infinitely many times. For this control law, $r = 1$ in (4.12). Thus, all other control laws are going to be used at most a finite number of times. According to Proposition 4.1, the output is going to remain bounded during that (finite) time interval where $r = 0$. For some t_o, the rewards obtained for $t \geq t_o$ are positive. Since the output is bounded at $t = t_o$, it will remain bounded for $t > t_o$. This means that the closed-loop system is input/output stable, which concludes the proof. □

It is important to stress that we do not describe how to choose the controller to be connected to the loop. In fact, we allow any controller in set S (see Fig. 4.3) to be selected. The choice of the controller is responsible for the performance of the closed-loop and should be taken care of by some adaptive control law that (probably) takes into account the model of the plant and the disturbances properties. One example of the applicability of the SO with a model reference adaptive control architecture is presented in the following section.

However, as shown in the sequel, the applicability of the SO is much wider. Many types of adaptive laws are eligible to be integrated with the SO, such as the schemes based on the identification of the plant parameters (see, for instance, [27, 100] and references therein), or the estimator-based methodologies in [31].

Remark 4.5: It should be noted that the choice of the parameters for the algorithm may be very sensitive in some cases, depending upon the plant dynamics and the disturbances intensity. In fact, if the norm of the output of the closed-loop system grows *very fast* whenever a destabilizing controller is picked, and if the time required to disqualify a controller is very large, one may not get "practical stability". This means that, although a stabilizing controller is eventually selected, the transients may not be reasonable from a practical point of view. Furthermore, this effect may be exacerbated if we randomly pick the controllers in S, instead of using a performance-based algorithm to select among the eligible control laws. $\diamond$

Remark 4.6: The reason for increasing ΔT can be explained in a very intuitive manner, that relates it to the classical performance/robustness tradeoffs. If ΔT is large enough, stabilizing controllers are not going to be disqualified. However, we may also be using destabilizing controllers for a long time, since we only switch to another one after at least a time interval ΔT. In case ΔT is very small, we may find the *right* controller faster, but we may also disqualify stabilizing controllers just because they were not used long enough. Therefore, large values of ΔT guarantee stability at the cost of large transients, while smaller values of ΔT can lead to smaller transients (higher performance) at the cost of reduced robustness. $\diamond$

Remark 4.7: The SO can also be interpreted in light of the so-called safe adaptive control considered in [53, 108–112], all of which are based on the application of the Morse-Mayne-Goodwin [114] hysteresis switching algorithm with *cost-detectable* cost functions. In fact, what we call here *disqualified controllers* is called *falsified controllers* in [108], while the *eligible/legal controllers* correspond to the *candidate* controllers. The fact that a controller in S_o is able to stabilize the plant (Property 4.2) is referred to as the *feasibilty condition* in [108]. Finally, if we use *the number of times the pair* $(\Delta T(n), l(n))$ *has been incremented*, as a cost function, then it would be *cost-detectable* in the sense of [108], since this cost function would tend to infinity if the closed-loop is not stable – see Claim 4.1. $\diamond$

4.4 Rohrs et al. Counterexample

To illustrate the usefulness of the SO for LTI plants, we use the so-called Rohrs *et al.* counterexample – see [39], where the concept of “robustness” was introduced to the then existing literature. We use the reference model adaptation law referred to as Continuous-Time Algorithm 1 and the same terminology as in [39]. Figure 4.4 depicts the architecture of this methodology.

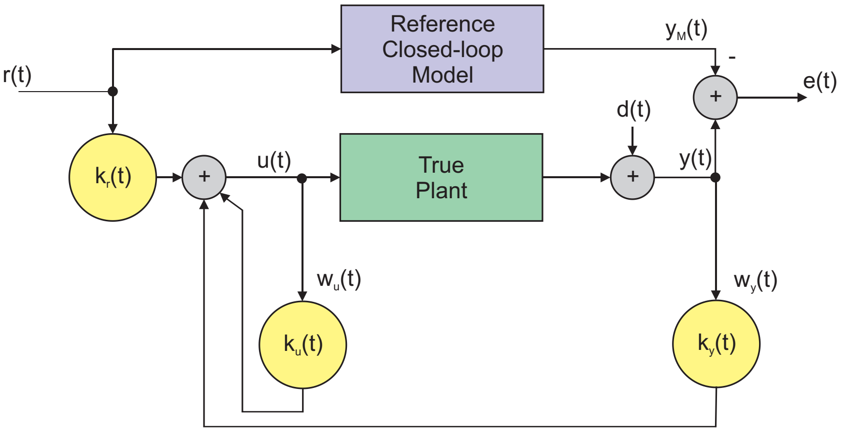


Figure 4.4: Continuous-Time Algorithm 1

The reference input is denoted by $r(t)$, the control input by $u(t)$, the disturbances by $d(t)$, and the output by $y(t)$. Signal $y_M(t)$ is the output of the reference model. The adaptive gains are denoted by $k_r(t)$, $k_u(t)$ and $k_y(t)$. Let $k(t) = \left[k_r^T(t) \quad k_u^T(t) \quad k_y^T(t)\right]^T$ and $w(t) = \left[r^T(t) \quad w_u^T(t) \quad w_y^T(t)\right]^T$, where $w_u(t)$ and $w_y(t)$ are defined in the sequel. The adaptation law evolves according to

$$\dot{k}(t) = -\Gamma w(t) e(t),$$

where $\Gamma = \Gamma^T > 0$ and

$$e(t) = y(t) - y_M(t).$$

We use the example in [39] where the plant model is given by

$$Y(s) = \left(\frac{2}{s+1}\right) \frac{229}{s^2 + 30s + 229} U(s), \tag{4.16}$$

the reference model is described by

$$Y_M(s) = \frac{3}{s+3} R(s), \tag{4.17}$$

and

$$k_y(0) = -0.65 \quad \text{and} \quad k_r(0) = 1.14. \tag{4.18}$$

Notice that, according to the design procedure, $k_u(t) = 0$. Moreover, $w_u(t) = u(t)$ and $w_y(t) = y(t)$. Let the reference be given by

$$r(t) = 2$$

and the output additive disturbance by

$$d(t) = \frac{1}{2} \sin(8t).$$

The first step in order to apply the SO is to discretize and bound the gains k_y and k_r, so that we have a finite number of controllers. Each pair (k_y, k_r) defines a controller. Therefore, if k_y is divided into n_y bins and k_r is divided into n_r bins, the set S_o will have $n_r n_y$ controllers.

Another design decision has to be made regarding what to do when the adaptive control law choses a controller that was previously disqualified, because it received a zero reward. A simple approach is to use the controller *closest* to that one, in a geometric sense, although it may not be very effective in terms of performance. Another strategy is to decrease the gain if it was increasing before a disqualified controller was obtained, and vice-versa. *We stress that a specific strategy need not be used to guarantee stability. The boundedness of the output $y(t)$ is guaranteed a priori by the use of the SO, no matter how we schedule the controllers in the SO algorithm, so the only aspect the control engineer has to account for is the performance.*

4.4.1 Simulation without the Stability Overlay

In this subsection, we replicate the results in [39], just for comparison purposes. Figures 4.5 and 4.6 illustrate that the closed-loop system becomes unstable as time goes by, due to the unmodeled dynamics, excited by the disturbances, that were not accounted for during the design of the adaptive control law. The infinite gain operators [39] from $e(t)$ to $u(t)$ and from $e(t)$ to $k(t)$, inherently present in the Continuous-Time Algorithm (CA) 1, are responsible for the instability of the closed-loop.

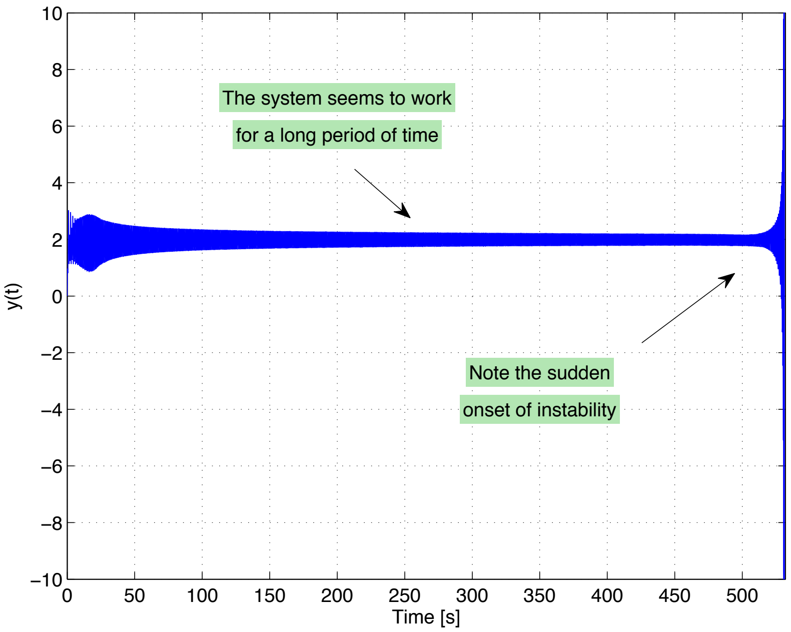


Figure 4.5: Output $y(t)$ of the closed-loop system using the continuous-time algorithm 1, without the stability overlay. Note that the system seems to work for a long period of time, while suddenly instability is observed.

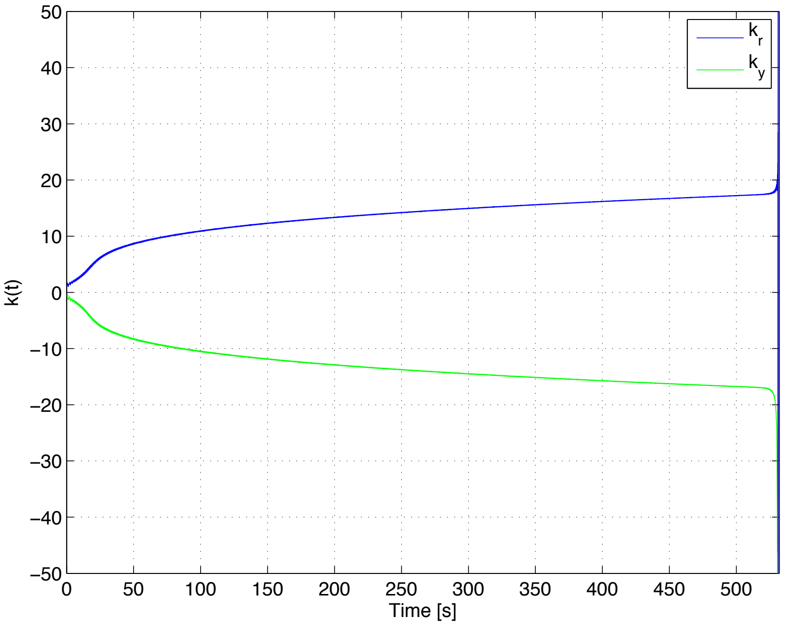


Figure 4.6: Time-evolution of the adaptive gains for the continuous-time algorithm 1, without the stability overlay. Note that the gains go to infinity in a sudden manner.

4.4.2 Simulation with the Stability Overlay

We now analyze the continuous-time algorithm supervised by the SO. The continuous gains k_r and k_y are discretized and bounded, in order to have a finite set of eligible controllers. For this simulation, we use bins of width 2 and the limits are defined as ± 50. In other words, we only allow the gains to take values in the set $K_r \times K_y$, *i.e.*,

$$(k_r, k_y) \in K_r \times K_y,$$

where

$$K_r := K_y := \{-50, -48, \cdots, -2, 0, 2, \cdots, 48, 50\}.$$

Figure 4.7 shows that the closed-loop is now stable, although some transients are also experienced, as explained in the sequel. The time instants when a controller is disqualified are also represented. If a disqualified controller is selected by the adaptive algorithm, the values of the adaptive gains k_r and k_y are updated to the *legal* ones closest to those obtained by the adaptive law.

The bursting phenomenon observed in Fig. 4.7 is due to the switching among the controllers and to the use (during a certain amount of time) of controllers that are not able to stabilize the plant. This can be avoided, for instance, resorting to fictitious signal cost-functions (*c.f.*, [50, 51, 54, 108, 109, 111, 112]), which can be used to disqualify control laws without inserting them into the feedback loop.

Figure 4.8 depicts the time-evolution of the adaptive gains. It should be noticed that, as soon as the instability is detected, the SO disqualifies the currently used controller and switches to another one. After only two switches, a controller is selected that stabilizes the plant. Due to the exogenous disturbances, the output error of the plant is not zero[1].

We stress that the adaptive gains in Fig. 4.8 are the ones effectively used in the feedback loop, and that they can differ from those obtained with the adaptive law, since we use discretized versions of these gains. The latter are only reset by the SO after a failure on the adaptive control law.

[1] The time-scale in Fig. 4.8 does not allow one to discern the time-variations of the output for $t > 1600$ s.

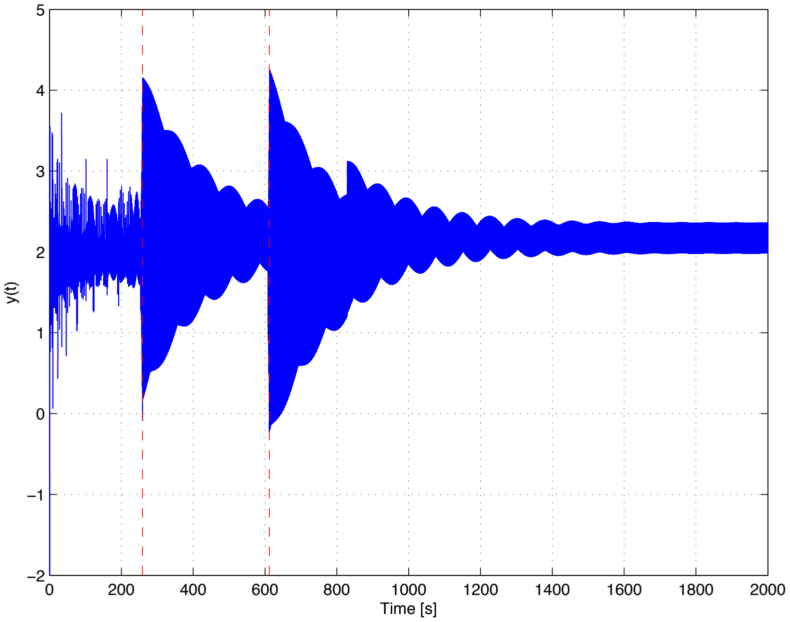


Figure 4.7: Output $y(t)$ of the closed-loop system using the continuous-time algorithm 1, with the stability overlay. The red dashed lines indicate the time instants when the currently scheduled controller fails.

Remark 4.8: Figure 4.8 indicates that one could simply saturate the adaptive gains and still get the stability result. However, if we do so, and use very large limits for the saturation, we can get an unstable system. On the other hand, if the limits are very small, we can get poor performance. The SO obtains those limits in a natural way, as shown in these simulations. $\diamond$

4.5 Stability Overlay for Nonlinear TV Plants

In what follows, the results presented in Section 4.3 are extended to a class of nonlinear time-varying plants. Although the key ideas of the Stability Overlay for the LTI case naturally arise in this more general framework, we must also take into account other factors that can affect stability, such as the time-variations of the plant.

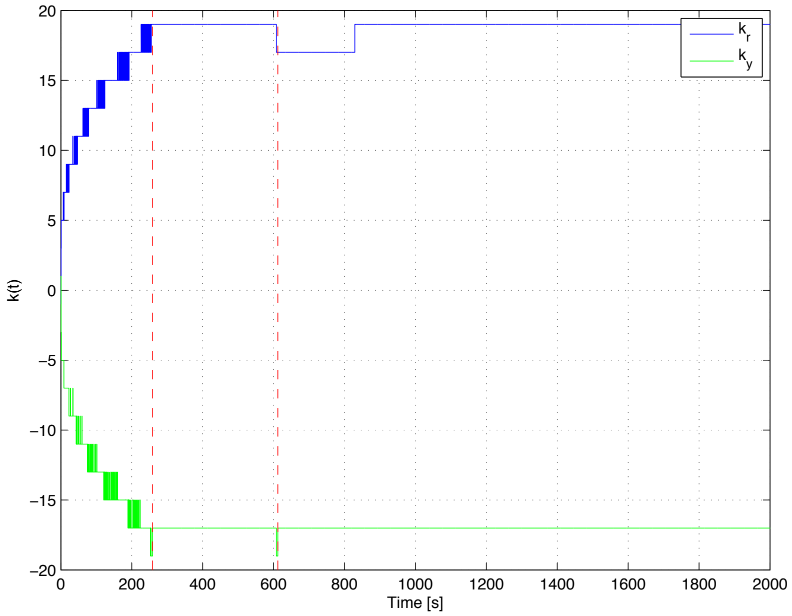


Figure 4.8: Time-evolution of the adaptive gains for the continuous-time algorithm 1, with the stability overlay. The red dashed lines indicate the time instants when the currently scheduled controller fails.

Let us consider a time-varying plant described by

$$\mathcal{P}(\rho(t)) := \begin{cases} \dot{x} = f(x, u, w, \rho), \quad x(0) = x_0, \\ y = g(x, u, w, \rho), \\ z = h(x, u, w, \rho), \end{cases} \tag{4.19}$$

where $x_0 \in \mathbb{R}^n$ is a bounded and fixed (but unknown) initial condition, $w(\cdot) \in \mathcal{L}_\infty$ is a bounded (but unknown) exogenous disturbance, $\rho(\cdot)$ is a vector of (possibly time-varying) parameters, $u(\cdot)$ is the control input and $t \in \mathbb{R}_0^+$ explicitly denotes the time-dependence of the plant dynamics. We also define $x(\cdot)$ as the state of the plant and $y(\cdot)$ and $z(\cdot)$ as the measurement and the performance outputs, respectively. As for the LTI case, the variable $z(\cdot)$ can include performance outputs. Notice that (4.19) can describe a broad class of dynamic systems acted by exogenous disturbances.

We assume that ρ can be time-dependent and that it cannot be measured or estimated with the desired accuracy. For instance, let us consider the case where the process model has one parametric uncertainty, $\rho \in [\rho_{\min}, \rho_{\max}]$. Although several switching MMAC methodologies are available in the literature to solve this kind of problems, they all share the same principles: in terms of design, we divide the (large) set of parametric uncertainty, Ω, into N (small) subregions, Ω_j, $j = \{1, \cdots, N\}$ – see Fig. 4.9 – and synthesize a non-adaptive controller for them; in terms of implementation, we try to identify which region the uncertain parameter, ρ, belongs to, and then use the controller designed for that region.

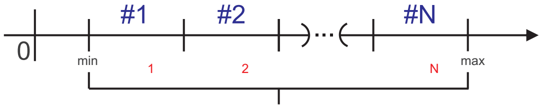


Figure 4.9: Uncertainty region, Ω, for the parameter ρ, split into N subsets.

In order to prove the stability of the closed-loop with the SO, we are going to posit the following assumptions throughout the remainder of this chapter.

Assumption 4.2: There exist continuous strictly increasing functions $\sigma_1 : \mathbb{R}^+ \to \mathbb{R}^+$ and $\sigma_2 : \mathbb{R}^+ \to \mathbb{R}^+$ and constant $\sigma > 0$, such that, for any finitely switching input (4.2), any $\Delta T > 0$, any $t_o \geq 0$, and any (possibly time-varying) $\rho(\cdot)$,

$$\left\| z|^{\sigma}_{[0,t_o+\Delta T]} \right\| \leq \sigma_1(\Delta T) \left\| z|^{\sigma}_{[0,t_o]} \right\| + \sigma_2(\Delta T).$$

□

Remark 4.9: Notice that a system complying with Assumption 4.2 cannot have a finite escape time. This is required since otherwise the stability of the system could be endangered by switching to a destabilizing controller, even for a very short period. ◇

Consider that $\rho(t) \in \Omega \in \mathbb{R}^{n_\rho}$ for all $t \geq 0$. The set Ω is the *uncertainty region* of the vector of parameters, ρ, as illustrated in Fig. 4.9. Let Ω_j denote a subset of Ω. Then, we posit the following assumption, which is concerned to the existence of a stabilizing controller as explained later.

Assumption 4.3: Let $\Omega_j \subset R^{n_\rho}$, $j = 1, 2, ..., N$ satisfy $\Omega \subset \bigcup_j \Omega_j$. There exist strictly positive constants l^*, ΔT^*, ν and γ, with $\gamma < 1$, such that, for any finitely switching input (4.2) with $K_* = K_j$, if for all $t \geq t_o$, i) $\rho(t) \in \Omega_j$ and ii) $|\dot{\rho}(t)| \leq \nu$, then

$$\left\| z|^{\sigma}_{[0,t_o+\Delta T]} \right\| \leq \gamma \left\| z|^{\sigma}_{[0,t_o]} \right\| + l^*,$$

for all $\Delta T \geq \Delta T^*$ and for σ as in Assumption 4.2. □

Remark 4.10: The value of ν is dictated by each controller and is referred to as the *allowable time-rate of variation* of the vector of parameters, $\rho(\cdot)$. ◇

The following definition is important when taking into account the allowable time-rate of variation of the dynamics of a plant.

Definition 4.1. *Suppose that $\rho(t_A) \in \Omega_j$. We say that the* plant dynamics drifted at time instant $t = t_A$ *if there exists $\delta^* > 0$ such that, for every $0 < \delta \leq \delta^*$, we have $\rho(t_A + \delta) \notin \Omega_j$. Furthermore, $t = t_A$ is referred to as a* drifting time instant.

Assumption 4.4: There exists $T_{\min} > 0$ such that, if $\rho(t) \in \Omega_j$, then there exist t_1 and t_2 such that

a) $|t_2 - t_1| \geq T_{\min}$;

b) $t_1 \leq t \leq t_2$;

c) $\rho(\tau) \in \Omega_j$ for all $\tau \in [t_1, t_2]$.

□

This last assumption guarantees that the plant dynamics remain in the same "region" for a time interval of at least $T_{\min}$. In other words, it means that if t_A and t_B are *drifting time instants* with $t_A \neq t_B$, then $|t_A - t_B| \geq T_{\min}$.

Remark 4.11: Assumptions 2–4 are not stringent, since they arise naturally from the control problem at hands. Namely,

- Assumption 4.2 ensures that, even if we select the *wrong* controller during a finite amount of time, the norm of the output of the plant does not increase to infinity in finite time – this is valid, for instance, for every linear time-invariant system [69];

- Assumption 4.3 simply states that, at each time instant t_o, at least one of the *eligible* controllers must be able to stabilize the plant with parameters $\rho(t) \equiv \rho(t_o), \underset{t \geq t_o}{\forall}$;

- Assumption 4.4 will guarantee that the adaptive control algorithm has enough time to adapt itself to the changes in the plant.

◇

The reward after using controller $K_p(\cdot)$ is defined as in (4.12), with $\sigma > 0$ as in Assumption 4.2.

4.5.1 SO Algorithm #2 for Known ΔT^* and l^*

The Stability Overlay Algorithm #2, depicted in Fig. 4.10, for time-varying plants, with known ΔT^* and l^*, is described next. Contrary to the SO Algorithm #1, in this case we need not increase ΔT and l, since they are assumed known a priori. Therefore, if all the controllers have received non-positive rewards, then the dynamics of the plant have drifted. Hence, a previously disqualified controller is the one which is able to stabilize the plant, and therefore we set $S = S_o$. We note that, in the next subsection, we are going to relax the assumption on the knowledge of ΔT^* and l^*.

Theorem 4.2. *Under Assumptions 2–4, for sufficiently large T_{min}, the Stability Overlay Algorithm #2 for Time-Varying Plants with known ΔT^* and l^* results in $\left\| z|^{\sigma}_{[0,t]} \right\|$ bounded.*

The proof for *sufficiently large* $T_{\min}$ is similar to the LTI case. In the sequel, an upper bound for $T_{\min}$ is derived.

In the algorithm depicted in Fig. 4.10, we suppose that ΔT^* and l^* are known (these can actually be upper bounds for ΔT^* and l^*, respectively). This assumption is going to be relaxed in subsection 4.5.2. The difference between the algorithms for time-invariant and for time-varying plants, is that in the later we never increase ΔT and l, since we know ΔT^* and l^*, and these are the values used for every controller.

Suppose that $T_{\min} > 2N\Delta T^*$. Then, during any time-interval ΔT such that $\Delta T \geq T_{\min}$, at least one controller receives a positive reward, even if the controllers are selected in a non-sequential fashion. To see this, consider that all the rewards we get during that time-interval are zero rewards. Then, all the N controllers have

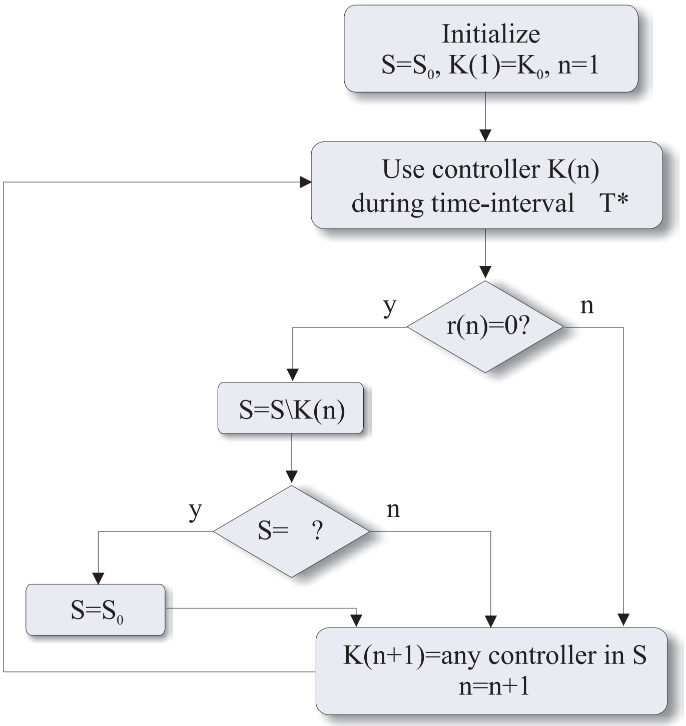


Figure 4.10: Stability Overlay (SO) Algorithm #2, for time-varying plants, with known ΔT^* and l^*. The notation $S = S\backslash K(n)$ means "the exclusion of element $K(n)$ from set S".

failed in a row, and, when connected again to the loop, they all failed once more. Since the plant dynamics can only drift once during the time-interval $2N\Delta T^*$ (see Definition 1), this is a contradiction.

According to Assumption 4.2, whenever a controller receives a zero reward, we have

$$\left\| z|^{\sigma}_{[0,(i+1)\Delta T^*]} \right\| \leq \Gamma \left\| z|^{\sigma}_{[0,i\Delta T^*]} \right\| + L,$$

where $\Gamma = \sigma_1(\Delta T^*)$ and $L = \sigma_2(\Delta T^*)$. On the other hand, and according to Assumption 4.3, whenever a controller receives a positive reward, we have

$$\left\| z|^{\sigma}_{[0,(i+1)\Delta T^*]} \right\| \leq \gamma \left\| z|^{\sigma}_{[0,i\Delta T^*]} \right\| + l^*.$$

Then, if for given integer n^*, we have $T_{\min} > (2N + n^*)\Delta T^*$, we conclude that

$$\left\| z|^{\sigma}_{[0,t_o+T_{\min}]} \right\| \leq a \left\| z|^{\sigma}_{[0,t_o]} \right\| + b,$$

where $a \leq \Gamma^{2N}\gamma^{n^*}$ and b is a function of L, l^*, Γ, γ, N and n^*, since in every $T_{\min}$ interval, a correct controller must be used at least n^* times. Notice that, if $a < 1$, then

$$\underset{z_0 \geq 0}{\exists} : \left\| z|^{\sigma}_{[0,t_o]} \right\| \leq z_0 \Rightarrow \left\| z|^{\sigma}_{[0,t_o+T_{\min}]} \right\| \leq z_0.$$

Thus, if

$$\gamma^{n^*} < \frac{1}{\Gamma^{2N}}$$

the system is input/output stable. Therefore, a sufficient condition for $T_{\min}$ is

$$T_{\min} > (2N + \log_{\gamma}\left(\Gamma^{-2N}\right))\Delta T^*.$$

This means that, the faster the stabilizing controllers are capable of reducing the norm of the output, *i.e.*, the smaller the value of γ is, the smaller $T_{\min}$ can be. However, the larger the value of Γ and the larger the number of controllers, N, the larger the interval $T_{\min}$ must be. Notice that $\Gamma > 1$, otherwise the solution is trivial.

4.5.2 SO Algorithm #3 for Unknown ΔT^* and l^*

Define

$$\overline{z}\,(l, \epsilon) = \frac{l}{1-\gamma} + \epsilon,$$

for γ as in Assumption 2 and $\epsilon > 0$. Notice that, if

$$\left\| z|_{[0,t]}^{\sigma} \right\| \geq \overline{z}^* := \frac{l^*}{1-\gamma},$$

and a controller that receives a positive reward (4.12) is selected, then, for $\Delta T \geq \Delta T^*$,

$$\left\| z|_{[0,t+\Delta T]}^{\sigma} \right\| \leq \left\| z|_{[0,t]}^{\sigma} \right\| .$$

In other words, for all $\epsilon > 0$, if a controller that only receives positive rewards (4.12) is used long enough, then,

$$\left\| z|_{[0,t+\Delta T]}^{\sigma} \right\| \leq \overline{z}^* + \epsilon.$$

Let S_o denote the set of available controllers for the SO. The Stability Overlay Algorithm #3, for time-varying plants with unknown ΔT^* and l^*, is shown in Fig. 4.11.

This algorithm has *two* differences when compared to the time-invariant case:

a) The sets of eligible controllers, S and Q, are "reset" ($S = S_o$ and $Q = S_o$) whenever the discounted norm of the output is below a given threshold, $\overline{z}\left(l(n), \epsilon\right)$;

b) A controller can only be disqualified if it fails twice in between "resets" *and* $\left\| z|_{[0,t]}^{\sigma} \right\| \geq \overline{z}\left(l(n), \epsilon\right)$.

As explained in the sequel, these modifications guarantee the boundedness of ΔT and l, while keeping the input/output stability of the closed-loop. Similarly to what happened with the SO Algorithm #1, the initializations ΔT_o and l_o can be performed arbitrarily, without affecting the stability of the closed-loop.

Remark 4.12: The reasoning behind the use of set Q is as follows. Suppose that a given controller is not able to stabilize the plant and, hence, is disqualified. Then, suppose that the plant dynamics change and that the formerly disqualified controller is now the *only* eligible controller able to stabilize the plant. Using the architecture without set Q, we could only *re-use* previously disqualified controllers whenever we increased $\Delta T(n)$ and $l(n)$. Therefore, the set Q accounts for time-variations of the

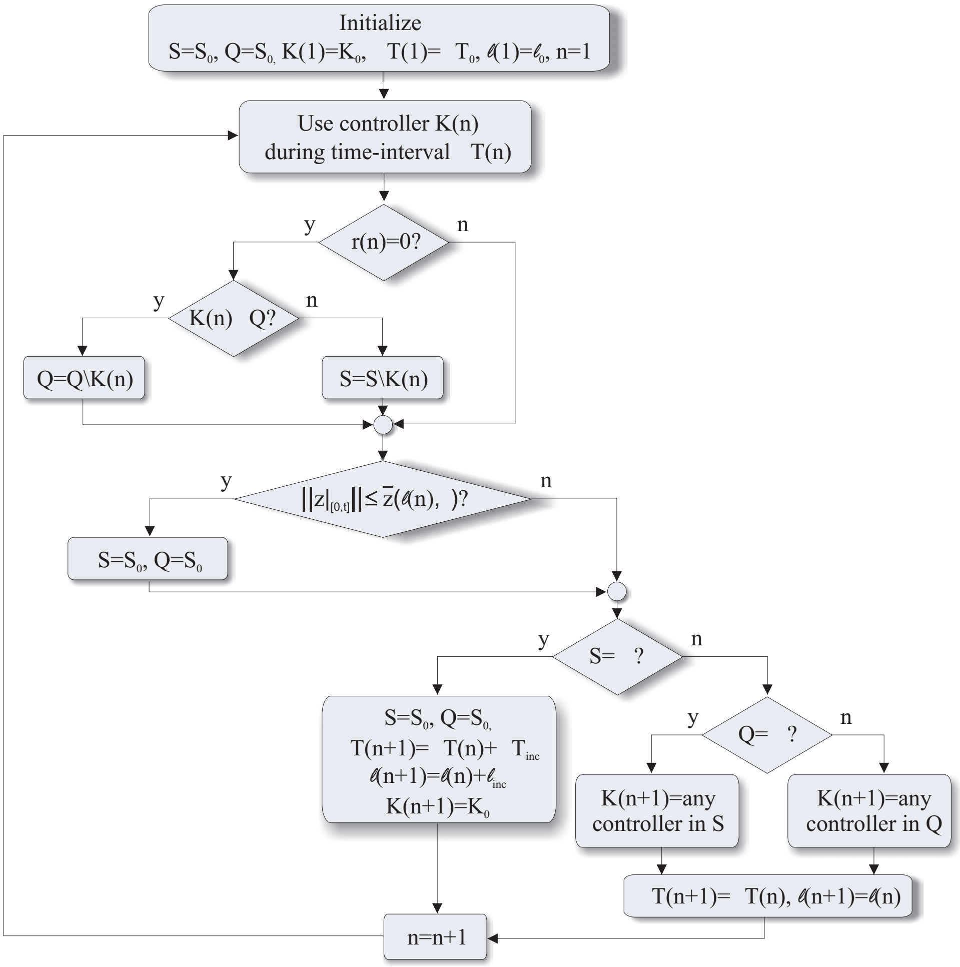

Initialize
S=S0, Q=S0, K(1)=K0, T(1)= T0, ℓ(1)=ℓ0, n=1
Use controller K(n) during time-interval T(n)
r(n)=0?
y
n
K(n) Q?
y
n
Q=Q\K(n)
S=S\K(n)
||z|[0,t]|| ≤ z̄(ℓ(n),)?
y
n
S=S0, Q=S0
S= ?
y
n
S=S0, Q=S0, T(n+1)= T(n)+ Tinc ℓ(n+1)=ℓ(n)+ℓinc K(n+1)=K0
Q= ?
y
n
K(n+1)=any controller in S
K(n+1)=any controller in Q
T(n+1)= T(n), ℓ(n+1)=ℓ(n)
n=n+1

Figure 4.11: Stability Overlay (SO) Algorithm #3, for time-varying plants with unknown ΔT^* and l^*.

plant dynamics, since $\Delta T(n)$ and $l(n)$ are only increased whenever both sets, S and Q, are empty. $\diamond$

Remark 4.13: The parameters $l(n)$ and $\Delta T(n)$ can also be re-initialized, whenever the condition $\left\| z|^{\sigma}_{[0,t]} \right\| \leq \bar{z}(l(n), \epsilon)$ is satisfied. On one hand, this prevents the loss in terms of performance that can arise from the use of large values of $\Delta T(n)$ and $l(n)$, which in turn can be caused by occasional adverse disturbances. On the other, longer periods may be required to select a stabilizing controller whenever the plant parameters drift, since the values of $\Delta T(n)$ and $l(n)$ might have to be increased every time the condition $\left\| z|^{\sigma}_{[0,t]} \right\| \leq \bar{z}(l(n), \epsilon)$ is satisfied. $\diamond$

The following theorem summarizes the main result of this section.

Theorem 4.3. *Under Assumptions 2–4, for sufficiently large T_{min}, the Stability Overlay Algorithm #3 for Time-Varying Plants results in $\left\| z|^{\sigma}_{[0,t]} \right\|$ bounded.*

Proof. For the sake of simplicity, we divide the proof of Theorem 4.3 into several small steps. We start by showing that bounded $\Delta T(n)$ and $l(n)$ imply bounded $\left\| z|^{\sigma}_{[0,t]} \right\|$, and finally we show that $\Delta T(n)$ and $l(n)$ are bounded.

Claim 4.2. *If $\lim\limits_{n\to\infty} \Delta T(n) < \infty$ and $\lim\limits_{n\to\infty} l(n) < \infty$, then $\lim\limits_{t\to\infty} \sup \left\| z|^{\sigma}_{[0,t]} \right\| < \infty$.*

Proof. In words, this means that if $\left\| z|^{\sigma}_{[0,t]} \right\|$ is unbounded, then $\Delta T(n)$ (and, consequently, $l(n)$) is also unbounded. For the sake of simplicity of the proof, we consider the following cases separately: i) $\underset{t_o}{\exists} : \underset{t\geq t_o}{\forall}, \left\| z|^{\sigma}_{[0,t]} \right\| > \overline{z}\left(l(n), \epsilon\right)$; ii) $\underset{t_1}{\forall} : \underset{t\geq t_1}{\exists}, \left\| z|^{\sigma}_{[0,t]} \right\| \leq \overline{z}\left(l(n), \epsilon\right)$.

We prove case i) by contradiction. Indeed, consider that $\left\| z|^{\sigma}_{[0,t]} \right\|$ satisfies i) and is unbounded, but that $\Delta T(n)$ (and, consequently, $l(n)$) is bounded. In that case, we have that: a) none of the controllers is persistently receiving positive rewards (4.12), since the norm of the output is not bounded, and b) sets S and Q are never being reset, since $\left\| z|^{\sigma}_{[0,t]} \right\| > \overline{z}\left(l(n), \epsilon\right)$ for all $t \geq t_o$. Hence, at some point, $S \cup Q = \emptyset$. Thus, $\Delta T(n)$ is increased according to $\Delta T(n+1) = \Delta T(n) + \Delta T_{inc}$. If $\Delta T(n)$ is increased infinitely many times using this law, then $\lim\limits_{n\to\infty} \Delta T(n) \to \infty$, which is a contradiction. The same conclusion applies to $l(n)$.

In case ii), we assume that the norm of the output decreases below $\overline{z}\left(l(n), \epsilon\right)$ from time to time. In the sequel, we are going to show that, if $\Delta T(n)$ (and, consequently, $l(n)$) is bounded, then $\left\| z|^{\sigma}_{[0,t]} \right\|$ is also bounded.

If $\Delta T(n)$ is bounded, then, for some $t_2 \geq 0$, the update law $\Delta T(n{+}1) = \Delta T(n) + \Delta T_{\text{inc}}$ is not used for any $t \geq t_2$. Let $\tilde{t} \geq t_2$ be defined such that $\left\| z|^{\sigma}_{[0,\tilde{t}]} \right\| > \overline{z}\left(l(n), \epsilon\right)$. Further let $\underline{t}$ and $\overline{t}$ be successive times such that $\underline{t} < \overline{t}$, $\left\| z|^{\sigma}_{[0,\underline{t}]} \right\| \leq \overline{z}(l(n), \epsilon)$, $\left\| z|^{\sigma}_{[0,\overline{t}]} \right\| \leq \overline{z}(l(n), \epsilon)$, and

$$\underset{t \in]\underline{t},\, \overline{t}[}{\forall} : \left\| z|^{\sigma}_{[0,t]} \right\| > \overline{z}\left(l(n), \epsilon\right).$$

Again, according to Assumption 4.2, for $t \geq t_2$, whenever a controller receives a zero reward, we have

$$\left\| z|^{\sigma}_{[0,t+\Delta T]} \right\| \leq \Gamma \left\| z|^{\sigma}_{[0,t]} \right\| + L,$$

where $\Gamma = \sigma_1(\Delta T)$ and $L = \sigma_2(\Delta T)$, and according to Assumption 4.3, whenever a controller receives a positive reward, we have

$$\left\| z|^{\sigma}_{[0,t+\Delta T]} \right\| \leq \gamma \left\| z|^{\sigma}_{[0,t]} \right\| + l^*.$$

Since we are assuming $t \geq t_2$, we conclude that we cannot receive non-positive rewards (4.12) more than $2N-1$ times in the interval $[\underline{t},\, \overline{t}]$ (otherwise $\Delta T(n)$ would be increased, which would be a contradiction). Therefore, the norm of the output at time $\underline{t} < t < \overline{t}$ is bounded by

$$\left\| z|^{\sigma}_{[0,t]} \right\| \leq a \left\| z|^{\sigma}_{[0,\underline{t}]} \right\| + b = a\overline{z}\left(l(n), \epsilon\right) + b,$$

where $a = \Gamma^{2N-1}\gamma^{\tilde{n}}$, for some $\tilde{n} \geq 1$, and b is a positive constant (see the proof of Theorem 4.2). Since $a \leq \Gamma^{2N-1}$, we conclude that $\left\| z|^{\sigma}_{[0,t]} \right\|$ can be bounded by a constant which is independent of the choice of $\underline{t}$ and $\overline{t}$. Therefore, $\left\| z|^{\sigma}_{[0,t]} \right\|$ is uniformly bounded, which concludes the proof. □

Claim 4.3. *Suppose that t_A is a drifting time instant (see Definition 4.1) and that $\left\| z|^{\sigma}_{[0,t_A]} \right\| \leq \theta$. Then, for sufficiently large $T_{\min}$, ΔT and l,*

$$\left\| z|^{\sigma}_{[0,t_A+t_o]} \right\| \leq \overline{z}(l, \epsilon),$$

for some $t_o \in [0, T_{\min}]$.

Proof. (by contradiction) Suppose Claim 4.3 is not true. Then,

$$\underset{t_o \in [0,\, T_{\min}]}{\forall} : \left\| z|^{\sigma}_{[0,\, t_A + t_o]} \right\| > \overline{z}(l, \epsilon),$$

which means that the "reset" $S = S_o$ and $Q = S_o$ will never occur. Hence, the algorithm behaves as the SO for time-invariant plants – see Section 4.3. However, the SO for time-invariant plants guarantees that, for sufficiently large t_o,

$$\left\| z|^{\sigma}_{[0,\, t_A + t_o]} \right\| \leq \overline{z}(l, \epsilon),$$

which is a contradiction. □

Claim 4.4. *Suppose that $\rho(t_A) \in \Omega_j$, where t_A is a drifting time instant, and that, at $t = t_A$, the selected control law is $K_j(\cdot)$. Then, for sufficiently large $T_{\min}$, ΔT and l, and for sufficiently small $\delta > 0$, we have $S = S_o$ at time instant $t = t_A + T_{\min} - \delta$.*

Proof. Claim 4.4 states that before every drifting time instant and assuming $T_{\min}$, ΔT and l large enough, we have $S = S_o$, where S_o is the set of all available controllers for the adaptive law. Using Claim 4.3, for large $T_{\min}$,

$$\underset{t_o \in [0, T_{\min}]}{\exists} : \left\| z|^{\sigma}_{[0, t_A + t_o]} \right\| \leq \overline{z}(l, \epsilon),$$

we conclude that, for $t = t_A + t_o$, we have $S = S_o$ and $Q = S_o$. Since, for $\Delta T \geq \Delta T^*$, there is at least one controller that receives a positive reward (4.12), for every bounded (but unknown) exogenous disturbance, $w(\cdot) \in \mathcal{L}_\infty$, the set Q is never empty for $t \in [t_\mathrm{A} + t_o,\, t_\mathrm{A} + T_{\min}]$. Thus, $S = S_o$ at time instant $t = t_A + T_{\min} - \delta$, for sufficiently small $\delta > 0$. □

Claim 4.5. *The parameters $\Delta T(n)$ and $l(n)$ are uniformly bounded.*

Proof. Using Claim 4.4, we conclude that, for sufficiently large $T_{\min}$, ΔT and l, $Q = S_o$ at the drifting time instants and hence all the controllers are allowed to be connected to the loop. Since we are assuming $T_{\min}$, ΔT and l are large enough, at least one of the controllers is going to receive a positive reward (4.12), for any $w(\cdot) \in \mathcal{L}_\infty$. Given that ΔT and l can only be increased if all the controllers fail, these parameters are uniformly bounded. □

Using Claim 4.2 and Claim 4.5, we conclude that the output of the time-varying closed-loop system is bounded.

□

4.5.3 Computation of an Upper Bound for $T_{\mathbf{min}}$

This subsection suggests a method to compute an upper bound $T_{\min}$, when the SO Algorithm #3 is used. Let t_{A}, t_{B} and t_{C} be drifting time instants, satisfying Assumption 3, *i.e.*, $\min\{|t_{\mathrm{A}}-t_{\mathrm{B}}|,|t_{\mathrm{A}}-t_{\mathrm{C}}|,|t_{\mathrm{B}}-t_{\mathrm{C}}|\} \geq T_{\min}$.

Also, consider $\Delta T \geq \Delta T^{*}$ and $l \geq l^{*}$. According to Claim 4.3,

$$\left\| z|^{\sigma}_{[0,t_{\mathrm{A}_0}]} \right\| \leq \overline{z}(l,\epsilon),$$

for some $t_{\mathrm{A}_0} \in [t_{\mathrm{A}}, t_{\mathrm{B}}]$. Hence, similarly to what was derived in subsection 4.5.1, we have

$$\left\| z|^{\sigma}_{[0,t_{\mathrm{B}}]} \right\| \leq \gamma^{m^{*}}\Gamma^{2N}\overline{z}(l,\epsilon)+\psi =: \overline{z}_1,$$

where N is the number of controllers, ψ is a continuous function of l^{*}, L, Γ, γ and m^{*}, and where m^{*} is the number of time-intervals ΔT needed for a controller receiving positive rewards (4.12) to reduce the discounted norm of the output from $\overline{z}_1$ to $\overline{z}\,(l,\epsilon)$. $T_{\min}$ is some time interval *large enough* so that the stability of the SO Algorithm #3 is guaranteed. Thus, $T_{\min} \geq t_{\mathrm{B}_0} - t_{\mathrm{B}}$, where $t_{\mathrm{B}_0} \in [t_{\mathrm{B}}, t_{\mathrm{C}}]$ is such that

$$\left\| z|^{\sigma}_{[0,t_{\mathrm{B}_0}]} \right\| \leq \overline{z}(l,\epsilon).$$

Therefore,

$$T_{\min} \geq \left(2N + \log_{\gamma}\left(\frac{1}{\gamma^{2N}}\right)\right)\Delta T.$$

Notice that this result is similar to that of the SO algorithm #2, except that ΔT is in general larger than ΔT^{*}, which means that $T_{\min}$ has also to be larger, as expected.

An interpretation of this result is as follows: the larger the number of controllers we have to test, N, the larger the amount of time the plant dynamics must remain in the same region; moreover, if m^{*} is small, we can switch to another region sooner than if the time to recover from the use of a wrong controller is large; finally, the faster we can exclude a controller and the faster the *good* controllers decrease the output norm, the smaller the time the parameters must stay in the same region.

This result shows the compromise between performance for time-varying and for time-invariant systems. For instance, it is well known that the performance of the multiple-model adaptive control architectures, for time-invariant plants, increases with the number of eligible non-adaptive controllers. However, this new result shows that, the more controllers we have, the larger the transients are going to be.

4.6 Integration of the SO with the RMMAC/BMI

We recall that the time-rate of variation of the uncertain dynamics of the plant to be controlled is constrained by

a) $T_{\min}$ (in order to satisfy the assumptions of the SO);

b) the maximum rate of variation allowed by each of the (local) non-adaptive controllers.

In what follows, we assume that the plant to be controlled can be described or approximated by a (possibly time-varying) linear model.

In reference to Fig. 4.2, if, for instance, mixed-μ controllers are used, the plant must be time-invariant. Otherwise, the stability of the closed-loop system cannot be guaranteed. Another possible approach is to use the LPV/BMI controllers presented in Chapter 2.

The SO does not describe how one should choose among the set of eligible controllers, when there is more than a single element. The appropriate choice of such controllers is going to be responsible for the performance of the closed-loop system. As an example, we are going to use the *identification part* of the Robust Multiple-Model Adaptive Control (RMMAC) methodology. Figure 4.12 depicts the switching RMMAC architecture, for the case where N regions are used. At this point, it is not our intention to give an in-depth explanation of the RMMAC – the interested reader is referred to [57–59] – but rather to use the SO to endow the RMMAC with input/output stability capabilities.

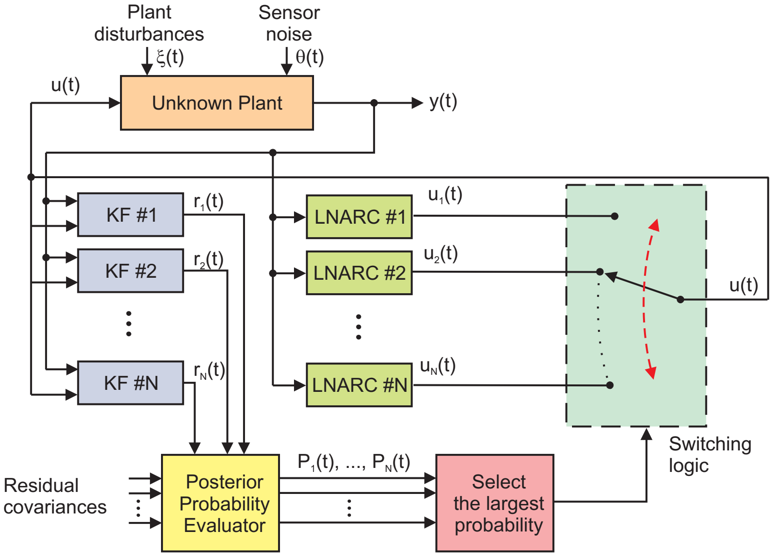


Figure 4.12: The RMMAC architecture with N models

4.6.1 The Double Mass-Spring-Dashpot Plant

We are going to use the (double) Mass-Spring-Dashpot (MSD) testbed example of Fig. 4.13, interconnected with the RMMAC/BMI scheme of Chapter 3, to illustrate the use of the SO for a time-varying plant. The detailed description of the MSD plant can be found in Section 3.2. For the sake of comprehension, we recall that the unknown parameters are the time-varying spring stiffness

$$k_1 \in K := [0.25 \quad 1.75] \text{ N/m}$$

and the constant input time-delay

$$0 < \tau < 0.05 \text{ s.}$$

The disturbance force $d(t)$ shown in Fig. 3.3 is a stationary first-order (colored) stochastic process generated by driving a low-pass filter, with transfer function $W_d(s)$, with continuous-time white noise $\xi(t)$, with zero mean and intensity $\beta(t)\Xi$, according to

$$d(s) = \frac{\alpha}{s+\alpha}\xi(s) = W_d(s)\xi(s).$$

The sensor noise considered is continuous-time white noise $\theta(t)$, with zero mean and intensity $\beta(t)\Theta$. $\beta(\cdot)$ is a time-varying unknown positive scalar, that is going to be

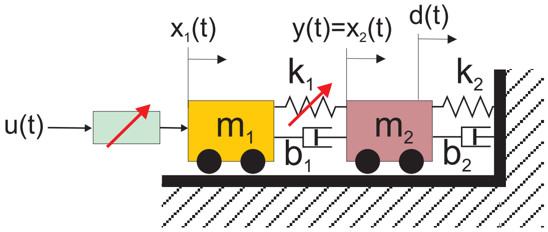


Figure 4.13: MSD system with uncertain spring constant, k_1, and disturbances denoted by $d(t)$. $u(t)$ is the control input and $z(t)$ is the system output.

described next. The KFs of the RMMAC/BMI were designed using $\beta(\cdot) = \beta = 1$. This design assumption is going to be violated in the following simulation, in order to evaluate the behavior of the SO.

Following the RMMAC synthesis methodology and using the same design choices as the ones described in [58, 59], we obtain $N = 4$ local non-adaptive robust controllers (LNARCs) – which are mixed-μ controllers in the original RMMAC design – in order to achieve at least 70% of the performance we would have obtained had we known the value of the uncertain parameter, k_1.

The mixed-μ controllers are thereafter replaced by LPV/BMI controllers with similar specifications, but assuming nonzero bounds on the rate of variation of the parameter, k_1. In this design, we assume a bound of 0.001 (N/m)/s for the slope (time-variation) of the parameter k_1. The state of the plant is augmented by the D-scales obtained from the μ-analysis, increasing the state up to 16 variables. For the LPV realization, we have adopted a polytopic description for the system, since, for each region, the plant dynamics can be naturally described as the convex combination of the dynamics on the boundaries of that region.

The RMMAC/BMI architecture is finally modified by the SO for time-varying plants, that serves as a supervisor to guarantee input/output stability of the closed-loop. The SO constrains the set of eligible controllers that the RMMAC/BMI can select, resorting to the control residuals, and should only act if the RMMAC/BMI is persistently choosing controllers that are not capable of stabilizing the plant.

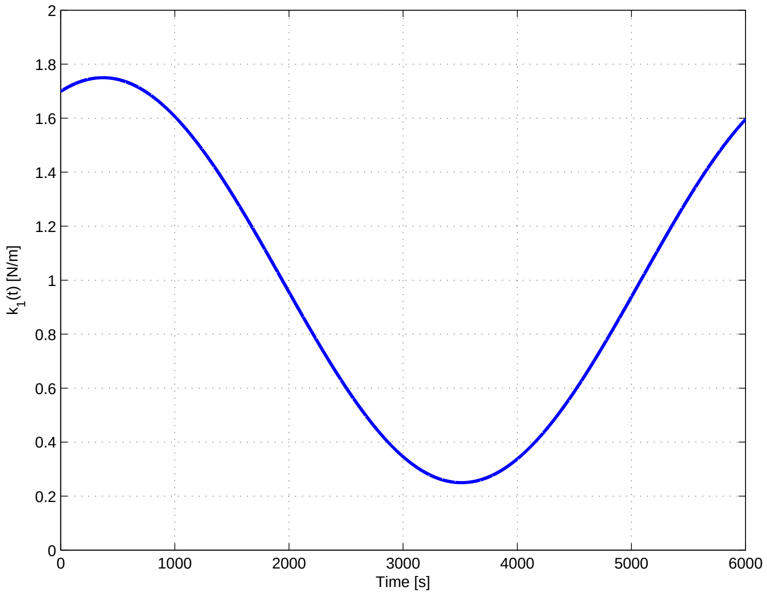


Figure 4.14: Uncertain spring stiffness, $k_1(t)$, time-evolution.

4.6.2 Simulation Results

Figure 4.14 depicts the time-evolution of the spring constant considered for simulation purposes. It is important to notice that $|\dot{k}_1(t)|$ is within the predefined bounds. Hence, every time instant, there is at least one controller that is able to stabilize the plant.

The disturbance and noise intensities used during the design of the KFs were multiplied by the variable $\beta(t)$, depicted in Fig. 4.15.

The simulation results are illustrated in Figs. 4.16 and 4.17, where a time-delay of $\tau = 10$ ms was considered in the control input. The standard switching RMMAC is used for comparison purposes.

During the first 3000 secs, the switching RMMAC/BMI with and without the SO behave approximately the same way. This happens because $\beta(t) = 1$ for $t \in [0,\ 3000]$ secs, which means that the design assumptions of the KFs are not violated (except for the time-variations of the plant), and hence the identification subsystem of the RMMAC/BMI converges to the *correct* parameter region. Therefore, the SO does not interfere in the selection of the controllers, since none of them has failed. The same comments are valid for $t \in [4200,\ 6000]$ secs.

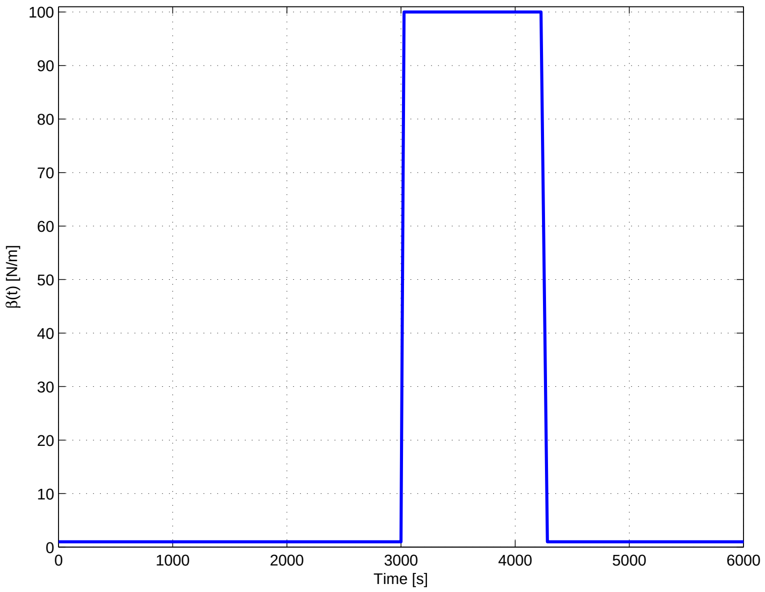


Figure 4.15: Disturbance and noise intensities factor, $\beta(t)$.

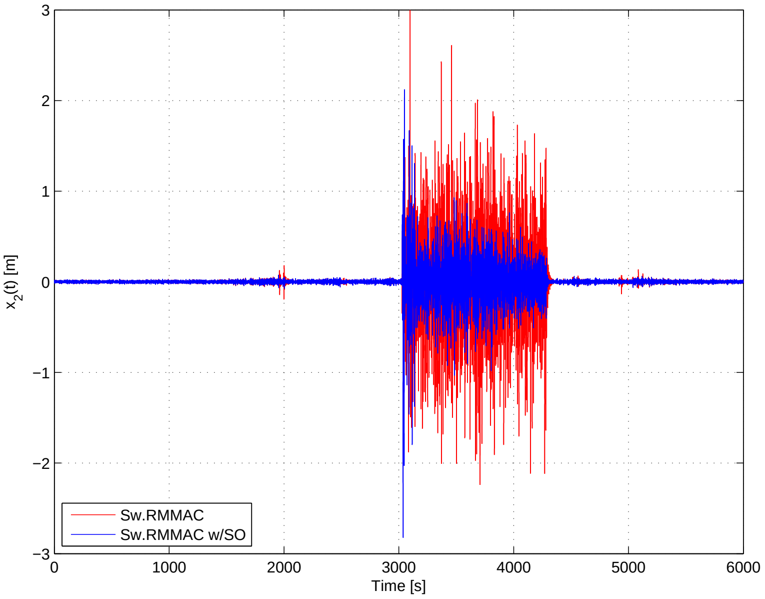


Figure 4.16: Mass 2 position, $x_2(t)$, time-evolution for the closed-loop, using the switching RMMAC/BMI integrated with the Stability Overlay for time-varying plants.

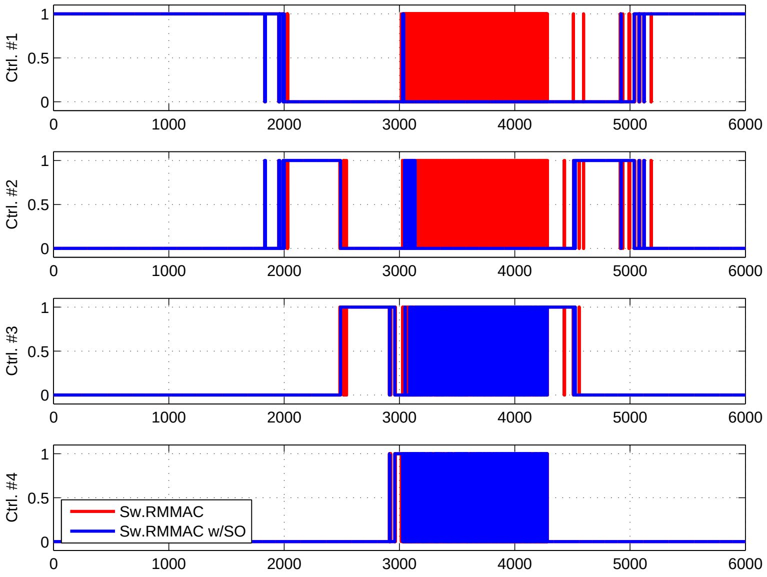


Figure 4.17: Controller selection for the switching RMMAC/BMI with and without the Stability Overlay.

For $t \in [3000, 4200]$ secs, the KFs do not converge. Therefore, the standard switching RMMAC/BMI selects almost arbitrarily the controllers at each sampling time, as illustrated in Fig. 4.17. This leads to a severe deterioration in terms of performance of the closed-loop. The SO, however, is able to *disqualify* controllers #1 and #2 after a few seconds, since they are not being able to stabilize the plant. Hence, we get the performance improvements depicted in Fig. 4.16. In terms of RMS, during that slice of time, we obtained

$$\text{RMS for the Sw. RMMAC} = 0.0178 \text{ m},$$
$$\text{RMS for the Sw. RMMAC integrated with the SO} = 0.0128 \text{ m}.$$

Using Theorem 4.3, we conclude that, if the time interval between any two drifting time-instants is always greater than $T_{\min}$ and if $\left|\dot{k}_1\right| < 0.001$ (N/m)/s, the closed-loop of the overall system with the RMMAC/BMI integrated with the SO for TV plants is stable, while no guarantees are available for the standard switching RMMAC/BMI.

Remark 4.14: Assumptions 2–4 can easily be verified resorting to arguments similar to the ones used for LTI systems. ◇

It should be noticed that, for $t \in [3000, 4200]$ secs, the SO is able to disqualify the two controllers that are not able to stabilize the plant. Therefore, besides guaranteeing stability, it also improves the performance, since the destabilizing controllers are not connected to the loop. Nevertheless, for a controller to be excluded from the set of eligible controllers, it has to be tested for a given period of time. In the present case, the intervention of the SO was only needed during a small time-interval, where the spring constant was small. We stress that the MSD plant with a soft spring can be considered as a slow system, since the magnitude of the output does not increase very fast when a destabilizing controller is connected.

If, however, the spring constant is large, we may not get *practical stability*, as stressed in Remark 4.5. For that case, using a destabilizing controller even for a short period can increase very significantly the magnitude of the output. Notwithstanding, a controller cannot be disqualified unless it has been used for a given time-interval. This means that, for stiff springs, the MSD plant with the SO can have very large transients and hence there can happen a severe deterioration in terms of performance.

An example of this issue is illustrated in Fig. 4.18, where a constant spring stiffness is used, with $k_1 = 0.8$ N/m. The disturbances are multiplied by a factor of 100, for $t \geq 500$ secs, instead of $t \in [3000, 4200]$ secs – see Fig. 4.15 –, leading to the aforementioned problems in the identification subsystem of the RMMAC/BMI. Indeed, the transients observed for $t \in [500, 600]$ secs are considerably large, when compared to the steady state output error. Although the SO eventually selects the *correct* controller – which is not the case, as seen in Fig. 4.18, for this example, when the standard switching RMMAC/BMI is used –, the transients observed may not be acceptable in practical applications.

This shortcoming arises from the fact that the SO only provides stability guarantees. In other words, the SO can only guarantee that a stabilizing controller is eventually selected. We emphasize that this problem is common to every stability-based algorithms.

Nevertheless, heuristics such as the one proposed in [124], that do not provide stability guarantees by themselves, can be used in parallel with the RMMAC to

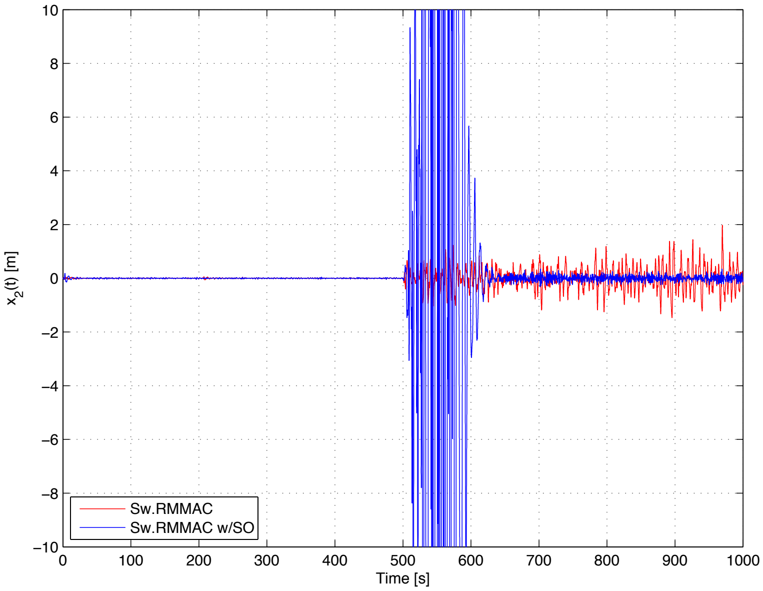


Figure 4.18: Mass 2 position, $x_2(t)$, time-evolution for the closed-loop, using the switching RMMAC/BMI integrated with the Stability Overlay for time-varying plants, with a constant spring stiffness, $k_1 = 0.8$ N/m.

potentially overcome this problem. In particular, the approach in [124] uses the norm of the state estimates of the KFs to discard controllers that are not being able to stabilize the plant.

In summary, these simulations illustrated the usefulness of the SO for time-varying plants, whenever the assumptions of the high-performance adaptive algorithm (the RMMAC/BMI, in the present case) are not satisfied in practice, while highlighting the practical problems that can arise in stability-based algorithms.

4.7 Conclusions

This chapter presented a strategy, referred to as Stability Overlay (SO), that provides input/output stability guarantees for a wide set of adaptive control schemes, when applied to both linear and nonlinear (possibly) time-varying plants. We take advantage of the on-line evaluation of the selected control law to disqualify those that do not comply with the stability requirements. Unlike other adaptive control strategies, we take into account both stability objectives – often robust to a very wide class of disturbances and model uncertainty – and performance requirements – that, in general, assume a stronger knowledge about the plant to be controlled.

For LTI plants, this approach only requires that at least one of the *eligible* controllers is able to stabilize the plant. For nonlinear plants, however, a bound on the rate-of-growth of the output of the plant is also assumed, whenever a destabilizing control law is connected to the loop.

The SO is responsible for disqualifying controllers that are not able to stabilize the plant, while some other adaptive control law, that (possibly) takes into account the plant model and the disturbances properties, is responsible for selecting the *best* controller, *i.e.*, the controller in set S_o that leads to the highest performance.

The theory was illustrated with two examples. We used the Rohrs *et al.* counterexample to show how to endow an adaptive control law for LTI plants with robust stability properties. These stability guarantees were obtained by first discretizing and bounding the adaptive gains, and then by using the same principles as in the previous example. Simulation results were presented, illustrating the benefits of using the SO with the Continuous-time Algorithm 1 of [39]. As an example to illustrate the applicability of the algorithm for TV plants, the SO was used to endow the "Robust Multiple-Model Adaptive Control with LPV/BMI Controllers" (RMMAC/BMI) architecture of Chapter 3 with robust stability properties when the plant to be controlled is uncertain and time-varying.

The proposed method can be applied to a much wider class of adaptive controllers, with little effort, guaranteeing stability properties otherwise not available.

We claim that adaptive control strategies must somehow resort to *control residuals* to guarantee stability of the closed-loop, while they should also rely on the *identification residuals* (typically faster than the control residuals) to enhance performance. Therefore, strategies such as the RMMAC/BMI with the SO for time-varying plants ensure stability of the closed-loop for a wide class of disturbances and model uncertainty, while providing high performance capabilities when the process model matches closely the actual plant.

As a caveat of the SO, the choice of the parameters for the algorithm may be very sensitive if the norm of the output of the closed-loop system grows *very fast*, whenever a destabilizing controller is picked, and if the time required to disqualify a controller is very large. In those situations, although a stabilizing controller is eventually selected, the transients may not be reasonable from a practical point of view. The solutions available in the literature for this type of problems are in general based upon the falsification of control laws without inserting them in the feedback loop (*c.f.*, [50, 51, 54, 108, 109, 111, 112]). However, these approaches require the estimation of a fictitious reference signal, which may be a difficult task in many applications.

Chapter 5

The Decision Subsystem

5.1 Introduction

In the previous chapter, a methodology, referred to as Stability Overlay (SO), that can provide input/output stability guarantees for several types of adaptive control laws was introduced. The SO, however, does not endow these adaptive control laws with any guarantees in terms of performance. This is due to the fact that only a small number of assumptions is made regarding the plant to be controlled. Since performance requirements are, in general, key in practical applications, a more sophisticated decision subsystem must be developed, by taking into account a deeper knowledge regarding the plant. A general decision subsystem is depicted in Fig. 5.1.

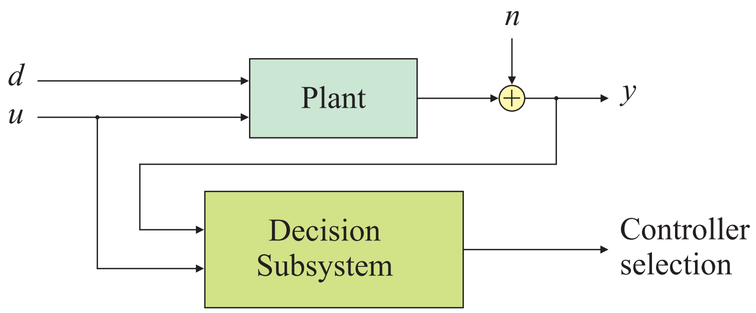


Figure 5.1: The decision subsystem.

There are several solutions available in the literature to the problem of designing adaptive controllers with performance optimization objectives. As previously mentioned, the Multiple-Model Adaptive Control (MMAC) architecture uses a *divide-and-conquer* strategy to stabilize/control an uncertain plant. Instead of designing a compensator for a largely uncertain plant, the family of *admissible* plants to be controlled is split into several subsets. For each uncertainty subset, a non-adaptive controller is synthesized. Then, the on-line model identification/estimation subsystem blends or switches the control signals that are applied to the plant, depending on the significance of each model [27, 32–38].

We address, in particular, the case where the process model has one or more parametric uncertainties, $\rho \in \Omega$. As previously mentioned, although several switching MMAC methodologies are available to solve this problem, they all share the same principles: in terms of design, we divide the (large) set of parametric uncertainty, Ω, into N (small) subregions, Ω_i, $i = \{1, \cdots, N\}$ – see Fig. 5.2, for a single uncertain parameter case – and synthesize a non-adaptive compensator for them; in terms of implementation, we try to identify which region the uncertain parameters vector, ρ, belongs to, and then select the controller designed for that region.

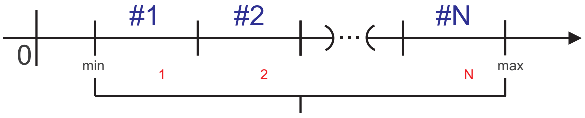


Figure 5.2: Uncertainty region, Ω, for one parameter, ρ, split into N subsets.

Figure 5.3 depicts a multiple-model decision subsystem. The estimates of each model are used by the *Logic* block, to decide which controller should be connected into the loop. The way these models and this *Logic* block are designed is what differentiates each approach. In some circumstances, a convex combination of the outputs of more than one controller can lead to smaller transients. In such cases, the *controller selection* signal in Fig. 5.3 is used to weight the outputs of each controller.

The classical MMAC [25, 29, 30] approaches use Multiple-Model Adaptive Estimation (MMAE) schemes [33, 35] as decision subsystems.

The Robust Multiple-Model Adaptive Control (RMMAC, [57–59]), which was also mentioned in the previous chapters, also resorts to MMAE techniques. In this

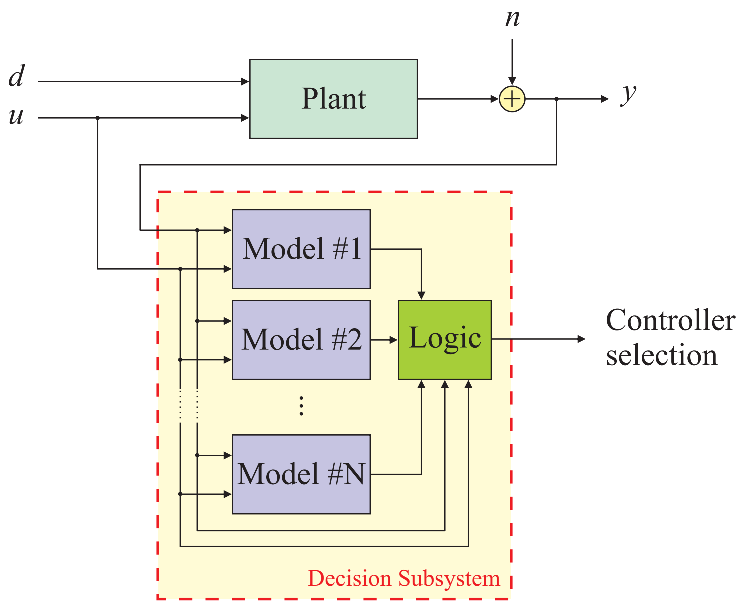


Figure 5.3: Multiple-model decision subsystem.

case, the decision subsystem is composed of a set of Kalman filters (KFs), synthesized for given nominal models, and by a posterior probability evaluator (PPE), which uses the residuals of the KFs to compute an estimation of the posterior probabilities of a given uncertain parameter being in a specific interval. Therefore, the nominal model of the plant and the intensity of the disturbances, as well as their spectral properties, are required to properly design the KFs and the PPE. The behavior of this decision subsystem under several violations of the aforementioned assumptions was presented in [60, 61]. A large number of Monte-Carlo simulations was performed, illustrating the advantages and shortcomings of this method. On the one hand, when the intensity of the disturbances (and measurement noise) are close to the one used to design the KFs, the decision subsystem is able to correctly identify the model of the plant, requiring only a few seconds in the cases presented in [60, 61]. On the other hand, if the intensity of the disturbances (and measurement noise) is much larger than the one used to design the KFs, an incorrect model of the plant may be selected, and a large transient may be observed.

The so-called supervisory control strategies in [31, 36, 37, 55] rely on the output (control or estimation) error to select, at each time instant, the controller that yields the smallest output error, while avoiding fast switchings in the control signals. Thus, the decision subsystem uses this error signal to evaluate the performance of the currently selected controller and to decide which of the eligible models of the plant is, *most likely*, the correct one. As mentioned in Chapter 4, relying on control residuals to select the controllers may lead to very large transients and, hence, to the deterioration of the performance of the closed-loop system.

The decision subsystem of the so-called Multiple Model Adaptive Control with Mixing (MMACwM – see [41, 100]) is composed of a parameter estimator, which is used to select the region to which the uncertain parameters belong. In reference to Fig. 5.2, the decision subsystem estimates the value of ρ, and determines the integer $i \in \{1, \cdots, N\}$ such that $\rho \in \Omega_i$.

A different approach to model selection is denoted *Model Falsification* or *Model Invalidation* – see [71] and references therein. Instead of identifying the model of the plant with the highest posterior probability, the idea in model falsification is to eliminate models that are not compatible with the input/output observations. Thus, by discarding those models, this technique aims to eventually disqualify all but one model. This model, in turn, is the *correct* model of the plant, since it is the only one which is compatible with all the input/output data sets. We are going to explain in further detail this approach in the following sections.

The *Unfalsified Control* theory (*c.f.*, [50, 51, 54, 108, 109, 111, 112]) indeed uses model falsification to select the appropriate controller for a given uncertain plant. Moreover, it allows for the falsification of control laws without inserting them into the feedback loop (*c.f.*, [50, 51, 54, 108, 109, 111, 112]). However, these approaches require the estimation of a fictitious reference signal, which may be a difficult task.

5.1.1 Main Contributions and Organization

The main goal of this chapter is to properly define the concept of *model falsification*, and to provide necessary and sufficient conditions for the falsification of all but one model, in a given set of *eligible* ones. The results are also going to be illustrated through a few examples.

The main contributions of this chapter are as follows:

a) The development of a new distinguishability concept, referred to as *absolute input distinguishability*;

b) The derivation of necessary and sufficient conditions for absolute input distinguishability of Linear Time-Invariant (LTI) and Linear Parameter-Varying (LPV) systems;

c) The derivation of a persistence type of condition on the disturbances that suffices for absolute input distinguishability;

d) The application of the aforementioned technique to a set of examples.

The remainder of this chapter is organized as follows. We start by describing the concept of *model falsification* in Section 5.2. The concept of *absolute input distinguishability* is introduced in Section 5.3. Several necessary and sufficient conditions for absolute input distinguishability of linear systems are also derived, and the applicability of the methods is illustrated in simulation. Finally, some conclusions regarding model falsification and absolute input distinguishability are provided in Section 5.5.

5.2 Model Falsification

The problem of *model falsification* appears in several areas where we are interested in distinguishing among an eligible set of dynamic systems. The simplest model falsification problem one can think of is that of stating whether or not a given dynamic model is *compatible* with the current observed input/output data. However, it is important to notice that a model can never be validated in practice. Indeed, if the model is compatible with the input/output data up to time t, it need not be compatible at time $t + \delta$, where $\delta > 0$. Therefore, one can only say that *a given model is not falsified (or invalidated) by the current input/output data.* On the other hand, a model is obviously *invalidated* or *falsified* once it is not compatible with the observations. Hence, we usually refer to *model falsification* rather than *model validation*, since the latter is not achievable in practice.

As an example, suppose that there are four possible models, $M\#1$, $M\#2$, $M\#3$, and $M\#4$, for a given plant. We are interested in deciding which model (if any) is able to explain the input/output data sequence that we are obtaining from the sensors and actuators' commands. Therefore, assume that, at a given initial time, t_o, all the four models are plausible, as depicted in Fig. 5.4. Further suppose that, at time t_1, model $M\#4$ is invalidated, *i.e.*, the sensors readings cannot be explained by model $M\#4$. Moreover, consider that, at time t_2, model $M\#2$ is invalidated and that, finally, model $M\#1$ is invalidated at time t_3. Then, at time t_3, we conclude that the only model capable of explaining the input/output time-series generated by the plant is model $M\#3$.

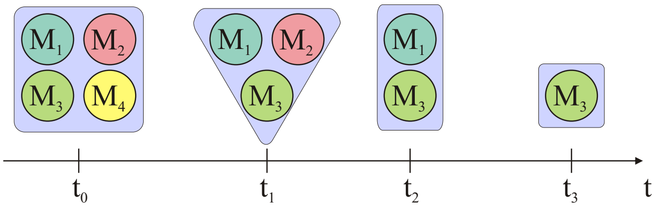


Figure 5.4: Example of the time-evolution of a set of models that are able to describe the input/output behavior of a given plant.

A natural assumption on the plant is that it is observable and, moreover, that it is controllable from the disturbances and control input. This guarantees that there are disturbances and/or control input signals that allow for the identification of the dynamic system at hand. Indeed, for LTI systems, the observability assures the ability to measure all the modes of the system, while controllability from the exogenous disturbances/control input guarantees that we are able to excite all the modes of the system.

5.2.1 Model Falsification in the Literature

Unmodeled dynamics (present in virtually every physical system) and adverse exogenous disturbances, can result in erroneous model falsification. Therefore, worst-case approaches, rather than stochastic approaches, are more suitable to address this type of problems. In fact, the solution proposed in [71] for uncertain LTI systems, and later on extended to LPV systems in [125], assumes that the system is described by an LTI nominal model interconnected with an LTI or LTV unknown system, denoted by Δ. This uncertain system Δ can be used, for instance, to describe unmodeled dynamics and parametric uncertainty. However, the methods provided in [71, 125] are not recursive, which means that, after a given amount of input/output data is obtained, we check whether or not the data is compatible with our model of the system. Hence, the complexity of the algorithms grows with the number of iterations.

A different approach to model falsification can be found in the Fault Detection and Isolation (FDI) literature. The main idea in such architectures stems from the designing of filters that are more sensitive to faults than to disturbances and model uncertainty. This can be achieved, for instance, by using geometric considerations regarding the plant [126–129], or by optimizing a particular norm minimization objective, such as the $\mathcal{H}_\infty$- or l_1-norm [130–134]. The latter approach provides, in general, important robustness properties, as stressed in [130, 135–137], by explicitly accounting for model uncertainty. After designing the filters, a set of residuals is then generated by comparing the actual output of the plant with the one estimated by each filter – see Fig. 5.5 for an example where one residual is generated from the difference between the actual plant output and the expected output for a non-faulty plant. A model is then invalidated if the corresponding residual is greater than a given threshold, which may be time-varying and that, in general, depends on the model uncertainty and on the amplitude of the disturbances. As a caveat, these methodologies are, in general, conservative or can only be applied to a restrictive class of systems.

The model falsification strategy presented in this thesis uses a philosophy similar to that of [71, 125], but proposes a recursive algorithm which can be used to run in real-time. As shown in the sequel, this method guarantees that valid models of the plant are never falsified. Moreover, under certain distinguishability conditions

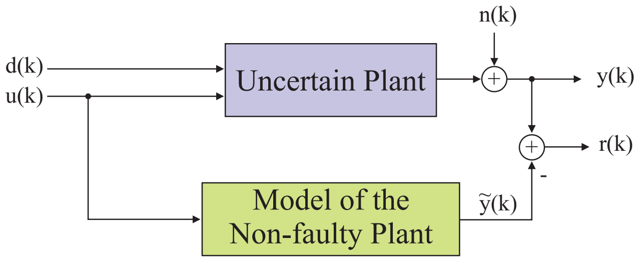


Figure 5.5: Residual generation in a classical Fault Detection (FD) architecture.

discussed in this chapter, it is also shown that the *correct* model of the plant is selected.

5.3 The (In)Distinguishability Problem

The *identifiability* of dynamic systems plays an important role in certain areas, where it is fundamental to ensure that the model of the plant can be inferred from input/output data. In particular, model estimation methods require the system to be identifiable. Otherwise, the estimation problem may not be well-posed and lead to erroneous results. Such methods also require, in general, a persistence of excitation type of condition in the exogenous inputs, in order to avoid the issues related to the indistinguishability due to the small amplitude of the disturbances.

In applications such as Fault Detection and Isolation (FDI) [131, 133, 136, 138, 139], identification of hybrid systems [140, 141], and Multiple-Model Adaptive Control (MMAC) [41, 58, 142], it suffices to guarantee that we can identify the family of systems to which the true plant belongs, among a finite set of families of dynamic systems. If we indeed can identify those families, we say that they are *distinguishable*. In particular, the FDI strategy presented in [143] "discards" models that are not compatible with the observations. This method guarantees that there will not be false alarms, as long as the model of the non-faulty plant remains valid. Moreover, we need not address the difficult problem of computing the decision threshold used to declare whether or not a fault has occurred. Nonetheless, we can only guarantee that a certain fault is going to be detected and isolated if certain assumptions on the distinguishability of the models is posed.

Therefore, the distinguishability analysis should be seen as a tool for the design of multiple-model systems, providing guarantees in terms of model selection.

An example of the (in)distinguishability problem is illustrated in Fig. 5.6. Let $\bar{n}$ denote the maximum amplitude of the measurement noise for a given dynamic system, and let S_A and S_B denote the dynamics of two *plausible* models describing this system. For small values of the measurement noise, *i.e.*, in this example, for $\bar{n} < \bar{n}_3$, the output sequences generated by systems S_A and S_B do not intersect each other. However, for $\bar{n} > \bar{n}_3$, there are output sequences that can be generated by either system S_A or system S_B.

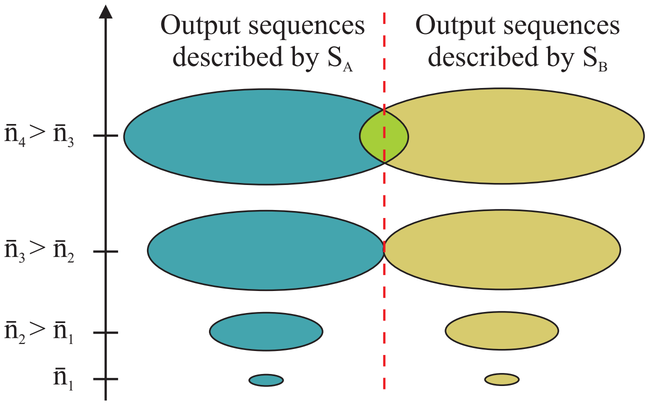


Figure 5.6: Output sequences described by two systems as a function of the maximum amplitude of the measurement noise.

The notion of *distinguishability* was firstly introduced in [144], where in fact the authors relate the concept of identifiability with that of distinguishability. Indeed, these two concepts (see, for instance, [144–146]) are naturally related to each other. If a system can be identified by using a certain input signal, then it can be distinguished from any other system with the same structure. A discussion on this topic can be found in [147].

However, most of the results on system identifiability are not enough for system *absolute distinguishability*. In particular, for two absolutely distinguishable systems we require, for *all* the admissible input signals, the corresponding outputs to be different from one another, unlike the most common definition of (structurally) identifiable system, which only demands that the unknown parameters of the plant can be estimated for *some* input signal.

The most common notion of (in)distinguishability [144, 147] states that two systems are indistinguishable only if there are neither initial conditions nor exogenous inputs that can generate different outputs for the systems. This definition, however, is not useful in applications where the distinguishability of the systems must be guaranteed from a worst-case perspective.

This section introduces a new definition of system distinguishability, referred to as *absolute distinguishability*, as detailed in the sequel. We also provide necessary and sufficient conditions for the absolute distinguishability of two dynamic LTI or LPV systems under different scenarios.

We are going to start by considering the broad class of time-varying discrete-time dynamic systems described by

$$y(k) = F_k\left(x_o, \phi_o, \phi_1, \phi_2, \cdots, \phi_k, p\right), \tag{5.1}$$

where $F_k : \mathbb{R}^n \times \overbrace{\Phi \times \cdots \times \Phi}^{k+1 \text{ times}} \times \Omega \to \mathbb{R}^{n_y}$, $\phi_i \in W \times U := (W_d \times W_n) \times U =: \Phi \subseteq \mathbb{R}^{n_u+n_d+n_n}$ for $i = 0, 1, \cdots, k$, and $p \in \Omega \subseteq \mathbb{R}^{n_p}$ is a vector of parameters. The sequence $(\phi_o, \phi_1, \cdots, \phi_k)$, where $\phi_i = [d_i^{\mathrm{T}}, n_i^{\mathrm{T}}, u_i^{\mathrm{T}}]^{\mathrm{T}}$, denotes the exogenous disturbances, $d_i \in W_d \subseteq \mathbb{R}^{n_d}$, measurement noise, $n_i \in W_n \subseteq \mathbb{R}^{n_n}$, and control input signals, $u_i \in U \subseteq \mathbb{R}^{n_u}$, at time instant i, and $y(k)$ is the output of the system at time k. The initial state is represented by $x_o := x(0) \in X(0) \subseteq \mathbb{R}^n$.

The notion of distinguishability in [144, 147] states that two realizations of F_k, parametrized by the pair of parameter vectors (p_A, p_B), are indistinguishable in N measurements only if there are neither initial conditions nor exogenous inputs that can generate different outputs for the systems parametrized by p_A and p_B. Hence, this definition is not useful in some applications, namely to provide guarantees that, regardless the input signals, the pair of parameter values is distinguishable.

Thus, a new definition of distinguishability, referred to as U-input distinguishability, was presented in [141]. Consider the dynamic systems S_A and S_B, which correspond to realizations of (5.1), for $p = p_A$ and $p = p_B$ and with outputs at time k denoted by $y_A(k)$ and $y_B(k)$, respectively. The definition in [141] states that, unless all the initial conditions and inputs are zero, if two systems are U-input distinguishable in N measurements, then the corresponding outputs must be different at some time instant k smaller than or equal to N. This guarantees that we can distinguish two such systems in, at most, N measurements, just by measuring the outputs. Nevertheless, exogenous disturbances, $d(k) \in W_d$, and measurement noise, $n(k) \in W_n$, are not taken into account in this definition of distinguishability.

Therefore, this motivated the introduction of the following definitions, which are going to be used extensively throughout this and the following chapters:

Definition 5.1. *Systems S_A and S_B are said* absolutely distinguishable in N measurements *if, for any non-zero*

$$\left(x_o^A, x_o^B, \phi_o, \cdots, \phi_N\right) \in \mathbb{R}^n \times \mathbb{R}^n \times \overbrace{\mathbb{R}^{n_d+n_n+n_u} \times \cdots \times \mathbb{R}^{n_d+n_n+n_u}}^{N+1 \ times},$$

there exists $k \in \{0, 1, \cdots, N\}$ such that

$$y_A(k) \neq y_B(k).$$

Moreover, two systems are said absolutely distinguishable *if there exists $N \geq 0$ such that they are absolutely distinguishable in N measurements.*

Definition 5.2. *Systems S_A and S_B are said* absolutely (X_o, U, W)-input distinguishable in N measurements *if, for any non-zero*

$$\left(x_o^A, x_o^B, \phi_1, \phi_2, \cdots, \phi_N\right) \in X_o \times X_o \times \overbrace{\Phi \times \cdots \times \Phi}^{N \ times},$$

where $\Phi := U \times W = U \times (W_n \times W_d)$, there exists $k \in \{0, 1, \cdots, N\}$ such that

$$y_A(k) \neq y_B(k).$$

Moreover, two systems are said absolutely (X_o, U, W)-input distinguishable *if there exists $N \geq 0$ such that they are absolutely (X_o, U, W)-input distinguishable in N measurements.*

These two definitions are important when we want to guarantee that, regardless of the input signals, two systems can be distinguished in a given number of measurements. This fact is going to be further stressed in the sequel.

5.3.1 Absolute Input-Distinguishability of LTI Systems

In this subsection, we are going to specialize the concept of absolute input-distinguishability for linear time-invariant (LTI) models and discuss some of the properties of this class of dynamic systems.

Let S_i be a discrete-time LTI dynamic system described by

$$S_i : \begin{cases} x_i(k+1) &= A_i x_i(k) + B_i u(k) + L_i d_i(k), \\ y_i(k) &= C_i x_i(k) + N_i n_i(k), \end{cases} \tag{5.2}$$

where $x_i(0) = x_o^i$, $x_i(k) \in \mathbb{R}^n$, $u_i(k) \in U \subseteq \mathbb{R}^{n_u}$, $d_i(k) \in W_d \subseteq \mathbb{R}^{n_d}$, $y_i(k) \in \mathbb{R}^{n_y}$ and $n_i(k) \in W_n \subseteq \mathbb{R}^{n_n}$. We assume that S_i is observable and that it is controllable from $[u^{\mathrm{T}}(\cdot),\, d^{\mathrm{T}}(\cdot)]^{\mathrm{T}}$.

Notice that, according to Definition 5.2, two systems are absolutely (X_o, U, W)-input distinguishable in N measurements if, for *all* the input signals in U, the corresponding outputs are different at least at some time instant $k \leq N$. This is obviously a stronger constraint than simply saying that the two systems are distinguishable whenever the corresponding outputs are different for a particular input sequence, as in [144, 147].

Remark 5.1: In the remainder of this chapter, we are going to use the terms *distinguishable* and *absolutely distinguishable* interchangeably, as we are only interested in assessing absolute distinguishability of systems. ◇

For the sake of simplicity, let us consider, for the time being, that $U = \mathbb{R}^{n_u}$, $W_d = \mathbb{R}^{n_d}$ and $W_n = \mathbb{R}^{n_n}$. The following theorem can be used to test whether or not two systems, denoted by S_A and S_B, *i.e.*, two systems described by (5.2) with $i \in \{A, B\}$, are absolutely distinguishable (in the sense of Definition 5.1).

Theorem 5.1. *Let*

$$M_N = \left[\begin{array}{cc|c|c|c} C_A & -C_B & & & \\ C_A A_A & -C_B A_B & & & \\ C_A A_A^2 & -C_B A_B^2 & \bar{Q} & \bar{R} & \bar{J} \\ \vdots & \vdots & & & \\ C_A A_A^N & -C_B A_B^N & & & \end{array}\right],$$

where $\bar{Q} = diag(Q, Q, \cdots, Q)$, $Q = \begin{bmatrix} N_A & -N_B \end{bmatrix}$,

$$\bar{R} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ R_1^1 & 0 & \cdots & 0 \\ R_1^2 & R_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ R_1^N & R_2^N & \cdots & R_N^N \end{bmatrix},$$

$$R_i^k = \begin{bmatrix} C_A A_A^{k-i} L_A & -C_B A_B^{k-i} L_B \end{bmatrix},$$

$$\bar{J} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ J_1^1 & 0 & \cdots & 0 \\ J_1^2 & J_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ J_1^N & J_2^N & \cdots & J_N^N \end{bmatrix},$$

$$J_i^k = \begin{bmatrix} C_A A_A^{k-i} B_A - C_B A_B^{k-i} B_B \end{bmatrix}.$$

Systems S_A and S_B are absolutely distinguishable in N measurements if and only if there does not exist a non-zero vector $v \in \mathbb{R}^{2n+2(N+1)n_n+N(n_u+2n_d)}$ such that $M_N v = 0$, i.e., *if and only if,*

$$rank(M_N) = 2n + 2(N+1)n_n + N(n_u + 2n_d).$$

Proof. Systems S_A and S_B are indistinguishable on $\{0, 1, \cdots, N\}$ if and only if there exist initial conditions, and valid disturbance and noise signals such that

$$\begin{cases} y_A(0) &= y_B(0), \\ y_A(1) &= y_B(1), \\ & \vdots \\ y_A(N) &= y_B(N). \end{cases}$$

Notice that

$$\begin{aligned}
y_i(0) &= C_i x_i(0) + N_i n_i(0), \\
y_i(1) &= C_i x_i(1) + N_i n_i(1) = \\
&= C_i \left[A_i x_i(0) + B_i u(0) + L_i d_i(0)\right] + N_i n_i(1), \\
&\vdots \\
y_i(N) &= C_i x_i(N) + N_i n_i(N) = \\
&= C_i \Big[A_i^N x_i(0) + A_i^{N-1} B_i u(0) + \\
&\quad A_i^{N-1} L_i d_i(0) + \cdots + B_i u(N-1) + L_i d_i(N-1)] + N_i n_i(N),
\end{aligned}$$

for $i = \{A, B\}$, which is equivalent to $M_N v = 0$, where

$$v^{\mathrm{T}} = [x_A(0)^{\mathrm{T}},\, x_B(0)^{\mathrm{T}},\, n_A(0)^{\mathrm{T}},\, n_B(0)^{\mathrm{T}} \cdots,\, n_A(N)^{\mathrm{T}}, n_B(N)^{\mathrm{T}},\, d_A(0)^{\mathrm{T}},$$
$$d_B(0)^{\mathrm{T}} \cdots,\, d_A(N-1)^{\mathrm{T}}, d_B(N-1)^{\mathrm{T}},\, u(0)^{\mathrm{T}},\, u(1)^{\mathrm{T}} \cdots,\, u(N-1)^{\mathrm{T}}].$$

Moreover, stating that $M_N v = 0$ if and only if $v = 0$, is equivalent to stating that matrix M_N has full column rank. □

The importance of this theorem is twofold: on the one hand, it provides necessary and sufficient conditions for the absolute distinguishability of two systems; on the other hand, it shows that we can solve the distinguishability problem using linear programming.

As remarked in [141] for continuous-time systems, the distinguishability of two systems implies their observability. Nevertheless, there are also results available in the literature that provide sufficient conditions for absolute distinguishability. For instance, the authors of [148] show that, if $\bar{J}$ has full column rank and the sets X_o and W are compact, then the systems can be distinguished for sufficiently large input control signals, regardless of their direction.

Indeed, we are going to pose the following assumptions throughout the remainder of this chapter, in order to derive several results that can be useful in practice.

Assumption 5.1: The set of *admissible initial states*, X_o, *i.e.*, the minimum set containing all the possible initial states x_o of the systems, is a convex polytope. □

Under Assumption 5.1, the set of initial states can be represented by

$$X_o = \mathrm{Set}(M_{X_o}, m_{X_o}) := \{x : M_{X_o} x \leq m_{X_o}\}.$$

Analogously, suppose that:

Assumption 5.2: The set of *admissible exogenous disturbances*, W_d, is a convex polytope, *i.e.*,

$$W_d = \mathrm{Set}(M_d, m_d).$$

□

Assumption 5.3: The set of *admissible measurement noise*, W_n, is a convex polytope, *i.e.*,

$$W_n = \mathrm{Set}(M_n, m_n).$$

□

Then, we can state the following theorem, which is an extension of Theorem 5.1 for convex and compact X_o and W:

Theorem 5.2. *Suppose that Assumptions 5.1–5.3 are satisfied. Then, systems S_A and S_B are absolutely $(X_o, \mathbb{R}^{n_u}, W)$-input distinguishable if and only if*

$$\underset{v \in \mathfrak{G}}{\forall} : v \neq 0 \Rightarrow \begin{bmatrix} M_N \\ -M_N \\ \tilde{M}_{X_o} \\ \tilde{M}_W \end{bmatrix} v \nleq \begin{bmatrix} 0 \\ 0 \\ \tilde{m}_{X_o} \\ m_W \end{bmatrix}, \tag{5.3}$$

where

$$\tilde{M}_{X_o} = \begin{bmatrix} diag(M_{X_o}, M_{X_o}) & 0 & 0 & 0 \end{bmatrix}, \tilde{m}_{X_o} = \begin{bmatrix} m_{X_o} \\ m_{X_o} \end{bmatrix},$$

$$\tilde{M}_W = \begin{bmatrix} 0 & diag(M_n, \cdots, M_n) & 0 & 0 \\ 0 & 0 & diag(M_d, \cdots, M_d) & 0 \end{bmatrix},$$

$$m_W = \begin{bmatrix} m_n^T & \cdots & m_n^T & m_d^T & \cdots & m_d^T \end{bmatrix}^T,$$

and $\mathfrak{G} = \mathbb{R}^{2n} \times \mathbb{R}^{2n_n \times (N+1)} \times \mathbb{R}^{2n_d N} \times \mathbb{R}^{n_u \times N}$.

Proof. Systems S_A and S_B are not absolutely $(X_o, \mathbb{R}^{n_u}, W)$-input distinguishable if and only if there exist

a) two vectors $x_{i_o} \in \mathrm{Set}(M_{X_o}, m_{X_o})$, $i = A, B$,

b) two sequences of disturbances vectors $\{d_i(0), d_i(1), \cdots, d_i(N-1)\}$, $i = A, B$, with $d_i(k) \in W_d$,

c) two sequences of measurement noise vectors $\{n_i(0), n_i(1), \cdots, n_i(N)\}$, $i = A, B$, with $n_i(k) \in W_n$,

d) and a sequence of control input vectors $\{u(0), u(1), \cdots, u(N-1)\}$, with $u(k) \in U$,

such that

$$\begin{aligned} M_N[x_{A_o}^{\mathrm{T}},\, x_{B_o}^{\mathrm{T}},\, n_A^{\mathrm{T}}(0),\, \cdots,\, n_A^{\mathrm{T}}(N),\, n_B^{\mathrm{T}}(0),\, \cdots,\, n_B^{\mathrm{T}}(N),& \\ d_A^{\mathrm{T}}(1),\, \cdots,\, d_B^{\mathrm{T}}(N),\, u^{\mathrm{T}}(1),\, \cdots,\, u^{\mathrm{T}}(N)]^{\mathrm{T}} = 0,& \end{aligned} \tag{5.4}$$

with

$$\begin{aligned} [x_{A_o}^{\mathrm{T}},\, x_{B_o}^{\mathrm{T}},\, n_A^{\mathrm{T}}(0),\, \cdots,\, n_A^{\mathrm{T}}(N),\, n_B^{\mathrm{T}}(0),\, \cdots,\, n_B^{\mathrm{T}}(N),& \\ d_A^{\mathrm{T}}(1),\, \cdots,\, d_B^{\mathrm{T}}(N),\, u^{\mathrm{T}}(1),\, \cdots,\, u^{\mathrm{T}}(N)]^{\mathrm{T}} \neq 0& \end{aligned}$$

Otherwise, we cannot find a non-zero solution to (5.4), which concludes the proof.

□

Although Theorem 5.2 provides necessary and sufficient conditions for distinguishability of systems with polytopic constraints on the disturbances and initial states, the condition in (5.3) is seldom satisfied in practice. To see this, consider that indeed there exists $v \neq 0$ satisfying $M_N v = 0$, but that

$$\begin{bmatrix} \tilde{M}_{X_o} \\ \tilde{M}_W \end{bmatrix} v \not\leq \begin{bmatrix} \tilde{m}_{X_o} \\ m_W \end{bmatrix},$$

with $\tilde{m}_{X_o} > 0$ and $m_W > 0$. Then, for sufficiently small $\alpha > 0$, we have

$$\begin{bmatrix} \tilde{M}_{X_o} \\ \tilde{M}_W \end{bmatrix} \alpha v \leq \begin{bmatrix} \tilde{m}_{X_o} \\ m_W \end{bmatrix}.$$

Moreover, $M_N \alpha v = \alpha M_N v = 0$, and thus vector $\tilde{v} = \alpha v \neq 0$ voids the distinguishability of the two systems.

Therefore, we add the following constraint on the disturbances intensity, which can be interpreted as a persistence of excitation type of condition

$$\frac{1}{N} \sum_{k=0}^{N} \|w(k)\|^2 \geq \gamma. \tag{5.5}$$

where $w^{\mathrm{T}}(k) = [d^{\mathrm{T}}(k), \quad n^{\mathrm{T}}(k)]^{\mathrm{T}}$, and $d(N) := 0$. Such a condition can be easily merged with (5.3), as explained in the sequel.

We start by introducing the Fourier-Motzkin elimination method, described in [149]. This method can be used to project polyhedral convex sets on to subspaces. Indeed, we are interested in projecting a polytope described by $\{x \in \mathbb{R}^{n_x} : Ax \leq b\}$ on to $\mathbb{R}^{\tilde{n}_x}$, where $\tilde{n}_x < n_x$.

Let

$$(A_{\mathrm{LFM}}, b_{\mathrm{LFM}}) := \mathrm{LFM}(A, b, n),$$

and

$$(A_{\mathrm{RFM}}, b_{\mathrm{RFM}}) := \mathrm{RFM}(A, b, n),$$

where LFM and RFM stand for the *left-* and *right-Fourier Motzkin* elimination methods, respectively, $n = n_x - \tilde{n}_x > 0$, and where A_{LFM}, b_{LFM}, A_{RFM} and b_{RFM}, satisfy, for all $\tilde{x} \in \mathbb{R}^{\tilde{n}_x}$,

$$A_{\mathrm{LFM}}\tilde{x} \leq b_{\mathrm{LFM}} \Leftrightarrow \exists_{x \in \mathbb{R}^n} : A \begin{bmatrix} x \\ \tilde{x} \end{bmatrix} \leq b,$$

and

$$A_{\mathrm{RFM}}\tilde{x} \leq b_{\mathrm{RFM}} \Leftrightarrow \exists_{x \in \mathbb{R}^n} : A \begin{bmatrix} \tilde{x} \\ x \end{bmatrix} \leq b.$$

At this point, we can state the following theorem:

Theorem 5.3. *Let*

$$(A_N, b_N) = RFM \left\{ LFM \left(\begin{bmatrix} M_N \\ -M_N \\ \tilde{M}_{X_o} \\ \tilde{M}_W \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ \tilde{m}_{X_o} \\ m_W \end{bmatrix}, 2n \right), n_u \right\}.$$

Further define

$$P_A = \frac{1}{N} diag(I_{n_n}, 0_{n_n}, I_{n_d}, 0_{n_d}, I_{n_n}, 0_{n_n}, \cdots, I_{n_d}, 0_{n_d}),$$

and

$$P_B = \frac{1}{N} diag(0_{n_n}, I_{n_n}, 0_{n_d}, I_{n_d}, 0_{n_n}, I_{n_n}, \cdots, 0_{n_d}, I_{n_d}),$$

and let $\gamma_{min} \geq 0$ be such that

$$\gamma_{min} \geq \max_{A_N x \leq b_N} x^T P_A x, \text{ and } \gamma_{min} \geq \max_{A_N x \leq b_N} x^T P_B x. \tag{5.6}$$

Then, systems S_A and S_B are (X_o, U, W)-input distinguishable in N measurements if

$$\frac{1}{N}\sum_{k=0}^{N} \|w(k)\|^2 > \gamma_{min}. \tag{5.7}$$

Proof. Suppose that (5.7) is indeed satisfied. Then, for all $w_i^* \in \mathbb{R}^{2(N+1)n_n+2Nn_d}$, $i = \{A, B\}$, such that

$$(w_A^*)^{\mathrm{T}} P_A w_A^* = \frac{1}{N}\sum_{k=0}^{N} \|w(k)\|^2,$$

and

$$(w_B^*)^{\mathrm{T}} P_B w_B^* = \frac{1}{N}\sum_{k=0}^{N} \|w(k)\|^2,$$

we have

$$(w_A^*)^{\mathrm{T}} P_A w_A^* > \gamma_{\min} \geq \max_{A_N x \leq b_N} x^{\mathrm{T}} P_A x,$$

and

$$(w_B^*)^{\mathrm{T}} P_B w_B^* > \gamma_{\min} \geq \max_{A_N x \leq b_N} x^{\mathrm{T}} P_B x.$$

Thus $A_N w_A^* \not\leq b_N$. and $A_N w_B^* \not\leq b_N$. □

Notice that 5.6 can be interpreted as a concave quadratic programming problem, which can be solved, for instance, by testing the solution at the vertices of the polytope $S = \{x : A_N x \leq b_N\}$ (*cf.* [150]).

Remark 5.2: The Fourier-Motzkin elimination method removes the dependence of the distinguishability of systems S_A and S_B on the initial state and control inputs. Hence, it reduces the number of variables in the optimization procedure. ◇

The constraint in (5.5) can be replaced by a similar condition on the intensity of the output, $y(.)$. To see this, we rewrite the output sequence as

$$\begin{aligned} y(k) &= Cx(k) + Nn(k), \quad k = 0, 1, \cdots, N \\ &= \begin{bmatrix} C & N \end{bmatrix} \begin{bmatrix} x(k) \\ n(k) \end{bmatrix}, \quad k = 0, 1, \cdots, N \\ &= \bar{C}\bar{x}(k), \quad k = 0, 1, \cdots, N, \end{aligned}$$

where

$$\begin{aligned}
x(1) &= Ax(0) + Bu(0) + Ld(0), \\
x(2) &= A^2x(0) + ABu(0) + ALd(0) + Bu(1) + Ld(1), \\
x(3) &= A^3x(0) + A^2Bu(0) + A^2Ld(0) + ABu(1) + ALd(1) + Bu(2) + Ld(2), \\
&\vdots \\
x(N) &= A^Nx(0) + A^{N-1}Bu(0) + A^{N-1}Ld(0) + \cdots + Bu(N-1) + Ld(N-1).
\end{aligned}$$

Hence,

$$\begin{aligned}
\bar{x}(0) &= \bar{I}v_N, \\
\bar{x}(1) &= \bar{A}(0)v_N, \\
&\vdots \\
\bar{x}(N) &= \bar{A}(N-1)v_N,
\end{aligned}$$

where

$$v_N^{\mathrm{T}} = [x^{\mathrm{T}}(0),\, n^{\mathrm{T}}(0),\, \cdots,\, n^{\mathrm{T}}(N),\, d^{\mathrm{T}}(0),\, \cdots,\, d^{\mathrm{T}}(N-1),\, u^{\mathrm{T}}(0),\, \cdots\, u^{\mathrm{T}}(N-1)]^{\mathrm{T}}$$

and

$$\bar{I} = \left[\begin{array}{c|cccc|cccc|cccc} I & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & I & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \end{array}\right],$$

$$\bar{A}(0) = \left[\begin{array}{c|cccc|cccc|ccccc} A & 0 & 0 & \cdots & 0 & L & 0 & \cdots & 0 & \cdots & B & 0 & \cdots & 0 \\ 0 & 0 & I & \cdots & 0 & 0 & 0 & \cdots & 0 & & 0 & 0 & \cdots & 0 \end{array}\right],$$

$$\bar{A}(N-1) = \left[\begin{array}{c|ccc|ccc|cccc} A^N & 0 & \cdots & 0 & A^{N-1}L & \cdots & L & \cdots & A^{N-1}B & \cdots & B \\ 0 & 0 & \cdots & I & 0 & \cdots & 0 & & 0 & \cdots & 0 \end{array}\right].$$

Then, we have that

$$\begin{aligned}
\theta &:= \tfrac{1}{N}\sum_{k=0}^{N} \|y(k)\|^2 \\
&= \tfrac{1}{N}\left[\bar{x}^{\mathrm{T}}(0)\bar{C}^{\mathrm{T}}\bar{C}\bar{x}(0) + \bar{x}^{\mathrm{T}}(1)\bar{C}^{\mathrm{T}}\bar{C}\bar{x}(1) + \cdots + \bar{x}^{\mathrm{T}}(N)\bar{C}^{\mathrm{T}}\bar{C}\bar{x}(N)\right] \\
&= \tfrac{1}{N}(v_N)^{\mathrm{T}}\left(\tilde{V}_N\right)^{\mathrm{T}}\mathcal{C}(N)\tilde{V}_Nv_N
\end{aligned}$$

where

$$\tilde{V}_N = \begin{bmatrix} \bar{I} \\ \bar{A}(0) \\ \bar{A}(1) \\ \vdots \\ \bar{A}(N-1) \end{bmatrix},$$

and

$$\mathcal{C}(N) = \begin{bmatrix} \bar{C}^{\mathrm{T}}\bar{C} & 0 & \cdots & 0 \\ 0 & \bar{C}^{\mathrm{T}}\bar{C} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \bar{C}^{\mathrm{T}}\bar{C} \end{bmatrix}.$$

Therefore, we are now in conditions of stating the following result, which allows us to assess distinguishability by measuring the intensity of the output of the plant.

Corollary 5.1. *Define $\tilde{P}_A = \frac{1}{N}\tilde{C}_A^N$ and $\tilde{P}_B = \frac{1}{N}\tilde{C}_B^N$, and let $\theta_{min} \geq 0$ be such that*

$$\theta_{min} \geq \max_{\tilde{A}_N x \leq \tilde{b}_N} x^T \tilde{P}_A x, \qquad \textit{and} \quad \theta_{min} \geq \max_{\tilde{A}_N x \leq \tilde{b}_N} x^T \tilde{P}_B x, \tag{5.8}$$

where

$$\tilde{A}_N = \begin{bmatrix} \begin{bmatrix} \tilde{M}_A^N & -\tilde{M}_B^N \end{bmatrix} \\ \begin{bmatrix} -\tilde{M}_A^N & \tilde{M}_B^N \end{bmatrix} \\ M_{X_o}^* \\ M_W^* \\ I_u \end{bmatrix}, \quad \tilde{b}_N = \begin{bmatrix} 0 \\ 0 \\ m_{X_o}^* \\ m_W^* \\ 0 \end{bmatrix},$$

$$M_{X_o}^* = \left[\begin{array}{c|c|c|c|c|c|c|c} M_{X_o} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & M_{X_o} & 0 & 0 & 0 \end{array}\right], \quad m_{X_o}^* = \begin{bmatrix} m_{X_o} \\ m_{X_o} \end{bmatrix},$$

$$M_W^* = \left[\begin{array}{c|c|c|c|c|c|c|c} 0 & D(M_n) & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & D(M_n) & 0 & 0 \\ 0 & 0 & D(M_d) & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & D(M_d) & 0 \end{array}\right],$$

$D(M_n) = diag(M_n)$, $D(M_d) = diag(M_d)$,

$$m_W^* = \begin{bmatrix} m_n^T & \cdots & m_n^T & m_d^T & \cdots & m_d^T \end{bmatrix}^T,$$

$$I_u = \left[\begin{array}{c|c|c|c|c|c|c|c} 0 & 0 & 0 & I & 0 & 0 & 0 & -I \\ 0 & 0 & 0 & -I & 0 & 0 & 0 & I \end{array}\right],$$

and, for $i \in \{A, B\}$,

$$\tilde{M}_i^N = \left[\begin{array}{c|c|c|c} C_i & & & \\ C_iA_i & diag(N_i) & \tilde{L}_i^N & \tilde{B}_i^N \\ \vdots & & & \\ C_iA_i^N & & & \end{array}\right],$$

$$\tilde{L}_i^N = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ C_iL_i & 0 & \cdots & 0 \\ C_iA_iL_i & C_iL_i & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ C_iA_i^{N-1}L_i & C_iA_i^{N-2}L_i & \cdots & C_iL_i \end{bmatrix},$$

$$\tilde{B}_i^N = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ C_iB_i & 0 & \cdots & 0 \\ C_iA_iB_i & C_iB_i & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ C_iA_i^{N-1}B_i & C_iA_i^{N-2}B_i & \cdots & C_iB_i \end{bmatrix},$$

$$\tilde{C}_i^N = \left(\tilde{V}_i^N\right)^T \mathcal{C}_i(N)\tilde{V}_i^N,$$

$$\tilde{V}_A^N = \left[\begin{array}{c|c} \bar{I} & \\ \bar{A}_B^0 & \\ \bar{A}_B^1 & 0 \\ \vdots & \\ \bar{A}_B^{N-1} & \end{array}\right], \quad \tilde{V}_B^N = \left[\begin{array}{c|c} & \bar{I} \\ & \bar{A}_B^0 \\ 0 & \bar{A}_B^1 \\ & \vdots \\ & \bar{A}_B^{N-1} \end{array}\right].$$

Then, systems S_A and S_B are (X_o, U, W)-input distinguishable in N measurements if

$$\frac{1}{N}\sum_{k=0}^{N} \|y(k)\|^2 > \theta_{min}. \tag{5.9}$$

Proof. Similarly to what was done in the proof of Theorem 5.3, if (5.9) is verified, then, for all x such that

$$x^T\tilde{P}_Ax = \frac{1}{N}\sum_{k=0}^{N} \|y(k)\|^2$$

or

$$x^T \tilde{P}_B x = \frac{1}{N} \sum_{k=0}^{N} \|y(k)\|^2,$$

we have

$$\tilde{A}_N x \not\leq \tilde{b}_N.$$

□

5.3.2 Absolute Input-Distinguishability of Uncertain Systems

In the previous subsection, we presented a series of results that can be used to assess the distinguishability of a pair of dynamic LTI systems. However, in most applications, the models of these systems are known only up to a certain level of accuracy. Hence, uncertain models are, in general, more suitable to describe these dynamic systems. Therefore, this section describes a series of methods that allow us to decide whether or not a pair of dynamic uncertain LTI systems is distinguishable or not. Computationally efficient solutions are also provided whenever possible.

Uncertainty in the A Matrix

Consider the class of dynamic systems, S_i, described by

$$S_i : \begin{cases} x_i(k+1) &= \left(A_0^i + \sum_{j=1}^{n_\Delta} \Delta_j^i(k) A_j^i \right) x_i(k) + B_i u(k) + L_i d_i(k), \\ y_i(k) &= C_i x_i(k) + N_i n_i(k), \end{cases} \tag{5.10}$$

where $x_i(0) = x_o^i$, $x_i(k) \in \mathbb{R}^n$, $u_i(k) \in U \subseteq \mathbb{R}^{n_u}$, $d_i(k) \in W_d \subseteq \mathbb{R}^{n_d}$, $y_i(k) \in \mathbb{R}^{n_y}$, $n_i(k) \in W_n \subseteq \mathbb{R}^{n_n}$, and $\Delta \in \mathbb{R}^{n_\Delta}$. We assume that S_i is observable and that it is controllable from $[u^{\mathrm{T}}(\cdot), d^{\mathrm{T}}(\cdot)]^{\mathrm{T}}$. Moreover, we assume that $|\Delta_j^i(k)| \leq 1$. The uncertainty vector, $\begin{bmatrix} \Delta_1^i(k) & \cdots & \Delta_{n_\Delta}^i(k) \end{bmatrix}^{\mathrm{T}}$, represents uncertainty in the dynamics of the plant, and can appear, for instance, in the modeling of several types of physical systems.

Posing the problem of absolute input-distinguishability of systems S_A and S_B, described by (5.10) with $i = A$ and $i = B$, respectively, can be done in a similar way as in the previous subsection, with the exception that in this case we have the product between some of the variables, namely $x_i(k)$ and $\Delta_j^i(k)$, for $j = 1, \cdots, n_\Delta$. Thus, the distinguishability problem can no longer be posed as a linear program.

Nevertheless, in the case where $\text{rank}(A_j^i) = 1$ for all $i = A, B$ and $j = 1, \cdots, n_\Delta$, the problem can be rewritten in such a way that the aforementioned bilinearity can be avoided. For the sake of comprehension, let us consider that $n_\Delta = 1$, although the results presented in the sequel can be readily extended for the general case. Assuming $\text{rank}(A_1^i) = 1$, there exist vectors e_1^i and f_1^i such that

$$A_1^i = e_1^i (f_1^i)^{\mathrm{T}}.$$

Moreover, define $z_i(k) := (f_1^i)^{\mathrm{T}} x_i(k) \Delta_i(k)$. Then, system S_i in (5.10), for $n_\Delta = 1$, can be rewritten as

$$S_i : \begin{cases} x_i(k+1) = A_0^i x_i(k) + e_i z_i(k) + B_i u(k) + L_i d_i(k), \\ y_i(k) = C_i x_i(k) + N_i n_i(k), \end{cases} \tag{5.11}$$

with the additional constraint

$$|z_i(k)| \leq \left|(f_1^i)^{\mathrm{T}} x_i(k)\right|. \tag{5.12}$$

Notice that (5.11), for arbitrary $z_i(k)$, provides a description of $y_i(N)$ which is linear in the variables $x_i(k)$, $z_i(k)$, $u(k)$, $d_i(k)$ and $n_i(k)$, for $k = 0, \cdots, N$. The constraint in (5.12) is nonlinear regarding $x_i(k)$. However, for given k, the constraint

$$|z_i(k)| \leq \left|(f_1^i)^{\mathrm{T}} x_i(k)\right|$$

can be rewritten as

$$\begin{cases} -(f_1^i)^{\mathrm{T}} x_i(k) \leq z_i(k) \leq (f_1^i)^{\mathrm{T}} x_i(k) \\ \qquad\qquad\qquad \text{or} \\ (f_1^i)^{\mathrm{T}} x_i(k) \leq z_i(k) \leq -(f_1^i)^{\mathrm{T}} x_i(k). \end{cases} \tag{5.13}$$

We recall that two systems S_A and S_B are absolutely (X_o, U, W)-input distinguishable in N measurements, if there do not exist vectors $x_i(k)$, $z_i(k)$, $u(k)$, $d_i(k)$ and $n_i(k)$, for $k = 0, \cdots, N$ and $i = A, B$, such that

$$y_A(k) = y_B(k), \tag{5.14}$$

for all $k \in \{0, \cdots, N\}$. Thus, the distinguishability problem can be solved through several linear programs, each designed for a specific region of $z_i(k)$. As an example, for $N = 1$, we would check whether or not there exist vectors $x_i(k)$, $z_i(k)$, $u(k)$, $d_i(k)$ and $n_i(k)$, for $k = 0, 1$ and $i = A, B$, such that (5.14) is satisfied for all $k \in \{0, 1\}$, and for $z_A(0)$ satisfying one of the conditions in (5.13) with $i = A$, and for $z_B(0)$ satisfying one of the conditions in (5.14) with $i = B$. In the present case, this leads to at most $\big(2(N+1)\big)^2$ linear programs to be solved.

If $\text{rank}(A_1^i) > 1$, then A_1^i can be written as the sum of several matrices with unitary ranks, as follows

$$A_1^i = e_{1,1}^i (f_{1,1}^i)^{\mathrm{T}} + e_{1,2}^i (f_{1,2}^i)^{\mathrm{T}} + \cdots + e_{1,m}^i (f_{1,m}^i)^{\mathrm{T}},$$

for some integer $m \geq 0$. Thereafter, define

$$z_j^i(k) := (f_{1,j}^i)^{\mathrm{T}} x_i(k) \Delta_i(k),$$

for each $j \in \{1, \cdots, m\}$. Then, each $z_j^i(k)$ can be treated as an independent uncertainty, and the previously described method can be applied. However, this approach can add conservatism to the solution, since we assume no relation between each $z_j^i(k)$.

Uncertainty in the B Matrix

In the sequel, the problem of accounting for uncertainty in the input matrix is analyzed. Consider a dynamic system, S_i, described by

$$S_i : \begin{cases} x_i(k+1) = A_i x_i(k) + \left(B_0^i + \displaystyle\sum_{j=1}^{n_\Delta} \Delta_j^i(k) B_j^i \right) u(k) + L_i d_i(k), \\ y_i(k) = C_i x_i(k) + N_i n_i(k), \end{cases} \tag{5.15}$$

where $x_i(0) = x_o^i$, $x_i(k) \in \mathbb{R}^n$, $u_i(k) \in U \subseteq \mathbb{R}^{n_u}$, $d_i(k) \in W_d \subseteq \mathbb{R}^{n_d}$, $y_i(k) \in \mathbb{R}^{n_y}$, $n_i(k) \in W_n \subseteq \mathbb{R}^{n_n}$, and $\Delta \in \mathbb{R}^{n_\Delta}$. It is also assumed that

$$|\Delta_j^i(k)| \leq 1.$$

In this case, the uncertainty vector, $\begin{bmatrix} \Delta_1^i(k) & \cdots & \Delta_{n_\Delta}^i(k) \end{bmatrix}^{\mathrm{T}}$, represents uncertainty in the input of the plant.

Define

$$F_j^i(k) := F_j^i(u(k)) := B_j^i u(k), \tag{5.16}$$

for $j \in \{1, \cdots, n_\Delta\}$. Then, by substituting (5.16) in (5.15), we obtain the following equivalent description for the system

$$S_i : \begin{cases} x_i(k+1) &= A_i x_i(k) + B_0^i u(k) + \sum_{j=1}^{n_\Delta} F_j^i(k)\Delta_j^i(k) + L_i d_i(k), \\ y_i(k) &= C_i x_i(k) + N_i n_i(k), \end{cases} \tag{5.17}$$

where $|\Delta_j^i(k)| \leq 1$. Therefore, each $\Delta_j^i(\cdot)$ can be seen as a bounded exogenous disturbance, acting upon the system.

Uncertainty in the C Matrix

Finally, let us consider that the matrix C is also uncertain. Consider a dynamic system, S_i, described by

$$S_i : \begin{cases} x_i(k+1) &= A_i x_i(k) + B_i u(k) + L_i d_i(k), \\ y_i(k) &= \left(C_0^i + \sum_{j=1}^{n_\Delta} \Delta_j^i(k) C_j^i \right) x_i(k) + N_i n_i(k), \end{cases} \tag{5.18}$$

where $x_i(0) = x_o^i$, $x_i(k) \in \mathbb{R}^n$, $u_i(k) \in U \subseteq \mathbb{R}^{n_u}$, $d_i(k) \in W_d \subseteq \mathbb{R}^{n_d}$, $y_i(k) \in \mathbb{R}^{n_y}$, $n_i(k) \in W_n \subseteq \mathbb{R}^{n_n}$, and $\Delta \in \mathbb{R}^{n_\Delta}$. It is also assumed that $|\Delta_j^i(k)| \leq 1$. In this case, the uncertainty vector, $\begin{bmatrix} \Delta_1^i(k) & \cdots & \Delta_{n_\Delta}^i(k) \end{bmatrix}^{\mathrm{T}}$, represents uncertainty in the output of the plant.

Notice that S_i is equivalent to

$$S_i \equiv \left(\bar{S}_j^i + N_i \bar{n}_i\right) + \sum_{j=1}^{n_\Delta} \left(\Delta_j^i \bar{S}_j^i + N_i \bar{n}_i\right), \tag{5.19}$$

where

$$\bar{S}_0^i : \begin{cases} x_0^i(k+1) &= A_i x_0^i(k) + B_i u(k) + L_i d_i(k), \\ y_0^i(k) &= C_0^i x_0^i(k), \end{cases}$$

and, for $j \in \{1, \cdots, n_\Delta\}$,

$$\bar{S}_j^i : \begin{cases} x_j^i(k+1) &= A_i x_j^i(k) + B_i u(k) + L_i d_i(k), \\ y_j^i(k) &= C_j^i x_j^i(k), \end{cases}$$

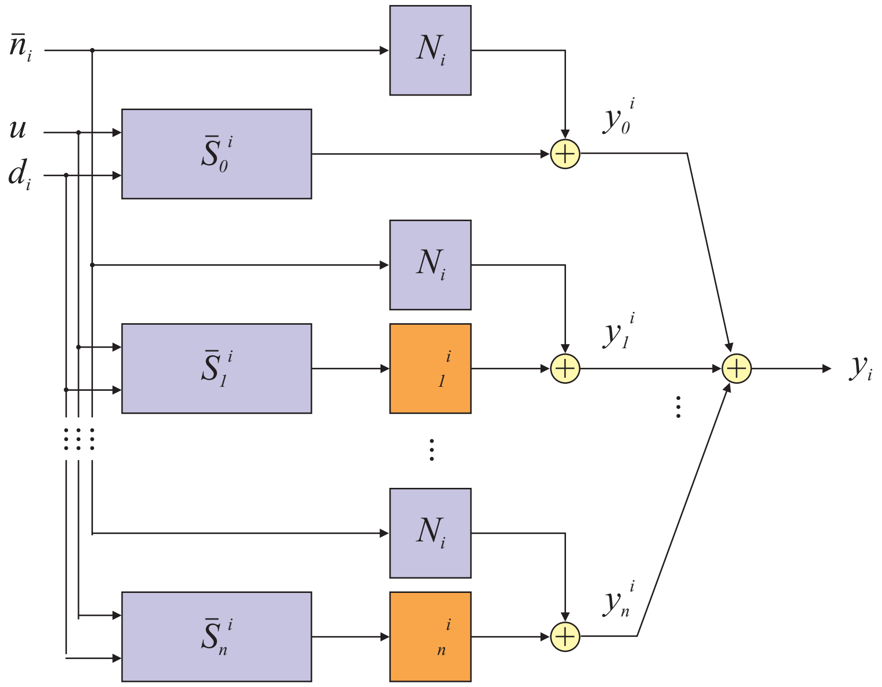


Figure 5.7: Block diagram of an LTI system with uncertainties in the output.

with $x_j^i(0) = x_i(0)$ for all $j \in \{0, \cdots, n_\Delta\}$, and $\bar{n}_i = \frac{n_i}{n_\Delta + 1}$. The block diagram of (5.19) is depicted in Fig. 5.7.

Since each $\bar{S}_j^i$, for $j \in \{0, \cdots, n_\Delta\}$, is a linear system, and each $\Delta_j^i(k)$, for $j \in \{1, \cdots, n_\Delta\}$ and $k \geq 0$, is an uncertain scalar, we obtain

$$S_i \equiv \left(\bar{S}_j^i + N_i \bar{n}_i\right) + \sum_{j=1}^{n_\Delta} \left(\tilde{S}_j^i + N_i \bar{n}_i\right), \tag{5.20}$$

where

$$\tilde{S}_j^i : \begin{cases} x_j^i(k+1) = A_i x_j^i(k) + B_i \Delta_j^i(k) u(k) + L_i \Delta_j^i(k) d_i(k), \\ y_j^i(k) = C_j^i x_j^i(k). \end{cases}$$

Notice that (5.20) describes an LTI system with uncertain input. Nevertheless, the exogenous disturbances are now multiplied by the uncertainties $\Delta_j^i(k)$, and hence $\tilde{S}_j^i$ depends upon $\Delta_j^i(k)$ and $d_i(k)$ in a bilinear fashion. However, this can be avoided by introducing the following relaxation. Since $|\Delta_j^i(k)| \leq 1$, we have that

$$\tilde{d}_j^i(k) := \Delta_j^i(k) d_i(k) \Rightarrow \|\tilde{d}_j^i(k)\| \leq \|d_i(k)\|. \tag{5.21}$$

Thus, by substituting $\Delta_j^i(k) d_i(k)$ in (5.20) by $\tilde{d}_j^i(k)$ as in (5.21), we obtain a description of the system which is linear in the unknown variables, at the cost of some conservatism due to the implication in (5.21).

5.3.3 Absolute Input-Distinguishability of LPV Systems

As shown in the sequel, the methods presented in the previous subsections can also be used to assess distinguishability of linear time-varying plants. In particular, we are going to extend the aforementioned results to Linear Parameter-Varying (LPV) models. As mentioned in Chapter 2, these LPV (see [62]) models represent nowadays a compromise between the global accuracy of nonlinear models and the straightforward controller synthesis and system analysis techniques available for LTI representations.

We start by extending Theorem 5.1 to LPV plants and present a systematic method to extend the remaining results of the previous subsection to LPV plants.

Consider an LPV dynamic system described by

$$S_i : \left\{ \begin{array}{rcl} x_i(k+1) & = & A_i\big(\rho(k)\big)x_i(k) + B_i\big(\rho(k)\big)u(k) + L_i\big(\rho(k)\big)d_i(k), \\ y_i(k) & = & C_i\big(\rho(k)\big)x_i(k) + N_i\big(\rho(k)\big)n_i(k), \end{array} \right. \tag{5.22}$$

where $x_i(0) = x_o^i$, $x_i(k) \in \mathbb{R}^n$, $u_i(k) \in U \subseteq \mathbb{R}^{n_u}$, $d_i(k) \in W_d \subseteq \mathbb{R}^{n_d}$, $y_i(k) \in \mathbb{R}^{n_y}$ and $n_i(k) \in W_n \subseteq \mathbb{R}^{n_n}$. Moreover, we assume that $\rho : \mathbb{Z}_o^+ \to \Omega \subset \mathbb{R}^{n_\rho}$, and that $\rho(k)$ is a vector of parameters for each $k \in \{0, 1, \cdots\}$. We denote by Ω the space of parameters. It is also assumed that this LPV system is observable and controllable from the disturbances and control input, for all admissible $\rho(\cdot)$. For the sake of simplicity of the notation, we redefine

$$\left\{ \begin{array}{rcl} A_i\big(\rho(k)\big) & := & A_k^i, \\ B_i\big(\rho(k)\big) & := & B_k^i, \\ L_i\big(\rho(k)\big) & := & L_k^i, \\ C_i\big(\rho(k)\big) & := & C_k^i, \\ N_i\big(\rho(k)\big) & := & N_k^i. \end{array} \right.$$

We are now in conditions of stating the following theorem, which is an extension of Theorem 5.1 to LPV plants.

Theorem 5.4. *Let*

$$\mathcal{M}_N =$$

$$\left[\begin{array}{cc|c|c|c} C_0^A & -C_0^B & & & \\ C_1^A A_0^A & -C_1^B A_B^0 & & & \\ C_2^A A_1^A A_0^A & -C_2^B A_1^B A_0^B & \bar{\mathcal{Q}}_N & \bar{\mathcal{R}}_N & \bar{\mathcal{J}}_N \\ \vdots & \vdots & & & \\ C_N^A A_{N-1}^A \cdots A_0^A & -C_N^B A_{N-1}^B \cdots A_0^B & & & \end{array}\right],$$

where

$$\bar{\mathcal{Q}}_N = diag(\mathcal{Q}_0, \mathcal{Q}_1, \cdots, \mathcal{Q}_N),\ \mathcal{Q}_j = \begin{bmatrix} N_j^A & -N_j^B \end{bmatrix},$$

$$\bar{\mathcal{R}}_N = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \mathcal{R}_1^1 & 0 & \cdots & 0 \\ \mathcal{R}_1^2 & \mathcal{R}_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{R}_1^N & \mathcal{R}_2^N & \cdots & \mathcal{R}_N^N \end{bmatrix},$$

$$\mathcal{R}_i^k = \begin{bmatrix} C_k^A A_{k-i-1}^A \cdots A_0^A L_{k-i-1}^A & -C_k^B A_{k-i-1}^B \cdots A_0^B L_{k-i-1}^B \end{bmatrix},$$

$$\bar{\mathcal{J}}_N = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \mathcal{J}_1^1 & 0 & \cdots & 0 \\ \mathcal{J}_1^2 & \mathcal{J}_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{J}_1^N & \mathcal{J}_2^N & \cdots & \mathcal{J}_N^N \end{bmatrix},$$

$$\mathcal{J}_i^k = \begin{bmatrix} C_k^A A_{k-i-1}^A \cdots A_0^A B_{k-i-1}^A - C_k^B A_{k-i-1}^B \cdots A_0^B B_{k-i-1}^B \end{bmatrix}.$$

Systems S_A and S_B are absolutely distinguishable in N measurements if and only if there does not exist a non-zero vector $v \in \mathbb{R}^{2n+2(N+1)n_n+N(n_u+2n_d)}$ such that

$$\mathcal{M}_N v = 0,$$

i.e., *if and only if,*

$$rank(\mathcal{M}_N) = 2n + 2(N+1)n_n + N(n_u + 2n_d).$$

Proof. The proof is similar to that of Theorem 5.1 and thus is omitted here. □

The same line of thought used to extend Theorem 5.1 to Theorem 5.4 can be used to obtain generalizations of the aforementioned results to LPV plants. Indeed, if instead of considering time-invariant matrices for the dynamics of the system, we assume that they can vary with time, then all the results follow naturally.

5.4 Simulations

In this section, we illustrate the concepts introduced in this chapter with a series of examples. The idea is to obtain the value of the minimum intensity of the disturbances that guarantees the distinguishability of a pair of dynamic systems, in a given number of measurements.

Example I: Different C Matrices – Case 1

Consider the LTI discrete-time systems described by

$$\begin{aligned} S_A : &\begin{cases} x(k+1) &= \begin{bmatrix} 0.1 & 1 \\ 0 & 0.2 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 1 \end{bmatrix} d(k), \\ y(k) &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x(k) + n(k), \end{cases} \\ S_B : &\begin{cases} x(k+1) &= \begin{bmatrix} 0.1 & 1 \\ 0 & 0.2 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 1 \end{bmatrix} d(k), \\ y(k) &= \begin{bmatrix} 1 & 0 \\ 0 & 0.6 \end{bmatrix} x(k) + n(k), \end{cases} \end{aligned} \tag{5.23}$$

where $|d| \leq 1$ and $|n| \leq 0.001$. Notice that system S_B can be seen as system S_A with a sensor failure. Further suppose that the only information regarding the initial state of the systems is $\|x(0)\| \leq 1$. Moreover, let

$$\gamma = \sum_{k=0}^{N} |d(k)|$$

denote the *intensity of the disturbances.*

Our goal is to obtain the value of the minimum intensity of the disturbances, $\gamma_{\min}$, such that systems S_A and S_B are absolutely distinguishable for a given number of measurements. The solution to this problem can be obtained using Theorem 5.3 and is illustrated in Fig. 5.8. Notice that in this case we only considered the disturbances intensity, while no assumptions have been posed regarding the intensity of the measurement noise. As expected, the value of $\gamma_{\min}$ decreases with the number of measurements, N.

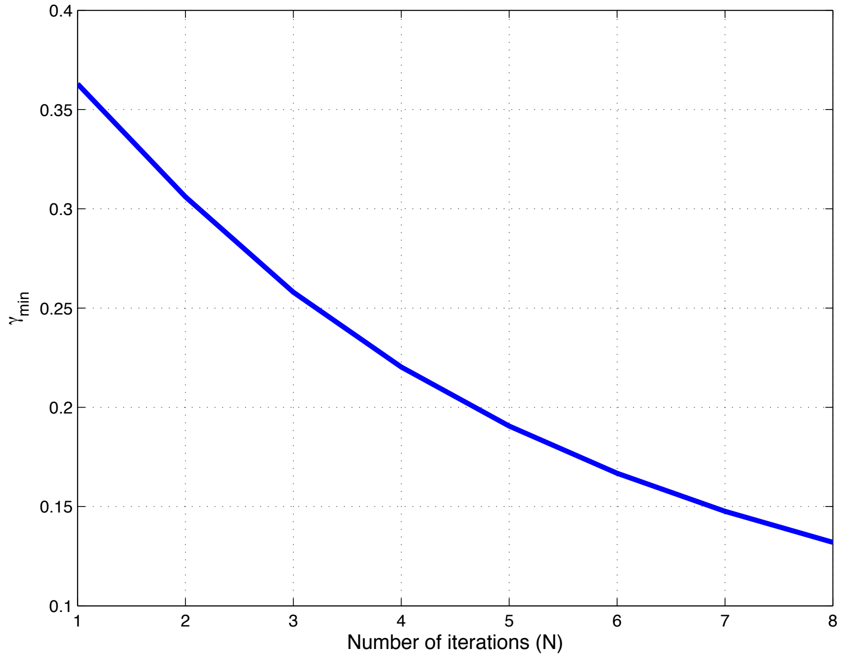


Figure 5.8: Minimum disturbances intensity that guarantee the distinguishability between systems S_A and S_B in Example I.

As an example, if $\gamma_{\min} > 0.15$, then we can be sure that it is possible to distinguish between S_A and S_B.

Example II: Different C Matrices – Case 2

In this second example we consider a harder distinguishability problem. Indeed, let us define

$$S_A \begin{cases} x(k+1) &= \begin{bmatrix} 0.1 & 1 \\ 0 & 0.2 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 1 \end{bmatrix} d(k), \\ y(k) &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x(k) + n(k), \end{cases}$$

$$S_B \begin{cases} x(k+1) &= \begin{bmatrix} 0.1 & 1 \\ 0 & 0.2 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 1 \end{bmatrix} d(k), \\ y(k) &= \begin{bmatrix} 1 & 0 \\ 0 & 0.9 \end{bmatrix} x(k) + n(k), \end{cases}$$

where $|d| \leq 1$ and $|n| \leq 0.001$. In comparison with Example I, the C matrices of systems S_A and S_B are much *closer.* In fact, while Example I can illustrate a sensor loss-of-effectiveness of 40%, Example II can represent a sensor loss-of-effectiveness of only 10%. Repeating the design procedures as in Example I, we obtain the minimum intensity of the disturbances, depicted in Fig. 5.9, such that the absolute distinguishability of the two systems is guaranteed.

As expected, for the same intensity of the disturbances, we require a higher number of measurements to guarantee the distinguishability between the two systems, when compared to Example I. Alternatively, for the same number of measurements, we require a higher intensity of the disturbances to guarantee the distinguishability.

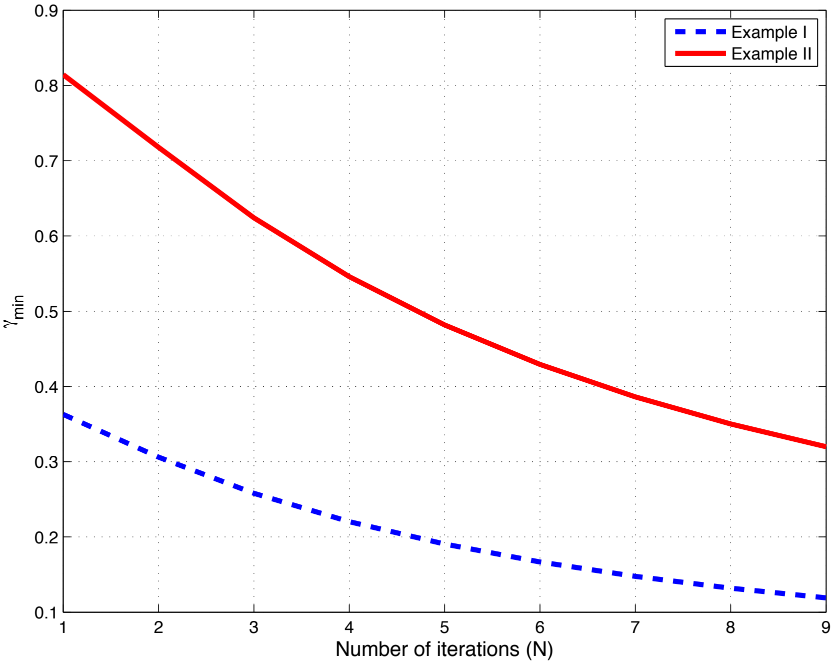


Figure 5.9: Minimum disturbances intensity that guarantee the distinguishability between systems S_A and S_B in Examples I and II. The results from Example I are also shown for comparison purposes.

Example III: Different A Matrices

The distinguishability problem can become even much harder when the differences between the systems lay in the dynamics matrix, A. For instance, consider that

$$S_A \begin{cases} x(k+1) &= \begin{bmatrix} 0.1 & 1 \\ 0 & 0.2 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 1 \end{bmatrix} d(k), \\ y(k) &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x(k) + n(k), \end{cases}$$

$$S_B \begin{cases} x(k+1) &= \begin{bmatrix} 0.1 & 1 \\ 0 & 0.3 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 1 \end{bmatrix} d(k), \\ y(k) &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x(k) + n(k), \end{cases}$$

where $|d| \leq 1$ and $|n| \leq 0.001$. In comparison with the former examples, the C matrices of systems S_A and S_B are the same, but the A matrices differ in one element. This example can illustrate a change in the dynamics of the system. Repeating the design procedures as in Examples I and II, we obtain the minimum intensity of the disturbances, depicted in Fig. 5.10, such that the absolute distinguishability of the two systems is guaranteed.

As seen in Fig. 5.10, the distinguishability of systems S_A and S_B is not guaranteed for a number of measurements smaller than 2, despite of the intensity of the disturbances. However, for $N \geq 2$, the systems are absolutely distinguishable if the intensity of the disturbances is higher than 0.5. This can be explained by the fact that the A matrices of S_A and S_B have a common eigenvalue. Therefore, if the intensity of the disturbances is small, we might be exciting only one of the modes of the systems. If, however, the intensity of the disturbances is large enough (and since they are constrained by a bound on the amplitude), we guarantee that we are exciting all the modes of the system and, thus, we can ensure the absolute distinguishability of S_A and S_B.

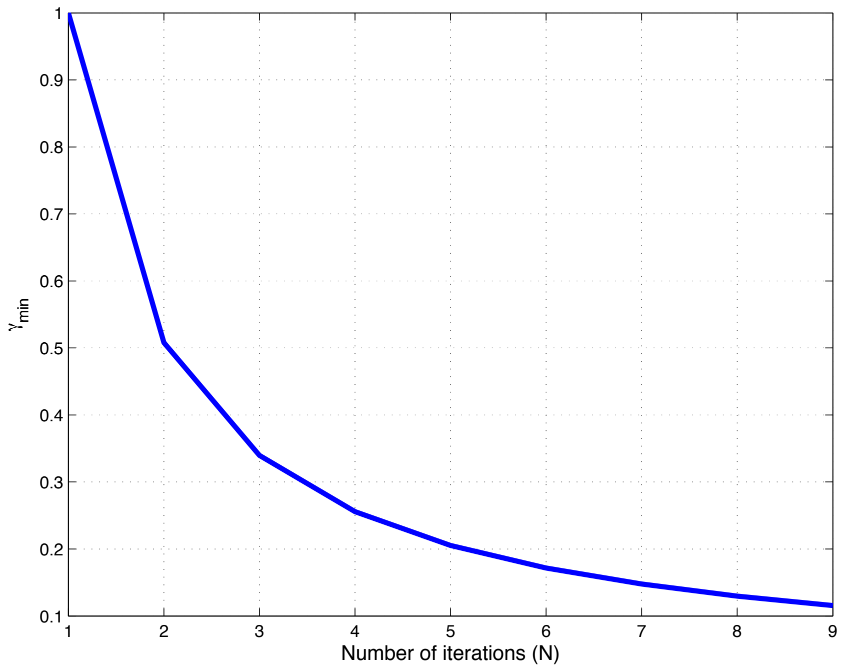


Figure 5.10: Minimum disturbances intensity that guarantee the distinguishability between systems S_A and S_B in Example III.

5.5 Conclusions

In this chapter, we dealt with the problem of model falsification and stated that its wide applicability ranges from Fault Detection and Isolation (FDI), to Multiple-Model Adaptive Control (MMAC). Guaranteeing that all except the "correct" model of the plant are falsified naturally led to the concept of *absolutely distinguishable* systems. We introduced the notion of absolutely distinguishable discrete-time dynamic systems, highlighting its applicability to Linear Time-Invariant (LTI) and to Linear Parameter-Varying (LPV) models. We further demonstrated that, in general, a persistence type of excitation condition is required on the exogenous disturbances. It turned out that this condition can be written as a lower bound on the intensity of the perturbations. Necessary and sufficient conditions for the distinguishability of two systems were also derived. The theory was illustrated with a set of examples that demonstrate the applicability of the results presented.

We argue that the concept of absolute distinguishability can be used as a tool for the design of model falsification schemes, in an analogous manner to the use of the concepts of observability and controllability for the synthesis of observers and controllers, respectively.

Chapter 6

Set-Valued Observers

6.1 Introduction

The previous chapter introduced the decision problem, which is common to all Multiple-Model Adaptive Control (MMAC) methodologies, and suggested the *model falsification* concept as one possible solution. This approach stems, on the one hand, from the premise that a dynamic model can never be *validated* in practice, which means that a model describing the plant up to a given time instant, may not be able to describe the subsequent input/output behavior. On the other hand, if a dynamic model is not able to explain the output of the actual system, given the applied control inputs and bounds on the exogenous disturbances, it is straightforward to conclude that such model is not compatible with the actual dynamics of the plant. Moreover, the previous chapter also introduced the notion of *absolute input distinguishability*, which is key in order to guarantee that a given dynamic model is falsified. Nevertheless, it was neither described how to falsify (or invalidate) dynamic models, nor how to design filters that can falsify models once the system is absolutely input distinguishable.

Therefore, this chapter is devoted to the description of a technique that is able to systematically design filters, which, in turn, are going to be used, in this thesis, for model falsification. These filters – which were suggested by Professor Jeff S. Shamma as an alternative decision subsystem – are referred to as Set-Valued Observers (SVOs) – see [151–154] and references therein for an overview on SVOs –, as they are able to provide set-valued estimates of the state of the plant, based upon

- the dynamic model of the system (which may be uncertain);
- the output measurements;
- the control inputs;
- and the bounds on the exogenous disturbances and measurement noise.

This type of observers, jointly with the model falsification paradigm described in Chapter 5, naturally arises as a solution to distinguish among models of dynamic systems. In particular, it is going to be shown in Chapter 7 that, under certain conditions, two systems can be distinguished using SVOs if they are absolutely distinguishable in the sense of Chapter 5.

The problem of designing set-valued observers – also referred to as set-membership filtering design – has been extensively studied in the literature. One of the first algorithms developed to compute (ellipsoidal) set-valued estimates of the state of a system was introduced in [152] and [153]. In [155], an approach to the synthesis problem of SVOs for LTV plants with nonlinear equality constraints is described. A method for active mode observation of switching systems, based on SVOs, has been recently proposed in [148].

Nevertheless, the problem of deriving the set of *all* the admissible states of a plant with an uncertain model has not been addressed, to the best of our knowledge, from a *worst-case* perspective. In other words, we would like to be able to compute, at each sampling time, the set that contains *all* the possible states of the plant, regardless of the sequence of exogenous disturbances and sensor noise.

The set-valued observers developed in this thesis are an extension of the work in [76, ShammaTu]. In fact, the results in [142, 143, 156] are a generalization of the set-valued state estimation for LPV systems (see [62]), which is able to handle model uncertainty. Indeed, this chapter describes how to design SVOs which are able to provide set-valued estimates of the state, under different scenarios, namely

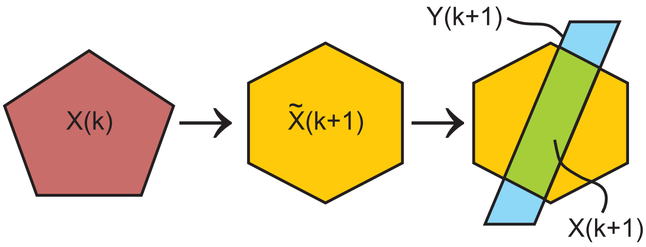


Figure 6.1: Predict and update cycles.

parametric uncertainty in the input, output or dynamics matrices of the state-space representation of the plant.

As illustrated in Fig. 6.1, the SVOs predict cycle consists in estimating the set of possible states, $\tilde{X}(k+1)$, at time $k+1$, based upon the model of the system and the set-valued estimate of the state at time k. The update cycle comprises the computation of the states, $Y(k+1)$, which are compatible with the measured output of the plant, and the intersection of this set with $\tilde{X}(k+1)$. In this chapter, it is demonstrated how to compute all these steps at once.

This chapter also addresses some of the computational and numerical issues related to the use of SVOs. Indeed, this type of observers may require several cumbersome computations, such as the union of convex polytopes and the application of the Fourier-Motzkin elimination method [149]. Therefore, as discussed in the sequel, the use of SVOs in many applications is not viable due to these large *on-line* computational requirements.

As a final remark, we assume discrete time-domain descriptions of the dynamic systems throughout this chapter, since the intersection of sets $\tilde{X}(k+1)$ and $Y(k+1)$, depicted in Fig. 6.1, cannot be performed continuously.

6.1.1 Main Contributions and Organization

This chapter thoroughly describes a technique to systematically synthesize set-valued observers for uncertain linear parameter-varying systems.

The main contributions of this chapter are as follows:

a) The extension of the work on set-valued observers to uncertain LPV plants;

b) The description of solutions to the main numerical and computational issues related to the SVOs;

c) The derivation of sufficient conditions for the convergence of the SVOs.

The remainder of this chapter is organized as follows. We start by introducing the notation used in this chapter and describing some of the techniques available in the literature for the design of SVOs in Section 6.2. An extension of the SVOs for uncertain models is presented in Section 6.3. In Section 6.4, several computational and numerical issues related to the SVOs are addressed. The theory is illustrated by means of an example, in Section 6.5. Finally, some conclusions regarding this chapter are discussed in Section 6.6.

6.2 Preliminaries and Notation

We represent the elements of $v(k) \in \mathbb{R}^m$, for some $m, k \in \mathbb{Z}, m > 0$, as $v_i(k)$, so that

$$v(k) = [v_1(k),\, v_2(k),\, \cdots,\, v_m(k)]^{\mathrm{T}}.$$

The concatenation of vectors $v(k)$, $v(k-1)$, $\cdots$, $v(k-N+1)$, for $N \in \mathbb{Z}^+$ is denoted as

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

For the sake of simplicity, v is used instead of v_N whenever N can be inferred from the context.

We assume that the available input/output dataset can be obtained through a Linear Parameter-Varying (LPV) system, described by

$$\begin{cases} x(k+1) &= A\big(\rho(k)\big)x(k) + B\big(\rho(k)\big)u(k) + L\big(\rho(k)\big)d(k), \\ y(k) &= C\big(\rho(k)\big)x(k) + N\big(\rho(k)\big)n(k), \end{cases} \tag{6.1}$$

with bounded exogenous disturbances, $d(\cdot)$, uncertain initial state, $x(0) \in X(0)$, control input, $u(\cdot)$, and measurement output, $y(\cdot)$, corrupted by additive noise, $n(\cdot)$. The system is assumed observable and controllable from the exogenous disturbances and control inputs, for all admissible $\rho(\cdot)$. The matrices of the system may be uncertain and are assumed to depend upon a time-varying vector of parameters, $\rho(\cdot)$. It is also assumed that

$$|d(k)| := \max_i |d_i(k)| \leq 1,$$

and

$$|n(k)| := \max_i |n_i(k)| \leq \bar{n}.$$

At each sampling time, k, the vector of states is denoted by $x(k)$, and we define

$$X(0) := \text{Set}(M_0, m_0),$$

where

$$\text{Set}(M, m) := \{q : Mq \leq m\} \tag{6.2}$$

represents a convex polytope. Moreover, let $x(k) \in \mathbb{R}^n$, $d(k) \in \mathbb{R}^{n_d}$, $u(k) \in \mathbb{R}^{n_u}$, and $y(k) \in \mathbb{R}^{n_y}$, for $k \geq 0$. For the sake of simplicity, we redefine

$$\begin{cases} A(k) & := & A\big(\rho(k)\big), \\ B(k) & := & B\big(\rho(k)\big), \\ L(k) & := & L\big(\rho(k)\big), \\ C(k) & := & C\big(\rho(k)\big), \\ N(k) & := & N\big(\rho(k)\big). \end{cases}$$

Let $X(k+1)$ represent the set of possible states at time $k+1$, *i.e.*, the state $x(k+1)$ verifies (6.1) with $x(k) \in X(k)$ if and only if $x(k+1) \in X(k+1)$. An SVO aims to find $X(k+1)$, based upon (6.1) and with the additional knowledge that $x(k) \in X(k), x(k-1) \in X(k-1), \cdots, x(k-N) \in X(k-N)$ for some finite N. We further require that for all $x \in X(k+1)$, there exists $x^\star \in X(k)$ such that, for $x(k) = x^\star$, the observations are compatible with (6.1). In other words, we want $X(k+1)$ to be the smallest set containing all the solutions to (6.1). A procedure for time-varying discrete-time linear systems was introduced in [76, ShammaTu], and a preliminary extension to uncertain plants is presented in [142].

The computation of $X(k+1)$ based upon $X(k)$ for systems with no model uncertainty can be performed using the technique described in [76, ShammaTu].

Indeed, let the system be described by (6.1), and assume that the matrices of the dynamics are exactly known. For the sake of simplicity, assume that $N\big(\rho(k)\big) = I$ for all $\rho(k), k \geq 0$.[1] Then, as shown in [76, ShammaTu], $x(k+1) \in X(k+1)$ if and only there exist $x(k)$, $n(k)$ and $d(k)$, such that, for the current measurement, $y(k+1)$, we have

$$P(k) \begin{bmatrix} x(k+1) \\ x(k) \\ d(k) \end{bmatrix} \leq \begin{bmatrix} B(k)u(k) \\ -B(k)u(k) \\ \mathbf{1} \\ \mathbf{1} \\ \tilde{m}(k) \\ m(k-1) \end{bmatrix} =: p(k) \tag{6.3}$$

where

$$P(k) := \begin{bmatrix} I & -A(k) & -L_d(k) \\ -I & A(k) & L_d(k) \\ 0 & 0 & I \\ 0 & 0 & -I \\ \tilde{M}(k) & 0 & 0 \\ 0 & M(k-1) & \end{bmatrix}, \; \tilde{M}(k) = \begin{bmatrix} C(k+1) \\ -C(k+1) \end{bmatrix},$$

$$\tilde{m}(k) = \begin{bmatrix} \bar{n} + y(k+1) \\ \bar{n} - y(k+1) \end{bmatrix},$$

and where $M(k-1)$ and $m(k-1)$ are defined such that $X(k) = \text{Set}\,(M(k-1), m(k-1))$. The inequality in (6.3) provides a description of a set in $\mathbb{R}^{2n+n_d}$, denoted by

$$\Gamma(k+1) = \text{Set}\,(P(k), p(k))\,.$$

[1] The case $N\big(\rho(\cdot)\big) \neq I$ can also be handled, by redefining the measurement noise as $n^*(k) := N\big(\rho(k)\big)n(k)$, for each $k \geq 0$, with bounds for the i-th component given by $\bar{n}_i^*(k) = \max\limits_{|n| \leq \bar{n}} \Big(N\big(\rho(k)\big)n\Big)_i$.

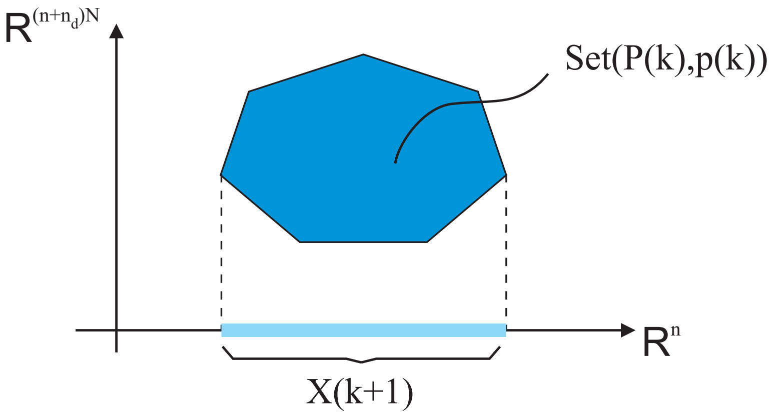


Figure 6.2: Projection of the set $\Gamma(k+1)$ onto $\mathbb{R}^n$.

Therefore, it is straightforward to conclude that

$$\hat{x} \in X(k+1) \Leftrightarrow \underset{x\in\mathbb{R}^n, d\in\mathbb{R}^{n_d}}{\exists} : \begin{bmatrix} \hat{x} \\ x \\ d \end{bmatrix} \in \Gamma(k+1)$$

Hence, the set $X(k+1)$ can be obtained by projecting $\Gamma(k+1)$ onto the subspace of the first n coordinates, as illustrated in Fig. 6.2.

The projection of $\Gamma(k+1)$ onto $\mathbb{R}^n$ can be done resorting to the *Fourier-Motzkin elimination method* (see [76, 149]), also described in Chapter 5. Therefore, we obtain a description of all the admissible $x(k+1)$, which does not depend upon specific $x(k)$ nor $d(k)$.

Notice that $X(k+1)$ is, in general, a set with a large (or infinite) number of elements, rather than a singleton. Moreover, it can be obtained by the intersection of two sets, namely $\tilde{X}(k+1)$ and $Y(k+1)$ (see Fig. 6.1), which are defined as follows:

$$\tilde{X}(k+1) := \{\tilde{x} : \tilde{x} = A(k)x + L_d(k)d + B(k)u(k), x \in X(k), |d| \leq 1\}, \tag{6.4}$$

$$Y(k) := \{x : y(k) = C(k)x + n, |n| \leq \bar{n}.\} \tag{6.5}$$

Therefore, we have that

$$X(k+1) = \tilde{X}(k+1) \cap Y(k+1).$$

Hence, (6.4) can be interpreted as a predictor, which estimates where the state of the system is going to take value in the next sampling time, while (6.5) can be used to update the predicted set-valued estimate of the state, based on the most recent observations. Figure 6.1 illustrates these steps.

The formulation in (6.3) can be easily extended, in case it is convenient to compute $X(k+1)$ not only based upon $X(k)$, but also upon $X(k-1)$, $\cdots$, $X(k-N)$. Indeed, $x(k+1) \in X(k+1)$ if and only if there exist $x(k+1)$, $\cdots$, $x(k-N+1)$, $y(k)$, and $d(k)$, $\cdots$, $d(k-N+1)$, such that,

$$
P_N(k) \begin{bmatrix} x(k+1) \\ x(k) \\ x(k-1) \\ \vdots \\ x(k-N+1) \\ d(k) \\ d(k-1) \\ \vdots \\ d(k-N+1) \end{bmatrix} \leq \begin{bmatrix} B(k)u(k) \\ -B(k)u(k) \\ \tilde{A}_1^k B(k\text{-}1)u(k\text{-}1) + B(k)u(k) \\ -\tilde{A}_1^k B(k\text{-}1)u(k\text{-}1) - B(k)u(k) \\ \vdots \\ \tilde{A}_{N\text{-}1}^k B(k\text{-}N)u(k\text{-}N) + \cdots + B(k)u(k) \\ -\tilde{A}_{N\text{-}1}^k B(k\text{-}N)u(k\text{-}N) - \cdots - B(k)u(k) \\ \mathbf{1} \\ \vdots \\ \mathbf{1} \\ \tilde{m}(k) \\ \vdots \\ m(k\text{-}N) \end{bmatrix} =: p_N(k) \tag{6.6}
$$

where

$$P_N(k) :=$$

$$\begin{bmatrix}
I & -\tilde{A}_0^k & \cdots & 0 & -L_d^k & 0 & \cdots & 0 \\
-I & \tilde{A}_0^k & \cdots & 0 & L_d^k & 0 & \cdots & 0 \\
I & 0 & \cdots & 0 & -L_d^k & -\tilde{A}_1^k L_d^{k-1} & \cdots & 0 \\
-I & 0 & \cdots & 0 & L_d^k & \tilde{A}_1^k L_d^{k-1} & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \cdots & \vdots \\
I & 0 & \cdots & -\tilde{A}_{N-1}^k & -L_d^k & \cdots & \cdots & -\tilde{A}_{N-2}^k L_d^{k-N+1} \\
-I & 0 & \cdots & \tilde{A}_{N-1}^k & L_d^k & \cdots & \cdots & \tilde{A}_{N-2}^k L_d^{k-N+1} \\
0 & \cdots & \cdots & 0 & I & 0 & \cdots & 0 \\
0 & \cdots & \cdots & 0 & -I & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \cdots & \cdots & 0 & 0 & \cdots & \cdots & I \\
0 & \cdots & \cdots & 0 & 0 & \cdots & \cdots & -I \\
\tilde{M}(k) & 0 & \cdots & 0 & 0 & \cdots & \cdots & 0 \\
0 & M(k-1) & \cdots & 0 & 0 & \cdots & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & \cdots & 0 & M(k-N) & 0 & \cdots & \cdots & 0
\end{bmatrix}$$

with

$$\tilde{A}_m^k = A(k)A(k-1)\cdots A(k-m).$$

6.3 Set-Valued Observers for Uncertain Plants

For plants with uncertainties, the set $X(k+1)$ is, in general, non-convex, even if $X(k)$ is convex. Thus, it cannot be represented by a linear inequality as in (6.2). For instance, consider a dynamic system described by the following equations:

$$\begin{cases} x(k+1) &= \left(\begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} + \Delta \begin{bmatrix} -2 & 0 \\ 4 & 0 \end{bmatrix} \right) x(k), \\ y(k) &= x(k), \end{cases}$$

with $\Delta \in [-1, 1]$, $x(0) \in X(0) := \{-1 \leq x_1 \leq 1, -1 \leq x_2 \leq 1\}$. Clearly, $X(0)$ is a convex set. Nevertheless, $X(1)$, depicted in Fig. 6.3, is a non-convex set.

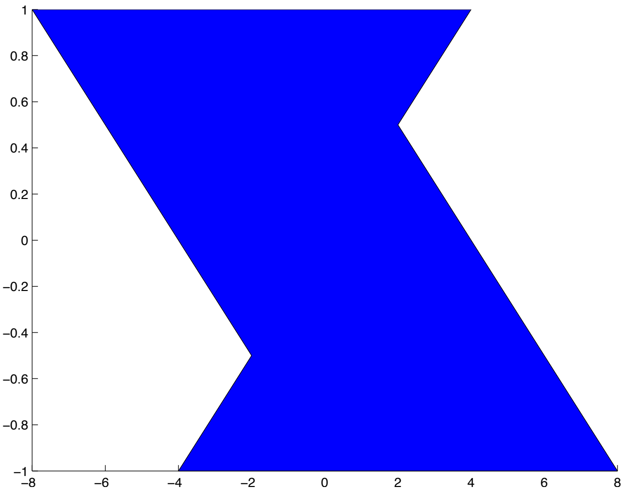


Figure 6.3: In the example, the set $X(1)$ is non-convex, despite the fact that $X(0)$ is convex.

Therefore, the set $X(k+1)$ cannot be represented, in general, by (6.2).

The formulation presented in the previous section is not able to cope with model uncertainty. We are particularly interested in explicitly taking into account parametric uncertainty in the dynamic models of the systems. This type of uncertainty arises naturally from the modeling of physical systems, such as flexible structures and vehicles moving through fluids, among others.

An implementable solution to the set-valued estimation of the state of an uncertain LPV system is presented in [75]. In the suggested approach, a set-valued state estimate is provided at each time, through the vertices of a polytope, $\mathcal{P}(k)$. However, it is not guaranteed that the *true* state, x_{true}, is contained in $\mathcal{P}(k)$, though the distance between x_{true} and $\mathcal{P}(k)$ is guaranteed to be bounded.

Implementable SVOs for LTV systems driven by exogenous disturbances were presented in [76, ShammaTu]. One of the main advantages of this solution is that it is *non-conservative.* In other words, this means that, given $X(k)$ as previously defined, the set-valued estimate of the state in the next sampling time, $X(k+1)$, contains only points that are feasible. Thus, if $x(k+1) \in X(k+1)$, then there exist

$d(k)$ and $x(k) \in X(k)$, such that (6.3) is satisfied. Moreover, the method guarantees that $X(k+1)$ contains all the states that are achievable at sampling time $k+1$.

Preliminary results of the extension of the work in [76, ShammaTu] to uncertain LPV plants were presented in [142]. In this section, we further extend these results, so that different types of model uncertainty can be handled.

6.3.1 Parametric Uncertainty in the Input Matrix

We start by considering uncertainty in the input matrix B, *i.e.*, we assume that the system can be described by

$$\begin{cases} x(k+1) &= A(k)x(k) + L(k)d(k) + \left(B_0(k) + \sum_{j=1}^{n_\Delta} \Delta_j(k) B_j(k) \right) u(k), \\ y(k) &= C(k)x(k) + N(k)n(k), \end{cases} \tag{6.7}$$

where $x(0) \in X(0)$, $x(k) \in \mathbb{R}^n$, $u(k) \in U \subseteq \mathbb{R}^{n_u}$, $d(k) \in W_d \subseteq \mathbb{R}^{n_d}$, $y(k) \in \mathbb{R}^{n_y}$, $n(k) \in W_n \subseteq \mathbb{R}^{n_n}$, and $\Delta(k) \in \mathbb{R}^{n_\Delta}$. As happened previously, the system is assumed observable and controllable from the exogenous disturbances and control inputs, for all admissible $\rho(\cdot)$. It is also assumed that

$$|\Delta_j(k)| \leq 1.$$

In this case, the uncertainty vector, $\Delta(k) = \begin{bmatrix} \Delta_1(k) & \cdots & \Delta_{n_\Delta}(k) \end{bmatrix}^{\mathrm{T}}$, represents uncertainty in the input of the plant. Define

$$F_j(k) := F_j(u(k)) := B_j(k)u(k), \tag{6.8}$$

for $j \in \{1, \cdots, n_\Delta\}$. Then, by substituting (6.8) in (6.7), we obtain the following equivalent description for the system

$$\begin{cases} x(k+1) &= A(k)x(k) + B_0(k)u(k) + \sum_{j=1}^{n_\Delta} F_j(k)\Delta_j(k) + L(k)d(k), \\ y(k) &= C(k)x(k) + N(k)n(k), \end{cases} \tag{6.9}$$

where $|\Delta_j(k)| \leq 1$. Therefore, each $\Delta_j(\cdot)$ can be seen as a bounded exogenous disturbance, acting upon the system. Hence, we recover the formulation in [76, ShammaTu], which means that the methodology described in the previous section can be used to obtain $X(k+1)$ based on $X(k)$.

6.3.2 Parametric Uncertainty in the Output Matrix

Consider a dynamic system, S, described by

$$S : \left\{ \begin{array}{rcl} x(k+1) & = & A(k)x(k) + B(k)u(k) + L(k)d(k), \\ y(k) & = & \left(C(k) + \sum_{j=1}^{n_\Delta} \Delta_j(k) C_j(k) \right) x(k) + N(k)n(k), \end{array} \right. \tag{6.10}$$

where $x(0) \in X(0)$, $x(k) \in \mathbb{R}^n$, $u(k) \in U \subseteq \mathbb{R}^{n_u}$, $d(k) \in W_d \subseteq \mathbb{R}^{n_d}$, $y(k) \in \mathbb{R}^{n_y}$, $n(k) \in W_n \subseteq \mathbb{R}^{n_n}$, and $\Delta \in \mathbb{R}^{n_\Delta}$. The system is assumed observable and controllable from the exogenous disturbances and control inputs, for all admissible $\rho(\cdot)$. It is also assumed that

$$|\Delta_j(k)| \leq 1.$$

In this case, the uncertainty vector, $\Delta(k)$, represents uncertainty in the output of the plant. Notice that S is equivalent to

$$S \equiv \left(\bar{S}_j + N(k)\bar{n}\right) + \sum_{j=1}^{n_\Delta} \left(\Delta_j \bar{S}_j + N(k)\bar{n}\right), \tag{6.11}$$

where, for $j \in \{1, \cdots, n_\Delta\}$,

$$\bar{S}_j : \left\{ \begin{array}{rcl} x_j(k+1) & = & A(k)x_j(k) + B(k)u(k) + L(k)d_i(k), \\ y_j(k) & = & C_j(k)x_j(k), \end{array} \right.$$

with $x_j(0) = x(0)$ for all $j \in \{0, \cdots, n_\Delta\}$, and $\bar{n}_i = \frac{n_i}{n_\Delta + 1}$. The block diagram of (6.11) is depicted in Fig. 6.4.

Since each $\bar{S}_j$, for $j \in \{0, \cdots, n_\Delta\}$, is a linear system, and each $\Delta_j(k)$, for $j \in \{1, \cdots, n_\Delta\}$ and $k \geq 0$, is an uncertain scalar, we obtain

$$S \equiv \left(\bar{S}_j + N(k)\bar{n}_i\right) + \sum_{j=1}^{n_\Delta} \left(\tilde{S}_j + N(k)\bar{n}\right), \tag{6.12}$$

where

$$\tilde{S}_j : \left\{ \begin{array}{rcl} x_j(k+1) & = & A(k)x_j(k) + B(k)\Delta_j(k)u(k) + L(k)\Delta_j(k)d(k), \\ y_j(k) & = & C_j(k)x_j(k). \end{array} \right.$$

Notice that (6.12) describes an LPV system with uncertain input. Nevertheless, the exogenous disturbances are now multiplied by the uncertainties $\Delta_j(k)$, and hence

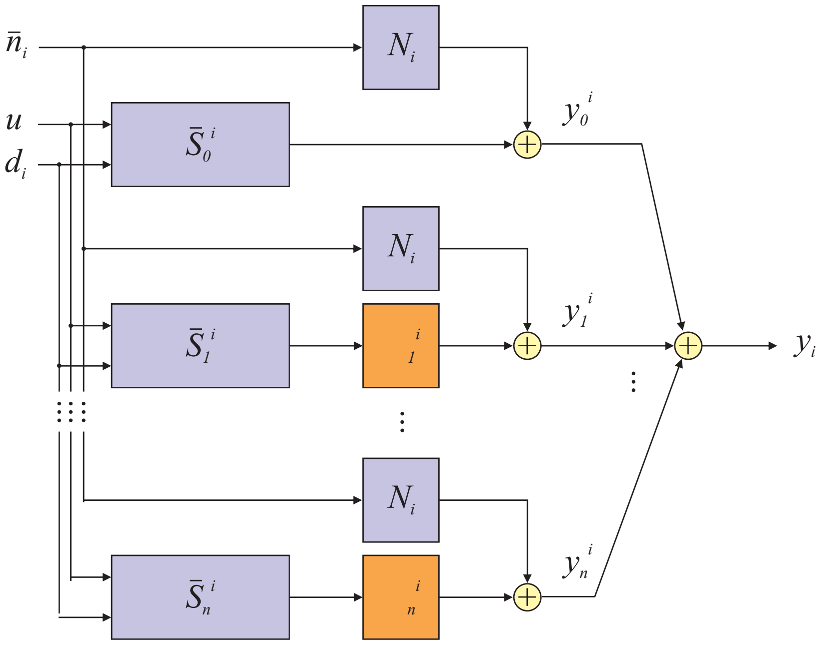


Figure 6.4: Block diagram of an LPV system with uncertainties in the output.

$\tilde{S}_j$ depends upon $\Delta_j(k)$ and $d(k)$ in a bilinear fashion. However, this can be avoided by introducing the following relaxation. Since $|\Delta_j(k)| \leq 1$, we have that

$$\tilde{d}_j(k) := \Delta_j(k)d(k) \Rightarrow \|\tilde{d}_j(k)\| \leq \|d(k)\|. \tag{6.13}$$

Thus, by substituting $\Delta_j(k)d(k)$ in (6.12) by $\tilde{d}_j(k)$ as in (6.13), we obtain a description of the system which is linear in the unknown variables, at the cost of some conservatism due to the implication in (6.13). Once again, the method in the previous section can be used to compute the set-valued estimate of the state.

6.3.3 Parametric Uncertainty in the Dynamics

We finally consider the problem of designing SVOs for LPV plants with uncertainty in the A matrix. Let S be described by

$$S : \begin{cases} x(k+1) & = & \left(A_0(k) + \displaystyle\sum_{j=1}^{n_\Delta} \Delta_j(k)A_j(k)\right) x(k) + B(k)u(k) + L(k)d(k), \\ y(k) & = & C(k)x(k) + N(k)n(k), \end{cases} \tag{6.14}$$

where $x(0) \in X(0)$, $x(k) \in \mathbb{R}^n$, $u(k) \in U \subseteq \mathbb{R}^{n_u}$, $d(k) \in W_d \subseteq \mathbb{R}^{n_d}$, $y(k) \in \mathbb{R}^{n_y}$, $n(k) \in W_n \subseteq \mathbb{R}^{n_n}$, and $\Delta(k) \in \mathbb{R}^{n_\Delta}$. The system is assumed observable and controllable from the exogenous disturbances and control inputs, for all admissible $\rho(\cdot)$. Moreover, we assume that

$$|\Delta_j(k)| \leq 1.$$

The uncertainty vector, $\Delta(k)$, represents uncertainty in the dynamics of the plant, and can appear, for instance, in the modeling of several types of physical systems.

Uncertainty Matrices with Rank 1

For the sake of simplicity, let us consider that $n_\Delta = 1$, although the results presented in the sequel can be readily extended to the case $n_\Delta > 1$. Further assume that $\operatorname{rank}(A_1(k)) = 1$ for all $k \geq 0$. Thus, there exist vectors $e_1(k)$ and $f_1(k)$, for each $k \geq 0$, such that

$$A_1(k) = e_1(k)\left(f_1(k)\right)^{\mathrm{T}}.$$

Moreover, define

$$g_1(k) := \left(f_1(k)\right)^{\mathrm{T}} x(k)\Delta(k).$$

Then, system S in (6.14), for $n_\Delta = 1$, can be rewritten as

$$S : \begin{cases} x(k+1) &= A_0(k)x_i(k) + e_1(k)g_1(k) + B(k)u(k) + L(k)d(k), \\ y(k) &= C(k)x(k) + N(k)n(k), \end{cases} \tag{6.15}$$

with the additional constraint

$$|g_1(k)| \leq \left|\left(f_1(k)\right)^{\mathrm{T}} x(k)\right|. \tag{6.16}$$

Notice that (6.15), for arbitrary $g_1(k)$, provides a description of $y(k)$ which is linear in the variables $x(k)$, $g_1(k)$, $u(k)$, $d(k)$ and $n(k)$. Nevertheless, the constraint in (6.16) is nonlinear regarding $x(k)$. However, for given k, the aforementioned constraint in (6.16) can be rewritten as

$$\begin{cases} -\left(f_1(k)\right)^{\mathrm{T}} x(k) \leq g_1(k) \leq \left(f_1(k)\right)^{\mathrm{T}} x(k) \\ \text{or} \\ \left(f_1(k)\right)^{\mathrm{T}} x(k) \leq g_1(k) \leq -\left(f_1(k)\right)^{\mathrm{T}} x(k). \end{cases} \tag{6.17}$$

Therefore, if, at time k, the set of possible states $X(k)$ is a convex polytope, it may happen that at iteration $k+1$ the set $X(k+1)$ will be the union of two convex polytopes. Hence, the description of $X(k+j)$, for $j > 0$, may require an arbitrarily large number of unions of convex polytopes. One way of dealing with this issue is to convexify the estimated set of possible states, by computing the smallest convex set that contains all the possible states. The shortcoming with this approach is that states which are not compatible with (6.1) may be included, thus adding conservatism to the solution.

General Case

If $\operatorname{rank}(A_1(k)) > 1$, then $A_1(k)$ can be written as the sum of several matrices with unitary ranks, as follows

$$A_1(k) = e_{1,1}(k)\,(f_{1,1}(k))^{\mathrm{T}} + \cdots + e_{1,m}(k)\,(f_{1,m}(k))^{\mathrm{T}},$$

for some integer $m \geq 0$. Thereafter, define

$$g_j(k) := (f_{1,j}(k))^{\mathrm{T}}\, x(k)\Delta(k),$$

for each $j \in \{1, \cdots, m\}$. Then, each $g_j(k)$ can be treated as an independent uncertainty, and the previously described method can be applied. However, this approach can add conservatism to the solution, since we assume no relation between each $g_j(k)$.

Convex Approach

A different method to handle uncertainty in the A matrix was presented in [156], and is going to be described herein for the sake of completeness. The proposed solution is to overbound set $X(k+1)$ by a convex one, denoted by $\hat{X}(k+1)$, which is going to be described as follows.

Let $v_i, i = 1, \ldots, 2^{(Nn_\Delta)}$, for some positive scalar N, denote a vertex of the hyper-cube

$$\mathcal{C} := \left\{\delta \in \mathbb{R}^{Nn_\Delta} : |\delta| \leq 1\right\}, \tag{6.18}$$

where $v_i = v_j \Leftrightarrow i = j$. Then, we denote by $\hat{X}_{v_i}(k+1)$ the set of points $x(k+1)$ that satisfy (6.14) with $[\Delta(k)^{\mathrm{T}}, \cdots, \Delta(k-N+1)^{\mathrm{T}}]^{\mathrm{T}} = v_i$ and with $x(k) \in \hat{X}(k)$, $\cdots$, $x(k-N+1) \in \hat{X}(k-N+1)$. Further define

$$\hat{X}(k+1) := \mathrm{co}\left\{\hat{X}_{v_1}(k+1), \cdots, \hat{X}_{v_{2^{(Nn_\Delta)}}}(k+1)\right\},$$

where $\mathrm{co}\{p_1, \ldots, p_m\}$ is the smallest convex set containing the points $p_1, \ldots, p_m$, also known as the convex hull of $p_1, \ldots, p_m$.

Since, as previously mentioned, $X(k+1)$ is, in general, non-convex even if $X(k)$ is convex, we are going to use $\hat{X}(k+1)$ to overbound the set $X(k+1)$. An illustration for the case $n_\Delta = 1$, $N = 1$, is depicted in Fig. 6.5.

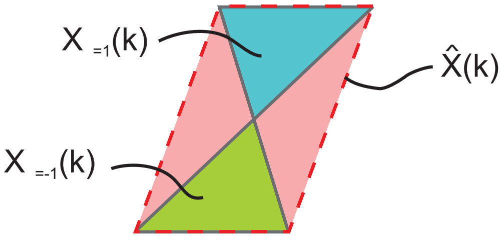


Figure 6.5: Convex hull, $\hat{X}(k)$, of the sets generated by the solutions to (6.1) with $n_\Delta = 1$, $N = 1$ and for $\Delta = 1$ and $\Delta = -1$.

The set $\hat{X}(k+1)$ contains $X(k+1)$, as demonstrated next.

Proposition 6.1. *Consider a system described by* (6.14) *and assume that* $X(0) \subseteq \hat{X}(0)$. *Then* $X(k) \subseteq \hat{X}(k)$ *for all* $k \in \{0, 1, 2, \cdots\}$.

Proof. Denote by $X_{v_i}(k+1)$ the set of points $x(k+1)$ that satisfy (6.14) with $[\Delta(k)^{\mathrm{T}}, \cdots, \Delta(k-N+1)^{\mathrm{T}}]^{\mathrm{T}} = v_i$ and with $x(k) \in X(k)$, $\cdots$, $x(k-N+1) \in X(k-N+1)$. Further define

$$X^*(k+1) := \mathrm{co}\left\{X_{v_1}(k+1), X_{v_2}(k+1), \cdots, X_{v_{2^{(Nn_\Delta)}}}(k+1)\right\}.$$

Then, it is clear that $X^*(k+1) \subseteq \hat{X}(k+1)$ if $\hat{X}(j) \subseteq X(j)$, for $0 \leq j \leq k$. By induction, it is straightforward to conclude that $X^*(i) \subseteq \hat{X}(i)$ for all $i \in \{0, 1, 2, \cdots\}$, since it was assumed that $X(0) \in \hat{X}(0)$.

Hence, we only have to prove that $X(k) \subseteq X^*(k)$. To see this, note that the disturbances and control input in (6.14) do not add any non-convexities to the set-valued estimate of the state at each sampling time. Therefore, without loss of generality, assume that $d(\cdot) \equiv 0$ and $u(\cdot) \equiv 0$ in (6.14). Let $x^\star \in X(k)$ be obtained by using a vector $\Delta \in \mathcal{C}$ in (6.18). Thus, it can be defined by

$$x^\star := \left[\prod_{i=1}^{N} \big(A_o(k-i) + \alpha_i A_1(k-i) - (1-\alpha_i) A_1(k-i) \big) \right] x,$$

where $x \in X(k-N)$,

$$0 \le \alpha_i \le 1,$$

for $i \in \{1, \cdots, N\}$, and assuming a single uncertainty, $|\Delta(i)| \le 1$, $i \ge 0$, for the sake of simplicity. (Recall that, in this scenario, $\Delta = \begin{bmatrix} \Delta(k-1) & \cdots & \Delta(k-N) \end{bmatrix}^{\mathrm{T}}$.) We now show that $x^\star$ can be obtained as a convex combination of points satisfying (6.14), with $d(\cdot) \equiv 0$, $u(\cdot) \equiv 0$, $x(k-N) \in X(k-N)$, and for Δ in the vertices of $\mathcal{C}$. Define

$$\begin{cases} A^+(k) &=& A(k) + A_1(k), \\ A^-(k) &=& A(k) - A_1(k), \end{cases}$$

and let

$$\begin{cases} x_1^1 &=& A^+(k-1)A^+(k-2)\cdots A^+(k-N)x, \\ x_2^1 &=& A^-(k-1)A^+(k-2)\cdots A^+(k-N)x, \\ x_3^1 &=& A^+(k-1)A^-(k-2)\cdots A^+(k-N)x, \\ x_4^1 &=& A^-(k-1)A^-(k-2)\cdots A^+(k-N)x, \\ & \vdots & \\ x_{2^N-1}^1 &=& A^+(k-1)A^-(k-2)\cdots A^-(k-N)x, \\ x_{2^N}^1 &=& A^-(k-1)A^-(k-2)\cdots A^-(k-N)x, \end{cases}$$

where $x \in X(k-N)$. Now let

$$\begin{cases} x_1^2(\gamma^1) &=& \Big[\Big(A_o(k-1) + \gamma^1 A_1(k-1) - (1-\gamma^1)A_1(k-1)\Big)A^+(k-2)\cdots A^+(k-N)\Big]x, \\ x_2^2(\gamma^1) &=& \Big[\Big(A_o(k-1) + \gamma^1 A_1(k-1) - (1-\gamma^1)A_1(k-1)\Big)A^-(k-2)\cdots A^+(k-N)\Big]x, \\ & \vdots & \\ x_{2^{N-1}}^2(\gamma^1) &=& \Big[\Big(A_o(k-1) + \gamma^1 A_1(k-1) - (1-\gamma^1)A_1(k-1)\Big)A^-(k-2)\cdots A^-(k-N)\Big]x. \end{cases}$$

Proceeding with this method, we obtain

$$x_1^{N+1}(\gamma^1, \cdots, \gamma^N) = \left[\prod_{i=1}^{N} \big(A_o(k-i) + \gamma^i A_1(k-i) - (1-\gamma^i) A_1(k-i) \big) \right] x.$$

Hence, we get

$$x_1^{N+1}(\gamma^1, \cdots, \gamma^N) = x^\star,$$

for $\gamma^i = \alpha_i$, for all $i \in \{1, \cdots, N\}$. A similar procedure can be used for the case where more than one uncertainty is considered, thus concluding the proof. □

Although this approach adds some conservatism to the solution, it possesses the valuable property summarized in Proposition 6.2.

Proposition 6.2. *Suppose that a system described by* (6.14) *with* $x(0) = X(0)$ *and* $u(k) = 0, \forall k$*, verifies, for sufficiently large* N^*,

$$\gamma_N := \max_{\substack{\Delta(k), \cdots, \Delta(k+N) \\ |\Delta(m)| \leq 1, \forall_m \\ k \geq 0}} \left\| \Pi_{j=k}^{k+N} \mathcal{A}(j) \right\| < 1,$$

for all $N \geq N^*$*, and where*

$$\mathcal{A}(j) := \left[A(j) + \sum_{i=1}^{n_\Delta} A_i(j) \Delta_i(j) \right].$$

Then, $\hat{X}(k)$ *cannot grow without bound.*

Proof. Consider the smallest hyper-cubes, denoted by $\Psi(1), \Psi(2), \cdots, \Psi(m)$, that contain the sets $\hat{X}(1)$, $\hat{X}(2)$, $\cdots$, $\hat{X}(m)$, respectively. Let $N \geq N^*$. Then, an SVO can be synthesized to generate the sets $\Psi(1)$, $\Psi(2)$, $\cdots$, $\Psi(m)$, using the following inequality:

$$|x(k+N)| \leq \gamma_N |x(k)| + \delta_N, \tag{6.19}$$

where

$$\delta_N = \max_{d(k), \cdots, d(k+N-1)} |A^{N-1} L d(k) + \cdots + L d(k+N-1)|.$$

Notice that, to prove this proposition, it suffices to show that the sequence $\Psi(1)$, $\Psi(2)$, $\cdots$, $\Psi(m)$ does not grow without bound, since it contains $\hat{X}(1), \hat{X}(2), \cdots, \hat{X}(m)$. Given that $\gamma_N < 1$ by assumption and that $|\delta_N| < \infty$ since $|d| < \infty$, the sets defined by (6.19) cannot grow without bound, which concludes the proof. □

Notice that, in order to guarantee that $\hat{X}$ does not grow without bound, an SVO should use the N most recent estimates. In other words, the estimation of $\hat{X}(k+N)$ should take into account the fact that $x(k) \in \hat{X}(k), x(k+1) \in \hat{X}(k+1), \cdots, x(k+N-1) \in \hat{X}(k+N-1)$.

In summary, this subsection proposed two methods, with different properties, to handle uncertainty in the A matrix. In the first case, no conservatism is added whenever the ranks of the uncertainty matrices, $A_j(k)$, are unitary. Nevertheless, it may require an arbitrarily large number of unions of convex polytopes. The solution proposed to overcome this problem, at the cost of some conservatism, was to compute, at each time k, the convex hull of the admissible sets, and use it as an estimate of $X(k)$. The second method proposed is, in general, less computationally demanding, but can also be conservative.

6.4 Computational and Numerical Issues

This section presents a discussion on some of the most important computational and numerical shortcomings of the SVOs. These limitations constraint the maximum number of states of the dynamic model of an SVO, and must be taken into account during design. It is also provided, in this section, a few tools to address some of the problems related to the implementation of the SVOs.

6.4.1 Fourier-Motzkin Elimination Method

The first issue in the computations performed by the SVOs is related to the implementation of the so-called Fourier-Motzkin elimination method, described in [149], that projects polyhedral convex sets on to subspaces, as illustrated in Fig. 6.2.

The Fourier-Motzkin algorithm leads to a set of linear inequalities, where some of them might be linearly dependent. This may be problematic, since the size of $M(k)$ and $m(k)$ (see (6.3)) could be increasing very fast with time. To overcome this problem, one has to eliminate the linearly dependent constraints. This can be done by solving several small linear programming problems at each sampling time, making the practical implementation of this type of observers somewhat computationally complex.

6.4.2 Union of Convex Polytopes

If the dynamic model of the plant in (6.1) is uncertain, it may be convenient to implement the union of several polytopes, as noted in Section 6.3.3. In one- or two-dimensional spaces, this can be easily implemented. For higher dimensional spaces, one can resort, for instance, to the algorithm in [157]. Nevertheless, as happened with the Fourier-Motzkin elimination method, the computational burden that arises from these calculations may be significant for plants with a large number of states. Hence, in many applications, it may be more convenient to use the convex hull of the set-valued estimate of the state, though this may add some conservatism.

6.4.3 Numerical Approximation of Convex Polytopes

Another possible shortcoming of the SVOs is related to the numerical approximations used during the computation of the set-valued estimates. In other words, since we do not have infinite precision in the computations that have to be carried out every sampling time to obtain the set-valued estimate $\hat{X}(k)$, the actual set where the state can take value, $X(k)$, need not be entirely contained inside $\hat{X}(k)$ – see Fig. 6.6. Therefore, it may happen that the true state does not belong to $\hat{X}(k)$, and hence we may end up by discarding the region which the parameter actually belongs to.

Thus, a very simple solution is to "robustify" the algorithm by slightly enlarging the set $\hat{X}(k)$, as illustrated in Fig. 6.6. As long as the maximum error in the computation of the set $X(k)$ is known, we have, for every time, k, a vector $\epsilon^*(k)$ such that $X(k) \subseteq \mathrm{Set}\,(M(k), m(k) + \epsilon^*(k))\,.$

Moreover, it may happen that, from time-step to time-step, the number of faces of the polytope containing the set-valued estimate of the state of the system increases exponentially. Hence, it is useful to overbound, in such circumstances, that polytope by another one, with a constrained number of faces.

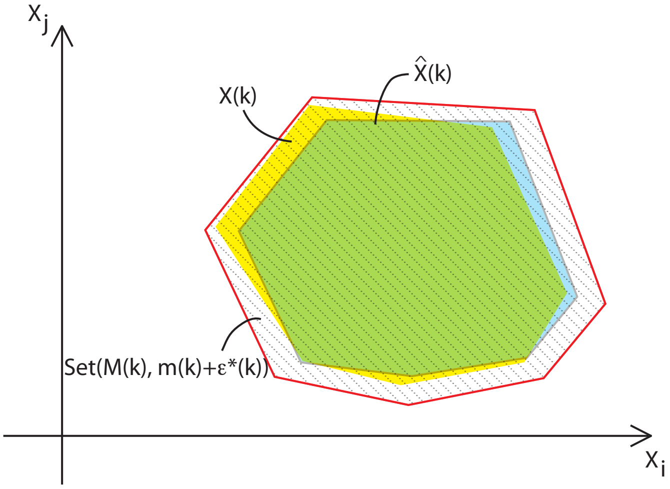


Figure 6.6: Overbound of set $\hat{X}(k)$ to include $X(k)$.

Nonetheless, as stressed in [142], using an overbound to guarantee that we do not discard valid states of the plant also has its shortcomings. Besides adding conservatism to the solution, it may be responsible for the unbounded increase with time of the area of the polytope of the set-valued estimate.

Remark 6.1: As stressed in the Introduction of this chapter, one of the first algorithms developed to compute (ellipsoidal) set-valued estimates of the state of a system was presented in [152] and [153]. Using ellipsoids to describe the set-valued estimates of the state is an alternative method to the one presented in this thesis, with the advantage of having less computationally demanding calculations. However, unlike the convex polytope-based approach presented in this chapter, the ellipsoid-based approach does not guarantee convergence of the set of state estimates, even if the system at hand is stable. Therefore, the SVOs described in this thesis should only be seen as an alternative to the ellipsoid-based approach. ◇

The remainder of this subsection is, therefore, devoted to the derivation of sufficient conditions that guarantee that, if $X(k)$ is bounded, so does the estimated set, $\hat{X}(k)$.

Consider a time-invariant plant described by (6.1) with no model uncertainty and with

$$\begin{cases} A(\rho(k)) & = & A \\ B(\rho(k)) & = & B \\ L(\rho(k)) & = & L \\ C(\rho(k)) & = & C \\ N(\rho(k)) & = & N \end{cases}$$

for all $k \geq 0$. Furthermore, let $X(0)$ be bounded. Throughout this subsection, we suppose that the eigenvalues of A belong to the unit circle. In that case, we can guarantee that $X(k)$ is bounded. However, *we are not constraining the applicability of the SVOs*. As shown in the previous section, the SVOs can be used to compute the set-valued estimate of the states of plant with a state-space realization as in (6.1), and, therefore, unstable plants can be handled.

Let $\Psi(k)$ denote the smallest hyper-cube centered at the origin, that contains the set $X(k)$, as depicted in Fig. 6.7. Define $\epsilon(k)$ as the maximum distance between a face of $\Psi(k)$ and the corresponding face of the estimate $\hat{\Psi}(k)$, as depicted in Fig. 6.7. Next, we try to derive sufficient conditions to guarantee that $\hat{\Psi}(k)$ is bounded. It should be noticed that $\hat{\Psi}(k)$ can be interpreted as a rough approximation of $\hat{X}(k)$, in the sense that $\hat{X}(k) \subseteq \hat{\Psi}(k)$, which means that if $\hat{\Psi}(k)$ is bounded, so does $\hat{X}(k)$.

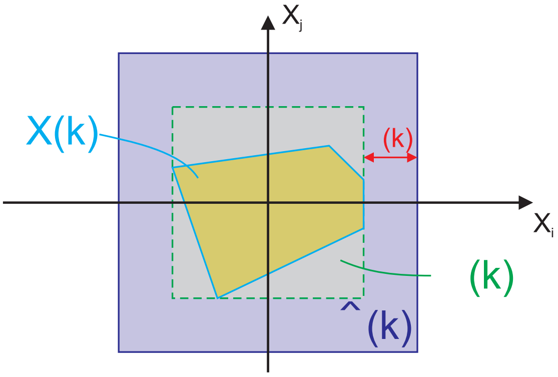


Figure 6.7: Bounding set $\Psi(k)$, corresponding estimate $\hat{\Psi}(k)$, and maximum numerical error $\epsilon(k)$.

Proposition 6.3. *Consider an* asymptotically stable *plant described by* (6.1) *with the aforementioned constraints. Suppose that the maximum numerical error (pre-*

viously defined) at every sampling time is $\epsilon(k)$, with $\epsilon(k) \leq \epsilon^*|x_i(k)|$, for some $0 \leq \epsilon^* < \infty$ and for every $x(k) \in X(k)$. Further suppose that $\epsilon^* < 1 - \rho(A)$, where $\rho(A)$ denotes the spectral radius of A. Then, $\hat{\Psi}(k)$ is bounded.

Proof. For given N, define

$$\delta := \sup_{d(k),\cdots,d(k+N-1)} |A^{N-1}Ld(k) + \cdots + Ld(k+N-1)|.$$

Notice that δ is bounded, since $d(\cdot)$ is also bounded. Then,

$$|x(k+N)| \leq \|A^N\||x(k)| + \delta.$$

Since the eigenvalues of A are inside the unit circle, we can find $\gamma > 0$ and $N > 0$ such that $\|A^N\| \leq 1 - \gamma$. Hence

$$|x(k+N)| \leq (1-\gamma)|x(k)| + \delta = \left[1 - \left(\frac{j-1}{j}\right)\gamma\right]|x(k)| - \frac{\gamma}{j}|x(k)| + \delta,$$

for any $j \neq 0$. Thus, for any $j > 0$, and sufficiently large $|x(k)|$, we have

$$-\frac{\gamma}{j}|x(k)| + \delta \leq 0,$$

which leads to

$$|x(k+N)| \leq \left[1 - \left(\frac{j-1}{j}\right)\gamma\right]|x(k)|.$$

However, to overcome the aforementioned numerical issues, we overbound this set by

$$|x(k+N)| \leq \left[1 - \left(\frac{j-1}{j}\right)\gamma + \epsilon^*\right]|x(k)|.$$

If $\epsilon^* < \frac{j-1}{j}\gamma$, then $|x(k+N)| < |x(k)|$. Taking the limit as j tends to infinity leads to the desired result. □

Remark 6.2: In an intuitive manner, the above result implies that systems driving their states to zero rapidly can have larger overbounds for the sets $X(\cdot)$ than slower systems. ◇

Corollary 6.1. *Consider a* stable *plant described by* (6.1) *with the aforementioned constraints. Suppose that the maximum numerical error at every sampling time is ϵ, with $0 \leq \epsilon < \infty$. Then, $\hat{\Psi}(k)$ is bounded.*

Proof. Using a similar approach to that in the previous proposition, we get

$$|x(k+N)| \leq \left[1 - \left(\frac{j-1}{j}\right)\gamma\right] |x(k)| + |\epsilon|,$$

for any $j > 0$ and sufficiently large $|x(k)|$ and N. Thus, also for sufficiently large $|x(k)|$,

$$\left[1 - \left(\frac{j-1}{j}\right)\gamma\right] |x(k)| + |\epsilon| \leq (1-\rho)\,|x(k)|,$$

with $0 < \rho < 1$, which concludes the proof. □

Remark 6.3: We stress that this type of errors can be modeled as an exogenous disturbance. This fact is in agreement with Corollary 6.1, since the only requirement is for the system dynamics matrix to have a spectral radius smaller than one. ◇

These shortcomings of the SVOs do not jeopardize the implementability of the algorithms. In fact, although some of the constraints may seem very stringent from a practical point of view, they may not be relevant when used to *discard* dynamic models in a Multiple-Model Adaptive Control (MMAC) architecture, as shown in Chapter 7.

6.4.4 Bounding Ellipsoids

One of the most widespread approaches to the design of set-valued observers was introduced in [152] and [153]. Instead of considering convex polytopes to overbound the set-valued estimates of the state, the authors in [152] and [153] resort to the use of bounding ellipsoids. This is indeed an interesting an alternative method to the SVOs developed in this thesis, due to its reduced computational requirements.

Nevertheless, the bounding ellipsoids also have their shortcomings. Let $X(k)$ denote the set where the state of the system can actually take values at time k, and let $\hat{X}(k)$ be the corresponding ellipsoid overbound. As stressed in [152], "*it cannot be concluded that the boundary of $\hat{X}(k)$ will necessarily touch $X(k)$ at any point*", which means that the conservatism added by the bounding ellipsoid can be arbitrarily large.

As an example, Fig. 6.8 depicts the time-evolution of the set $X(k)$, for a given dynamic system. In this case, the volume of the set where the states of the plant can take value is decreased every iteration.

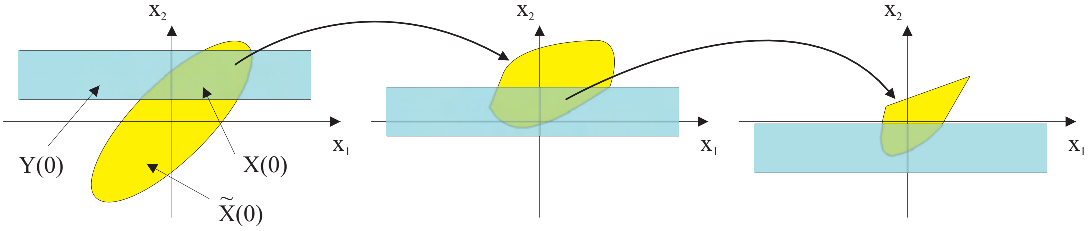


Figure 6.8: Time-evolution of set $X(k)$.

However, if we use bounding ellipsoids, the results obtained are illustrated in Fig. 6.9. In this example, we assume that the states added due to the conservatism of the bounding ellipsoids are responsible for the increase, in terms of volume, of the set-valued estimate of the state. Hence, it may happen that the volume of the bounding ellipsoid grows arbitrarily, although the actual set to which the state of the system belongs is bounded.

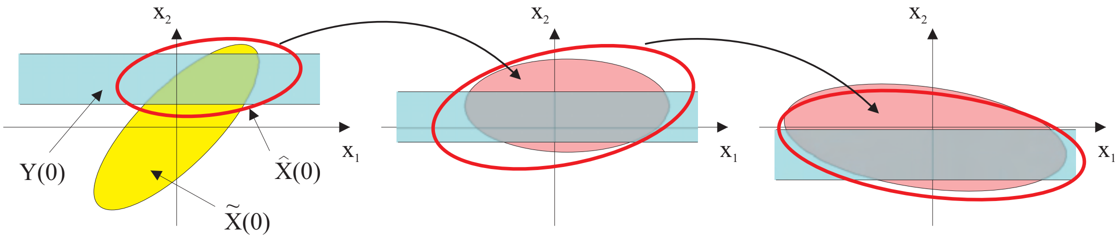


Figure 6.9: Time-evolution of the bounding ellipsoid, $\hat{X}(k)$, to the set $X(k)$.

As further stressed in Chapter 9, an insightful comparison between the bounding ellipsoids and the SVOs developed in this thesis is a possible direction of future research.

6.4.5 Discretization Issues

This subsection is devoted to an important topic when dealing with models of realistic (physical) systems. Several plants, like the Mass-Spring-Dashpot (MSD) plant described in Chapter 2, are naturally modeled in continuous-time. Nonetheless, the aforementioned SVOs can only be used with discrete-time dynamic models. The classical methods for discretizing LTI systems cannot be readily applied to uncertain models. Therefore, we start this subsection by deriving discrete-time uncertain models for continuous-time uncertain plants.

Another important discretization issue is related to the fact that the SVOs typically entail heavy computations, when compared to the dynamics of the plant. In other words, it is required, in general, different sampling frequencies for the control and decision subsystems of the multiple-model adaptive control methodology presented in the following chapter. Hence, the second part of this subsection tackles the problem of having different sampling rates for the same plant – the interested reader is also referred to [158].

Discretization of Continuous-Time Uncertain Models

Consider a continuous-time LTI stable plant described by

$$\left\{\begin{array}{rcl} \dot{x}(t) & = & Ax(t) + Bu(t) + Ld(t), \\ y(t) & = & Cx(t) + n(t). \end{array}\right. \tag{6.20}$$

Let T be the desired sampling period and assume that $u(\cdot)$ and, for the sake of simplicity, $d(\cdot)$, are constant during each sampling interval. Then, for $t = kT$, we can rewrite (6.20) as

$$\left\{\begin{array}{rcl} x((k+1)T) & = & A^d x(kT) + B^d u(kT) + L^d d(kT), \\ y(kT) & = & Cx(kT) + n(kT), \end{array}\right.$$

where $A^d := e^{AT}$, $B^d := \int_0^T e^{A\tau} B d\tau$. In the sequel, we use k instead of kT, for the sake of notational simplicity.

Next, we suppose that A is uncertain, with a single uncertainty. Redefine $A := \bar{A} + A_1\Delta$, where $|\Delta| \leq 1$. Then, for small T,

$$A^d = e^{AT} \approx e^{\bar{A}T}\left(I + A_1 T\Delta + \frac{(A_1 T\Delta)^2}{2} + \cdots\right). \tag{6.21}$$

We recall that if A_1 is nilpotent, then (6.21) becomes much simpler. In particular, suppose that $A_1^2 = 0$. Then

$$A^d = e^{AT} \approx e^{\bar{A}T}\left(I + A_1 T\Delta\right) = \bar{A}^d + A_1^d \Delta,$$

where $\bar{A}^d := e^{\bar{A}T}$, $A_1^d := e^{\bar{A}T} A_1 T$. In case the approximation in (6.21) does not hold, numerical methods can be employed to approximate A^d.

Now let us consider the discretized B matrix. The calculations for the L matrix are similar, and hence are not presented here. We have that

$$B^d = \int_0^T e^{A\tau} d\tau B = \int_0^T e^{\bar{A}\tau} e^{A_1 \Delta \tau} d\tau B. \tag{6.22}$$

As seen in (6.22), B^d is also uncertain. Describing B^d as a function of Δ can lead to cumbersome calculations. However, in some practical problems, it is possible to bound the integral in (6.22) resorting to numerical methods. In such circumstances, we can rewrite B^d as $B^d = \tilde{B}^d + d_b$, where d_b represents a fictitious disturbance.

Sampling of Discrete-Time Models

As mentioned before, due to the computational requirements associated with the set-valued observers (in particular, requiring the on-line elimination of several linear inequalities), different sampling rates should be used for the control and for the estimation part of the multiple-model adaptive control algorithm presented in the next chapter. Therefore, let T_s be the sampling period for the SVOs, and $T_c << T_s$ be the sampling period for the controllers. Assume that $T_s = mT_c$, where m is a positive integer. We further define

$$A_s = e^{AT_s},$$

and

$$A_c = e^{AT_c}.$$

We consider that the model used for control is described by (omitting the disturbances)

$$\begin{cases} x(k+1) &= A_c x(k) + B_c u(k), \\ y(k) &= Cx(k) + n(k). \end{cases}$$

Hence, $x(k+m) = A_s x(k) + \bar{B}\bar{u}(k)$, where

$$\begin{aligned} \bar{B} &= \begin{bmatrix} A_c^{m-1}B_c & A_c^{m-2}B_c & \cdots & A_cB_c & B_c \end{bmatrix}, \\ \bar{u}(k) &= \begin{bmatrix} u(k) & u(k-1) & \cdots & u(k+m-1) \end{bmatrix}^T. \end{aligned}$$

Thus, by augmenting the control input, $u(\cdot)$, and using $\bar{n}(k) = n(mk)$, the model used by the SVOs, that is, the model with sampling period T_s, can be described by

$$\begin{cases} \bar{x}(k+1) &= A_s\bar{x}(k) + \bar{B}\bar{u}(k), \\ \bar{y}(k) &= C\bar{x}(k) + \bar{n}(k). \end{cases}$$

Hence, the SVOs can run at a much lower sampling rate than that of the control system, thus avoiding the deterioration of the performance of the closed-loop system.

6.5 Simulation

In this section, the applicability of the set-valued observers introduced in this chapter is illustrated in simulation. The example selected is the same as in [76, ShammaTu], although in the following simulations we also consider model uncertainty.

Let

$$\begin{cases} x(k+1) &= \left(\begin{bmatrix} 0.7 & 0.7 \\ -0.7 & 0.7 \end{bmatrix} + c\Delta \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} \right) x(k) + \begin{bmatrix} 1 \\ 1 \end{bmatrix} d(k) \\ y(k) &= \begin{bmatrix} 1 & 1 \end{bmatrix} x(k) + n(k), \end{cases} \tag{6.23}$$

where c is a known constant and Δ, with $|\Delta| \leq 1$, is a surrogate for model uncertainty. Moreover, the exogenous disturbance and measurement noise are known to be bounded by $|d| \leq 1$ and $|n| \leq 1$, respectively, and the initial state is assumed to be contained inside the set defined as

$$x(0) \in \left\{ x \in \mathbb{R}^2 : |x| \leq 1 \right\} = \mathrm{Set}\left(\begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \right)$$

Our goal is to design an observer that provides set-valued estimates for the state of the plant at each sampling time, based on the aforementioned information only.

The horizon of the simulation is $k = 0, 1, 2, 3$, and the disturbance and noise histories used are

$$\begin{aligned} d(0,1,2,3) &= (1,-1,-1,1), \\ n(0,1,2,3) &= (1,-1,1,-1). \end{aligned}$$

The true initial condition for the simulations was defined as $x(0) = 0$.

The sets of admissible states, for $c = 0$ in (6.23), at times $k = 2, 3$, are depicted in Fig. 6.10 and Fig. 6.11, respectively. In this case, there is no model uncertainty, which means that these sets are not conservative, *i.e.*, for each state, $x^\star$, inside these sets, there exists a sequence of disturbances and measurement noise that drives the true state of the plant to $x^\star$.

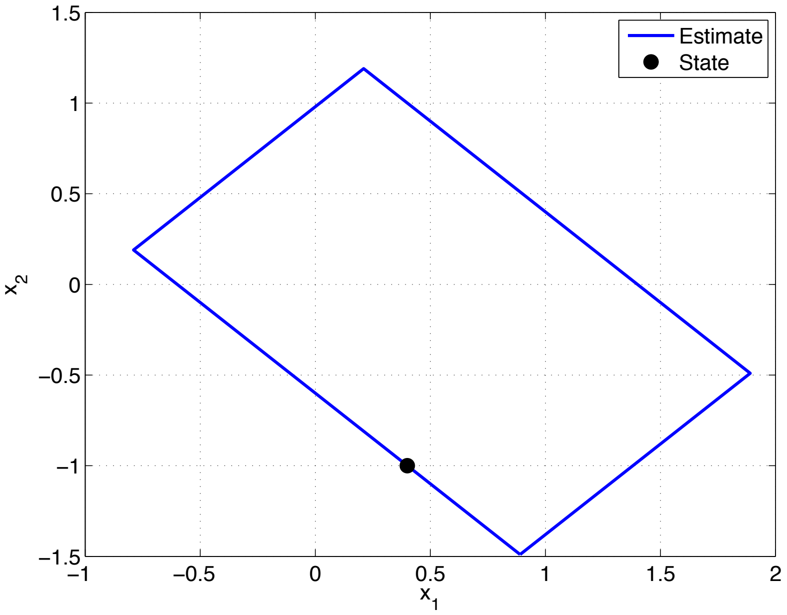


Figure 6.10: True state of the system and estimated set of admissible states, at time $k = 2$ and for $c = 0$ in (6.23).

Now let us consider that $c \neq 0$, *i.e.*, the model of the plant is uncertain. For the simulations, we let $\Delta = 0$ in (6.23). The sets of admissible states, for several values of c, at time $k = 3$, are depicted in Fig. 6.12. The SVOs were designed using the *Convex Approach* in Section 6.3. As expected, the size of these sets increases with c. Moreover, if we denote by $X_{\bar{c}}(k)$ the set of admissible states at time k and with

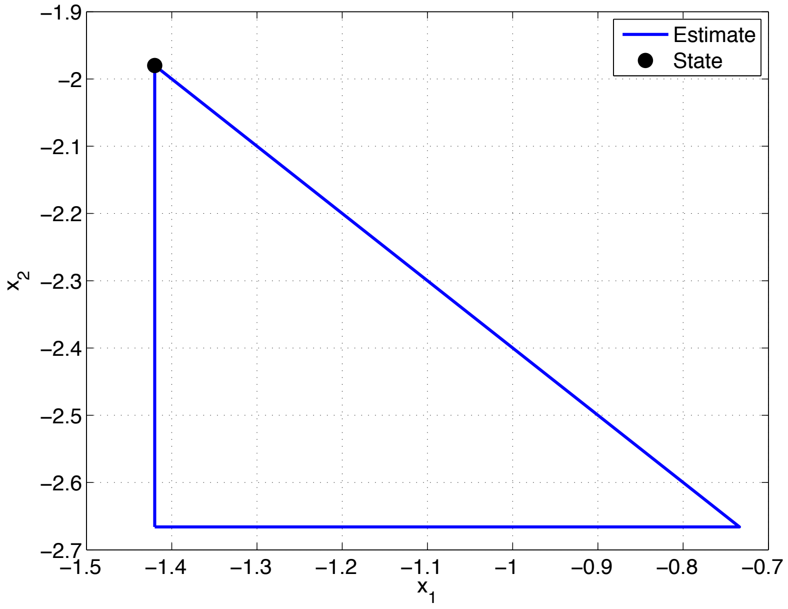


Figure 6.11: True state of the system and estimated set of admissible states, at time $k = 3$ and for $c = 0$ in (6.23).

$c = \bar{c}$, then we conclude that

$$X_0(k) \subseteq X_{0.01}(k) \subseteq X_{0.1}(k) \subseteq X_{0.2}(k).$$

6.6 Conclusions

This chapter was devoted to the development of Set-Valued Observers (SVOs), that are able to estimate the set of admissible states of dynamic Linear-Parameter Varying (LPV) uncertain systems, and that can ultimately be used to invalidate models of a plant.

Notwithstanding the rich literature available on the subject of SVOs design, not much attention was devoted to the synthesis of SVOs which provide, at each sampling time, the set that contains *all* the possible states of the plant, regardless of the sequence of exogenous disturbances and sensor noise, and assuming an uncertain dynamic model for the system.

This motivated the extension of the SVOs presented in [76, ShammaTu] to uncertain plants. However, it was shown that, for uncertain models, the sets of admissible

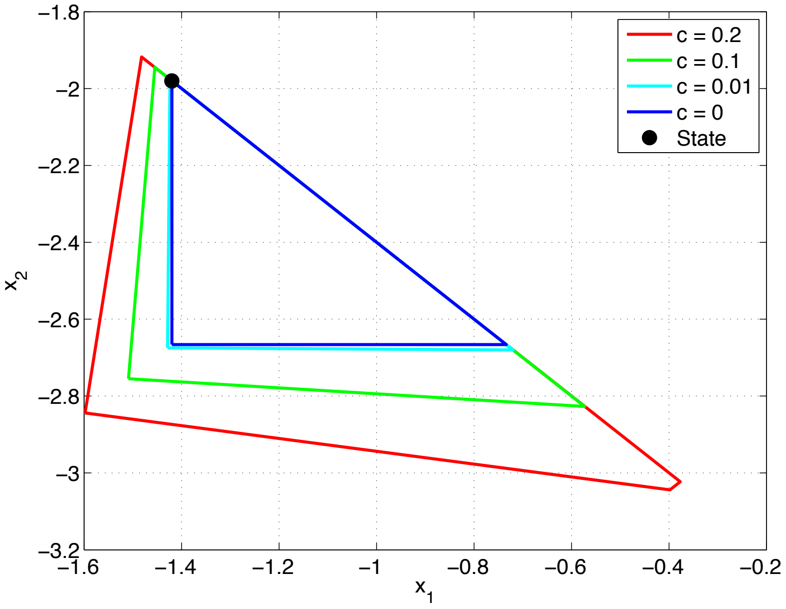


Figure 6.12: True state of the system and estimated set of admissible states, at time $k = 3$ and for several values of c in (6.23).

states of the plant may be non-convex, which typically results in an increased computational burden. Several solutions to this problem were proposed, although they all entail some conservativeness issues.

Some of the most important numerical and computational shortcomings of the SVOs were also addressed in this chapter. Despite the fact that a few constraints may seem very stringent from a practical point of view, they may not be relevant when used to invalidate models in a Multiple-Model Adaptive Control (MMAC) architecture, as shown in Chapter 7.

As an illustration of the capabilities of the SVOs in set-valued state estimation, an example taken from [76, ShammaTu] was used. As expected, the size of the sets of admissible states of the plant increased with the “amount” of uncertainty on the model of the system.

In the following chapter, the SVOs are going to be used for the falsification of uncertain dynamic models.

Chapter 7

Model Falsification Using SVOs

7.1 Introduction

In Chapter 5, it was stated that unmodeled dynamics and adverse exogenous disturbances, can result in erroneous model falsification. Therefore, worst-case approaches, rather than stochastic approaches, are more suitable to address this type of problem. Indeed, in Chapter 6, a type of robust filters, referred to as Set-Valued Observers (SVOs), was used to compute the set of admissible states of a given plant.

Hence, we are now able to integrate the ideas of Chapter 5 with the SVOs developed in Chapter 6. Using such an approach, we are able to construct recursive algorithms – unlike those proposed in [71, 125] – to falsify models.

The main idea in *model falsification using set-valued observers* consists in invalidating dynamic models associated with SVOs whose state estimate, at a given time instant, is the empty set. Thus, as long as a given SVO is providing non-empty set-valued estimates for the state of the plant, the corresponding dynamic model cannot be discarded. Using this line-of-thought, and since the SVOs developed in Chapter 6 are able to cope with model uncertainty, several architectures and algorithms are going to be proposed in this chapter, in order to achieve robust model falsification, *i.e.*, in order to falsify uncertain dynamic models.

As stressed in Chapter 5, model falsification strategies have a wide applicability. Indeed, there are several areas of research where we are interested in distinguishing among a set of eligible dynamic models. Therefore, this chapter is also devoted to the application of this model falsification strategy to Multiple-Model Adaptive Control (MMAC) and to Fault Detection and Isolation (FDI). In the former case, the advantage of using SVOs as a tool for model invalidation has the advantage, over other MMAC schemes, such as [34], of guaranteeing that the "correct" model of the plant is eventually selected, under mild assumptions on the exogenous disturbances. These constraints on the exogenous disturbances are naturally connected to the concept of absolute input distinguishability, introduced in Chapter 5. In terms of FDI, the architectures proposed in this chapter guarantee that, also under mild assumptions on the exogenous disturbances, a fault is *always* detected and isolated after a given number of measurements. Both applications of the model falsification approach using SVOs are illustrated with several numerical examples.

7.1.1 Main Contributions and Organization

This chapter introduces several architectures and algorithms for model falsification using set-valued observers. As stressed, the aforementioned approaches have applicability, for instance, in multiple-model adaptive control and in fault detection and isolation. Moreover, they can be used with uncertain linear parameter-varying systems.

The main contributions of this chapter are as follows:

a) The development of a model falsification strategy using SVOs;

b) The derivation of sufficient conditions for the eventual falsification of all but the "correct" dynamic model of the plant;

c) The application of this model falsification strategy to MMAC, providing stability and performance guarantees;

d) The application of this model falsification strategy to FDI, providing guarantees in terms of fault detection and isolation in finite time, and avoiding false alarms;

e) The illustration of the aforementioned strategies with several numerical examples.

The remainder of this chapter is organized as follows. We start by introducing the main architecture and algorithm for model falsification using SVOs in Section 7.2. In Section 7.3, conditions guaranteeing the selection of a single dynamic model are derived. A numerical example, illustrating the applicability of the technique, is provided in Section 7.4. In Section 7.5, an MMAC architecture using SVOs is presented, while in Section 7.6, a method for FDI using SVOs is introduced. Finally, some conclusions regarding this chapter are discussed in Section 7.7.

7.2 Main Architecture

We assume that the reader is familiar with the model falsification paradigm of Chapter 5. In summary, consider the set of dynamic models depicted in Fig. 7.1. Further suppose that the actual plant can be described by model S_5 in Fig. 7.1. Then, our goal is to *invalidate* or *falsify* all except model S_5, using Set-Valued Observers (SVOs).

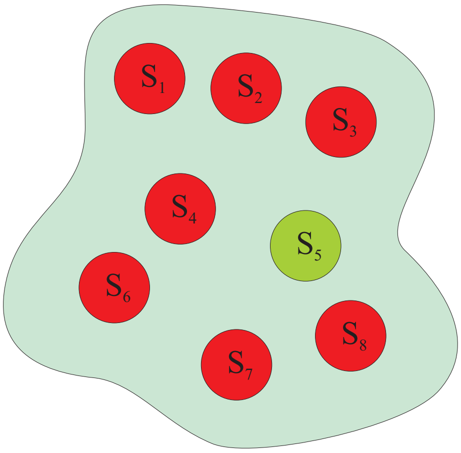


Figure 7.1: Set of models for a dynamic system. Model S_5 denotes the *true* plant.

Let us suppose that the plant whose dynamic model we are trying to identify is described by

$$\begin{cases} x(k+1) &= A\big(\rho(k)\big)x(k) + B\big(\rho(k)\big)u(k) + L\big(\rho(k)\big)d(k), \\ y(k) &= C\big(\rho(k)\big)x(k) + N\big(\rho(k)\big)n(k), \end{cases} \tag{7.1}$$

where $x(\cdot) \in \mathbb{R}^n$ is the state of the plant, $y(\cdot) \in \mathbb{R}^{n_y}$ is the measurement output, corrupted with noise, $n(\cdot) \in \mathbb{R}^{n_n}$, and where $u(\cdot) \in \mathbb{R}^{n_u}$ is the control input, and $d(\cdot) \in \mathbb{R}^{n_d}$ denotes the exogenous disturbances. The vector of parameters, $\rho(k)$, for $k \geq 0$, might be completely or partially known, although it is assumed a priori that $|\rho(\cdot)| \leq \bar{\rho}$, for some known finite $\bar{\rho} \in \mathbb{R}$. Similarly to Chapter 6, the exogenous disturbances and sensor noise are also assumed to be bounded by known finite constants.

For an SVO as in Chapter 6, designed for the dynamic system in (7.1), we denote by $\hat{X}(k)$ the set-valued estimate of the state, provided by that SVO, at time k.

The following lemma is required in what follows to guarantee that the *correct* model of the plant can never be invalidated.

Lemma 7.1. *Consider a Set-Valued Observer (SVO) as in Chapter 6, for the plant in* (7.1)*. Then, if $\hat{X}(k) = \emptyset$ for some $k \geq 0$, the input/output data cannot be described by* (7.1)*.*

Proof. By construction of the SVOs in Chapter 6, if $\hat{X}(k)$ is empty for some k, then the observations are not compatible with the model of the plant. □

The architecture depicted in Fig. 7.2, together with the result in Lemma 7.1, can be used to address the problem of invalidating a given discrete-time LPV system. In reference to Fig. 7.2, the model invalidation block is composed of an SVO and a *Logic* block, which decides whether or not the input/output sequence is compatible with the LPV model of the plant, according to the emptiness or not of the set-valued estimate of the state, at each sampling time.

Therefore, while a given model is not invalidated (or falsified), it can be seen as a *plausible* description of the input/output behavior of the plant. This strategy can be readily extended to the case where several models are *tested* at each sampling time. Indeed, let $\mathcal{S}$ denote the set of *plausible* or *admissible* models of the plant.

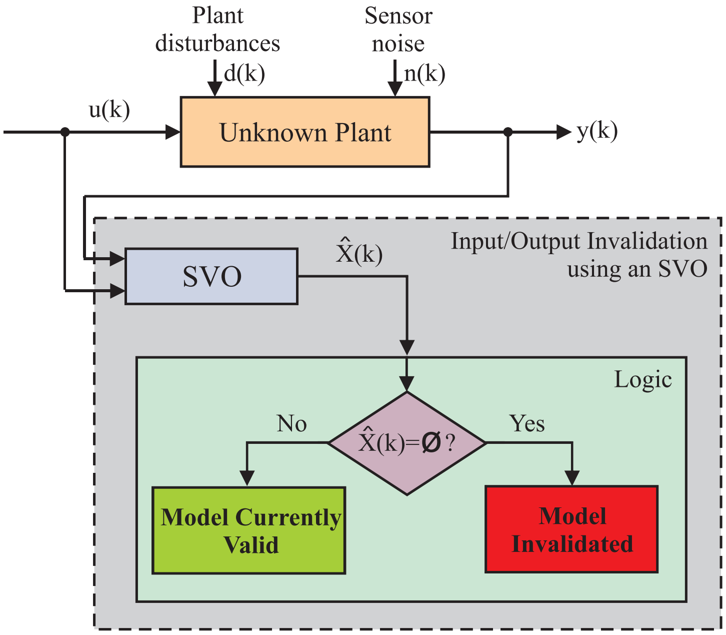


Figure 7.2: Falsification of a dynamic model using a set-valued observer (SVO).

We assume that $\mathcal{S}$ is a finite set, with cardinality $N_{\mathcal{S}}$, and that each $S_i \in \mathcal{S}$ can be described by

$$S_i : \left\{ \begin{array}{rcl} x_i(k+1) & = & A_i\big(\rho(k)\big)x_i(k) + B_i\big(\rho(k)\big)u(k) + L_i\big(\rho(k)\big)d_i(k), \\ y_i(k) & = & C_i\big(\rho(k)\big)x_i(k) + N_i\big(\rho(k)\big)n_i(k), \end{array} \right. \tag{7.2}$$

for each $i \in \{1, \cdots, n_{\mathcal{S}}\}$, and using a nomenclature similar to that of (7.1).

Hence, we are now in conditions of proposing the architecture depicted in Fig. 7.3, for model falsification using SVOs.

Each SVO provides the *Logic* block in Fig. 7.3 with set-valued estimates of the state, according to their corresponding a priori information regarding the dynamic model of the plant, the initial condition, the exogenous disturbances and measurement noise, and the model uncertainty.

Different approaches can be used in the implementation of this *Logic* block. Nevertheless, in this thesis we restrict ourselves to the algorithm depicted in Fig. 7.4. This algorithm takes advantage of Lemma 7.1 to guarantee that the correct dynamic model is not invalidated.

Therefore, the architecture in Fig. 7.3, jointly with the algorithm in Fig. 7.4, selects the first dynamic model whose corresponding SVO did not yet provide an

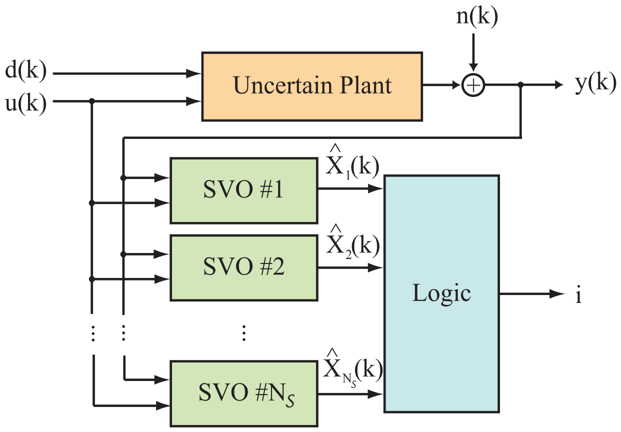


Figure 7.3: Architecture for model falsification using SVOs.

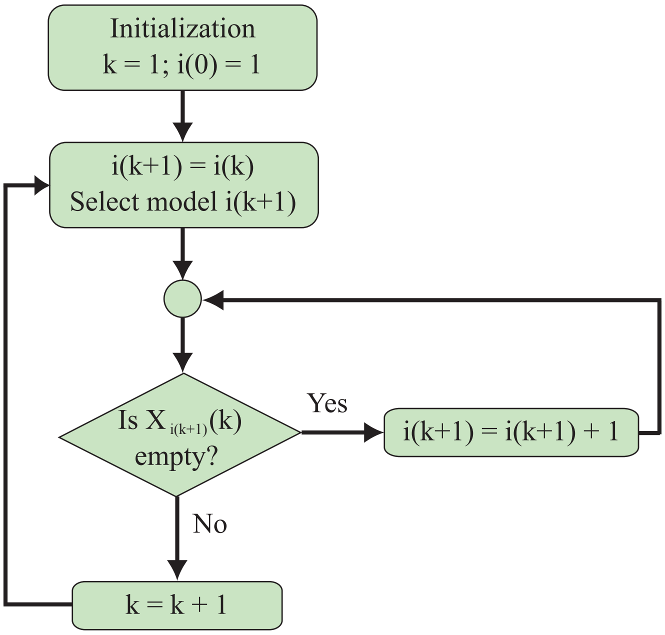


Figure 7.4: Algorithm proposed to select a model at each sampling time.

empty-valued estimate for the state of the plant. This heuristic can obviously be substituted by any other, as long as the select dynamic model has a corresponding SVO which provides non-empty set-valued estimates for the state of the plant.

7.3 Guarantees of Model Selection

In the previous section, an architecture was developed for model falsifications, using SVOs, that guarantee that, by construction, the "correct" dynamic model of the plant can never be invalidated. The main goal in this section is to provide sufficient conditions assuring that all except *one* single dynamic model (or one single family of dynamic models) are falsified after a given number, N, of measurements. Moreover, we would also like to estimate a lower bound for N.

In order to do that, we use the notion of absolute input distinguishability, introduced in Chapter 5. Indeed, let N be a given constant and suppose the following assumption is satisfied.

Assumption 7.1: Let $\mathcal{S}$ be the (finite) set of *admissible* models of the plant. If $S_i \in \mathcal{S}$ and $S_j \in \mathcal{S}$, with $i \neq j$, then S_i and S_j are absolutely (X_o, U, W)-input distinguishable in N sampling times. □

Testing whether or not two given LPV uncertain models are absolutely (X_o, U, W)-input distinguishable in N sampling times can be done using the methods described in Chapter 5. In particular, it is important to notice that a persistence of excitation condition is required on the exogenous disturbances. Indeed, it was shown in Chapter 5 that this condition can be written as a lower bound on the intensity of the perturbations. We recall that the theory in Chapter 5 can be seen as a tool to aid the design of model falsification schemes.

In addition to Assumption 7.1, let us also posit the following one.

Assumption 7.2:

a) The initial state of the plant satisfies $x(0) \in X_o$;

b) The control input sequence satisfies $u(j) \in U$ for all $j \geq 0$;

c) The sequence of disturbances satisfies $(d(j), n(j)) \in W$ for all $j \geq 0$.

□

Lemma 7.2. *Let $\mathcal{S}$ be the (finite) set of* admissible *models of the plant and suppose that Assumptions 7.1 and 7.2 are satisfied. Further suppose that we use the algorithm in Fig. 7.4, and that the Set-Valued Observers (SVOs) are non-conservative. Then, at any time k such that $k \geq N$, there is, at most, a single SVO whose state estimate is not the empty set.*

Proof. Since Assumptions 7.1 and 7.2 are satisfied, we have that, at any time k such that $k \geq N$, no more than a single dynamic model in $\mathcal{S}$ is compatible with the input/output dataset. Moreover, the SVOs are assumed non-conservative, which means that every single element in the corresponding set-valued state estimate is compatible with the input/output dataset. Therefore, there cannot be two or more SVOs with non-empty state estimates. □

Hence, we are now in conditions of guaranteeing that the *correct* model of the plant is selected using the proposed architecture and algorithm.

Theorem 7.1. *Consider a dynamic system, S_r, described by* (7.2) *with $i = r$, and suppose that $S_r \in \mathcal{S}$. Further suppose that we use the algorithm in Fig. 7.4 with system S_r, and that the Set-Valued Observers (SVOs) are non-conservative. If Assumptions 7.1 and 7.2 are satisfied, then, at any time k such that $k \geq N$, there is exactly a single SVO whose state estimate is not the empty set.*

Proof. The proof is straightforward from Lemmas 7.1 and 7.2. □

Remark 7.1: Another interpretation of the result in Theorem 7.1 is as follows: If only a single model in $\mathcal{S}$ is able to explain the input/output dataset, then the algorithm in Fig. 7.4 guarantees that this model is the only one which will not be invalidated, as long as Assumptions 7.1 and 7.2 are satisfied and the SVOs are non-conservative. ◇

7.3.1 Uncertain Dynamic Models

Notice that Lemma 7.2 (and, consequently, Theorem 7.1) can only be applied if the SVOs are non-conservative. For the case of uncertainty in matrix C, however, it was shown in Chapter 6 that a relaxation was needed to compute the set-valued estimate of the state at the next sampling time. Thus, the distinguishability condition in Assumption 7.1 is no longer sufficient to guarantee the invalidation of all but the *correct* model of the plant.

Indeed, in Section 6.3, it was shown that a system modeled by (6.10) can also be described by

$$S \equiv \left(\bar{S}_j + N(k)\bar{n}_i\right) + \sum_{j=1}^{n_\Delta} \left(\tilde{S}_j + N(k)\bar{n}\right), \tag{7.3}$$

where

$$\tilde{S}_j : \begin{cases} x_j(k+1) &= A(k)x_j(k) + B(k)\Delta_j(k)u(k) + L(k)\bar{d}_j(k), \\ y_j(k) &= C_j(k)x_j(k), \end{cases}$$

and $\bar{d}_j(k)$ was defined in (6.13) for each $j \in \{1, \cdots, n_\Delta\}$ and $k \geq 0$.

Therefore, if instead of considering the system in (6.10), we take into account the description (7.3) in Assumption 7.1, the results in Lemma 7.2 and Theorem 7.1 will still hold.

The shortcoming of this approach stems from the fact that, typically, the family of dynamic systems described by (6.10) is smaller than (and contained inside of) the family of dynamic systems described by the corresponding formulation in (7.3). Hence, if Assumption 7.1 is satisfied for the original set of eligible models, $\mathcal{S}$, it may not be satisfied for the *modified* one.

A similar issue is found when the A matrices of the dynamics of the models are uncertain, except for the case where $A_j(k)$ for all $j \in \{1, \cdots, n_\Delta\}$ and $k \geq 0$ are matrices with unitary ranks. However, if indeed $\operatorname{rank}(A_j(k)) > 1$ for some $j \in \{1, \cdots, n_\Delta\}$ and $k \geq 0$, an analogous reformulation of Assumption 7.1 has to be performed, in order for Theorem 7.1 to hold.

7.4 Simulations

This section presents a case study that illustrates the applicability of the proposed technique for model falsification. We exploit the example in [159], which considers a single-link robotic arm, with a revolute elastic joint, rotating in a vertical plane, as depicted in Fig. 7.5. The dynamics of the system are described by

$$\begin{cases} J_1\ddot{q}_1 + F_\ell\dot{q}_1 + k(q_1 - q_2) + mg\ell\sin q_1 & = & 0, \\ J_m\ddot{q}_2 + F_m\dot{q}_2 - k(q_1 - q_2) & = & u, \end{cases} \tag{7.4}$$

where we omitted the time dependence of the states. The link and the rotor displacements are denoted by q_1 and q_2, respectively. The link inertia, J_ℓ, the motor rotor inertia, J_m, the elastic constant, k, the link mass, m, the gravity constant, g, the center of mass, ℓ, and the viscous friction coefficients, F_ℓ, F_m, are constant parameters, whose values are summarized in Table 7.1. The control input, u, is the torque delivered by the motor.

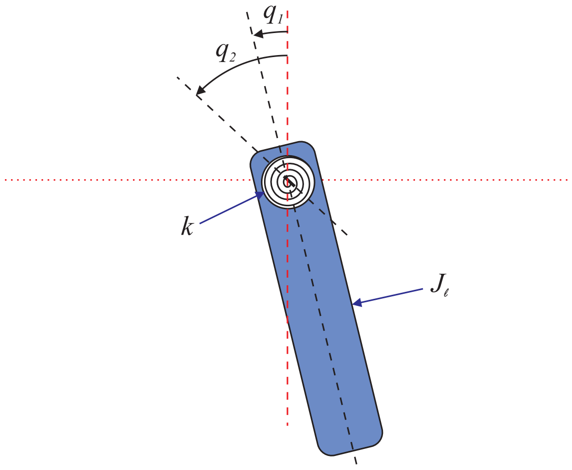


Figure 7.5: Single-link robotic arm, with a revolute elastic joint, rotating in a vertical plane.

Similarly to [159], we assume that q_1, q_2 and $\dot{q}_2$ are available for measurement.

Table 7.1: Parameters of the single-link robotic arm with revolute elastic joint, in a vertical plane.

Parameter	J_ℓ	J_m	k	m	g	ℓ	F_ℓ	F_m
Value (in SI units)	2	1	2	4	9.8	0.5	0.5	1

Thus, the model in (7.4) can be rewritten as

$$\begin{cases} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{-k}{J_\ell} & \frac{-F_\ell}{J_\ell} & \frac{k}{J_\ell} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{k}{J_m} & 0 & \frac{-k}{J_m} & \frac{-F_m}{J_m} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{-mg\ell}{J_\ell} \sin x_1 \\ 0 \\ \frac{u}{J_m} \end{bmatrix} + \begin{bmatrix} 0 \\ \eta \\ 0 \\ 0 \end{bmatrix} \\ y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \\ n_3 \end{bmatrix} \end{cases} \tag{7.5}$$

where $x_1 = q_1$, $x_3 = q_2$, and where $\eta : \mathbb{R}^+ \to \mathbb{R}$ represents the model uncertainty in the derivative of $\dot{q}_1$, with the constraint

$$|\eta| \leq \bar{\eta} = 0.05(mg\ell/J_\ell) \sin x_1.$$

In addition to the model uncertainty proposed in [159], we also consider that k is uncertain, with the constraint

$$k \in K := [1.65,\, 2.05] \text{ N/m}.$$

The measurement noise is obtained from a random generator with uniform probability distribution, and with the following bounds:

- $|n_1| \leq 0.001$ rad;

- $|n_2| \leq 0.01$ rad;

- $|n_3| \leq 0.01$ rad/s;

For the simulations presented next, we also considered that

$$u = 15\sin(2t)$$

and that the initial state of the plant is given by $x(0) = 0$. The system was discretized with a sampling period of $T_s = 250$ ms.

Regarding the implementation of the SVOs, there are some points of interest in this example. The nonlinearity due to the dependence of $\dot{x}_2$ on $\sin x_1$ can be seen as an exogenous disturbance, since $|\sin x_1| \leq 1$. A similar reasoning can be used regarding η. The uncertainty in k can also be handled by the SVOs, as explained in Chapter 6. Indeed, the value of k is assumed to belong to one of the following intervals:

- $K_1 := [1.65,\ 1.80]$ N/m;
- $K_2 := [1.80,\ 1.95]$ N/m;
- $K_3 := [1.95,\ 2.05]$ N/m.

The first question to arise is whether or not we are able to *distinguish* among the 3 a priori eligible (uncertain) models of the plant. In particular, it is desirable to ascertain an upper bound (whenever it exists) to the number of iterations, N, required to select the *correct* region of uncertainty. As stressed in Section 5.2, the value of N is dependent on the level of the exogenous disturbances, the measurement noise, and the control input signal.

Moreover, regions K_1 and K_2 are clearly not *absolutely distinguishable* since the intersection of these two intervals is not the empty set. Indeed, for $k = 1.80$ N/m, both K_1 and K_2 are admissible regions. A similar reasoning can be used regarding K_2 and K_3. Therefore, although in practice the aforementioned model falsification architecture with SVOs can be designed for those regions, it will not be possible to establish any meaningful results regarding absolute distinguishability, using these intervals.

This fact is further illustrated in Fig. 7.6, where the absolute distinguishability of the following two regions is assessed:

- $\bar{K}_1 := [1.65,\ 1.65 + \delta]$ N/m;

- $\bar{K}_2 := [1.95 - \delta,\ 1.95]$ N/m.

The value of δ is the width of the intervals. As depicted in Fig. 7.6 – which was obtained with the aforementioned constraints on the exogenous disturbances, measurement noise, and control input –, as we diminish the distance between intervals $\bar{K}_1$ and $\bar{K}_2$, the number of iterations required to ensure absolute distinguishability increases.

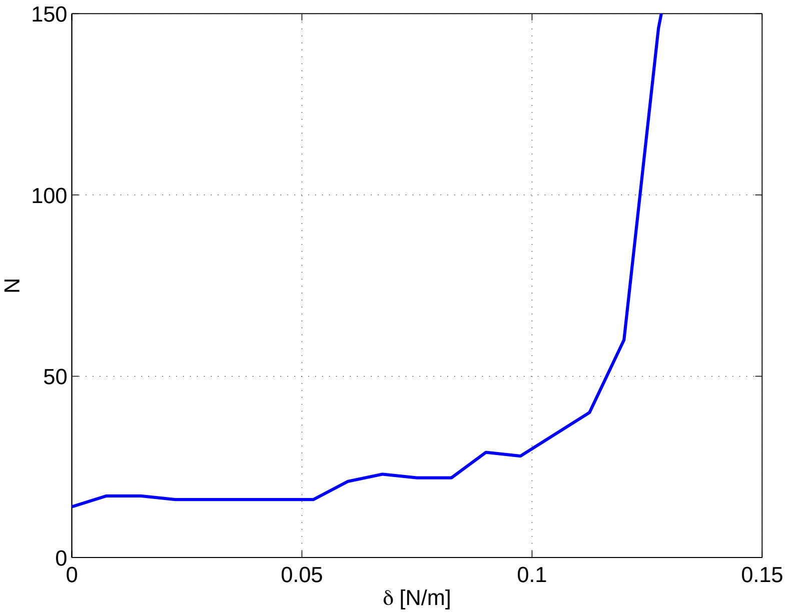


Figure 7.6: Number of iterations, N, required to ensure the absolute distinguishability of the models with $k \in \bar{K}_1$ or $k \in \bar{K}_2$.

Now let us suppose (aside from the fact that we cannot provide absolute distinguishability guarantees for regions K_1 and K_2, and regions K_2 and K_3) that we synthesize the SVOs for the 3 intervals. The idea at this point is to obtain the sets $\tilde{K}_i$, with $i = \{1, 2, 3\}$, such that:

a) $\tilde{K}_i \subseteq K_i$ for $i \in \{1, 2, 3\}$;

b) if $k \in \tilde{K}_j$, then all except region K_j are invalidated in N_r sampling times.

Let us consider that $N_r = 100$, *i.e.*, that we are interested in finding the largest sets $\tilde{K}_i$, with $i = \{1, 2, 3\}$, satisfying the aforementioned constraints with $N_r = 100$.

One way of computing these sets is to grid the space of parameters and to assess the distinguishability of the models at the points of that grid. In particular, suppose that we uniformly grid the set K. Then, for each point of the grid, we compute a upper bound (if it exists) on the number of iterations required to guarantee that all except the correct model of the plant are falsified, using the methods described in the previous section. The results are depicted in Fig. 7.7.

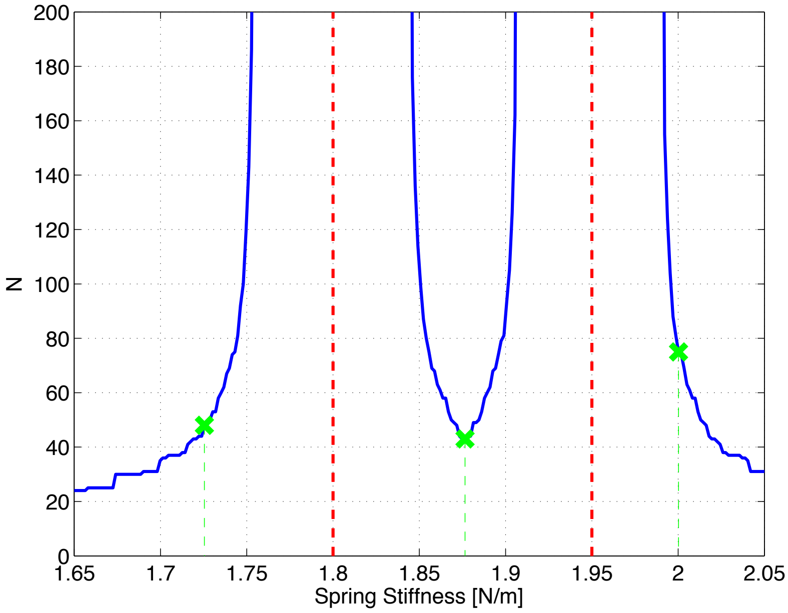


Figure 7.7: Number of iterations, N, guaranteeing all except one model are falsified, as a function of the spring stiffness. The green crosses denote the mean points of the spring stiffness intervals.

From Fig. 7.7, it is apparent that, as we approach to the boundaries of the spring stiffness intervals, the value of N increases rapidly, as expected. Therefore, for values of the spring stiffness near 1.75 N/m or near 1.95 N/m, there are no guarantees that eventually a single model is not falsified. Moreover, if indeed we consider $N_r = 100$, then it is straightforward to conclude that the sets $\tilde{K}_i$, with $i = \{1, 2, 3\}$ and $N_r = 100$, are defined as

- $\tilde{K}_1 = [1.65,\ 1.74]$ N/m;

- $\tilde{K}_2 = [1.85,\ 1.90]$ N/m;

- $\tilde{K}_3 = [1.99,\ 2.05]$ N/m.

Finally, we present some time-domain simulations illustrating the behavior of the proposed model falsification methodology. Figure 7.8 depicts the output of the plant for values of the spring constant, k, inside each of the intervals K_i, for $i = \{1, 2, 3\}$.

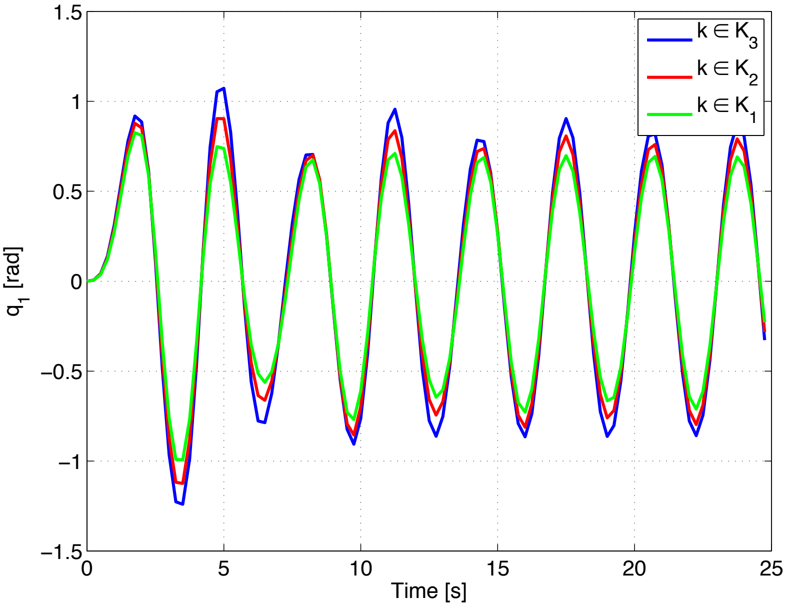


Figure 7.8: Output of the plant for different values of the spring stiffness, k.

We performed 10 Monte-Carlo (MC) simulation runs for each of the following spring stiffness coefficients:

a) $k_A = 1.725$ N/m;

b) $k_B = 1.875$ N/m;

c) $k_C = 2.000$ N/m.

The green crosses in Fig. 7.7 mark the upper bounds on the number of iterations, N, required to guarantee the invalidation of all except the correct model, for these values of the spring constant. Table 7.2 summarizes those theoretical upper bounds and also presents the results obtained by averaging the 10 MC runs.

From the simulation results, it is clear that the proposed model falsification strategy is, in general, able to falsify the invalid models of the plant in a significantly

Table 7.2: Number of iterations required to guarantee the invalidation of all except the correct model – theoretical vs simulation results.

k [N/m]	Theoretical Upper Bound	Average of 10 MC Runs	Maximum on 10 MC Runs
k_A	48	6.8	7
k_B	43	12.4	13
k_C	75	13	13

smaller number of iterations than the theoretical upper bound. Nevertheless, we stress that the upper bounds should only be interpreted as *worst-case* scenarios.

7.5 Application to MMAC

7.5.1 Introduction

In this section, the problem of Multiple-Model Adaptive Control (MMAC) is addressed using the SVO-based model falsification technique described in this chapter. In particular, for the sake of simplicity, we are going to address the case where the dynamic model of the system to be controlled has a single parametric uncertainty, $\rho \in [\rho_{\min}, \rho_{\max}]$. Although several switching MMAC methodologies are available to solve this problem, they all share the same principles: in terms of design, we divide the (large) set of parametric uncertainty, Ω, into N (small) subregions, Ω_i, $i = \{1, \cdots, N\}$ – see Fig. 7.9 – and synthesize a non-adaptive controller for them; in terms of implementation, we try to identify which region the uncertain parameter, ρ, belongs to, and then use the controller designed for that region. The approach presented herein *discards* the regions to which the uncertain parameter, ρ, *cannot* belong.

For a list of advantages of this type of control see, for instance, [31]. Several MMAC architectures have been proposed that provide stability and/or performance

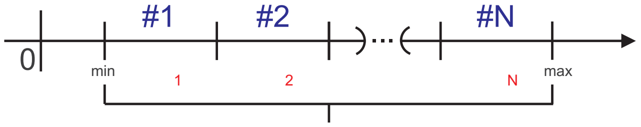


Figure 7.9: Uncertainty region, Ω, for the parameter ρ, split into N subsets, Ω_i, $i = \{1, \cdots, N\}$.

guarantees as long as a set of assumptions are met. For instance, [41] uses a parameter estimator to select a controller, guaranteeing stability of the closed-loop. Another MMAC approach, referred to as Robust Multiple-Model Adaptive Control (RMMAC), which was introduced in [58] and references therein, and that was previously mentioned in this thesis, uses a bank of Kalman filters for the identification system and a hypothesis testing strategy to select the controllers. For this case, although the simulation results indicate that high levels of performance are obtained, the only guarantees that can be provided are in terms of stability – see [69]. In [105], calibrated forecasts are used to guarantee the stability of the closed-loop. The theory of unfalsified control – see [50] and references therein – uses the controlled output error to decide whether the selected controller is delivering the desired performance or not. The authors in [54] use a Lyapunov-based approach to select controllers, and hence require an in-depth knowledge of the plant. Some of the assumptions required by these methodologies are often unnatural or cannot be verified in practice.

The approach adopted in this thesis is somewhat different to the above MMAC architectures. Instead of trying to identify the *correct* region, *i.e.*, the region where the uncertain parameter takes value, by hypothesis testing or parameter estimation, we exclude the *wrong* regions. In other words, if the time-evolution of the inputs and outputs of the plant cannot be explained by a model with uncertain parameter ρ, such that $\rho \in \Omega_i$, then region Ω_i cannot be the one which the uncertain parameter belongs to. This is naturally aligned with the SVO-based model falsification strategy introduced in this chapter.

In summary, the approach provided in this section is to use SVOs to decide which non-adaptive controllers should *not* be selected. Similarly to other MMAC architectures, we use a bank of observers – in our case, SVOs –, each of which *tuned* for a pre-specified region of uncertainty. However, we utilize the observers to *discard* regions, rather than to identify them. Using this strategy, we are able to provide robust stability and performance guarantees for the closed-loop, even when the model of the plant is uncertain.

7.5.2 MMAC/SVO Architecture

Figure 7.10 depicts one of the MMAC architectures adopted to use with the SVOs, referred to as MMAC/SVO architecture. As previously mentioned, suppose that, for the sake of simplicity, the plant depends upon only one uncertain parameter, ρ. It is known, however, that $\rho \in \Omega$, for some set $\Omega \subseteq \mathbb{R}$. The methodology presented herein can obviously be generalized for plants with a higher number of parametric uncertainties.

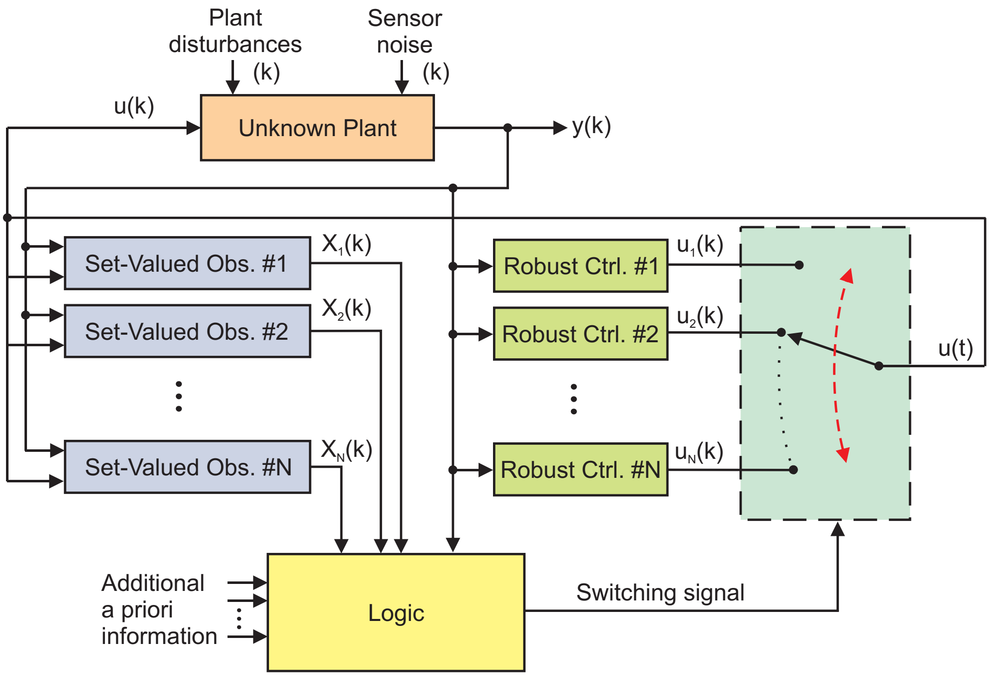


Figure 7.10: Multiple-model adaptive control with set-valued observers (MMAC/SVO) architecture. X_i is the set-valued state estimate provided by SVO #i.

We follow very closely the method presented in [58], to design the MMAC/SVO control system. For starters, we assume that a single and non-adaptive controller (referred to as Global Non-Adaptive Robust Controller - GNARC) is not able to achieve the desired performance for the whole uncertainty region. Therefore, we need to divide this region, Ω, into several smaller regions, say $\Omega_1, \Omega_2, \cdots, \Omega_N$, such that $\Omega_1 \bigcup \Omega_2 \bigcup \cdots \bigcup \Omega_N = \Omega$. In order to do so, we first compute the maximum (ideal) performance that we can achieve. This is obviously the case where we know the exact value of the otherwise uncertain parameter, ρ. To a controller designed for a fixed value of the uncertain parameter, ρ, we call FNARC (Fixed Non-Adaptive Robust Controller), using the same terminology as in [58].

The design proceeds by defining the desired performance for the closed-loop, when the parameter ρ is uncertain. Without loss of generality, we assume that, for each value of the uncertain parameter, ρ, we want the performance of the MMAC/SVO not to be smaller than a fixed percentage of the corresponding FNARC. This naturally leads to the splitting of set Ω into smaller subsets, as illustrated in Fig. 7.11.

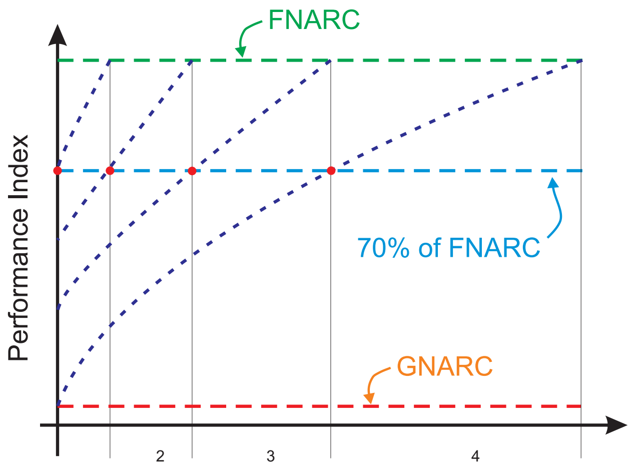


Figure 7.11: Design procedure to split the uncertainty region, Ω, into smaller subsets, so that the desired performance is achieved inside each of them. In this example, it is considered that the desired performance is 70% of that of the FNARC.

For each of these subsets Ω_i, $i = \{1, \cdots, N\}$, a controller referred to as LNARC

(Local Non-Adaptive Robust Controller) is synthesized. Furthermore, an SVO should also be designed for each of the subsets, using the methodology previously introduced.

Remark 7.2: We argue that, for any realistic application, these LNARCs should also be robust to plant model error in addition to parameter ρ. Thus, using an LFT representation for the system may be useful if, for instance, mixed-μ controllers are used – see [77, 78]. The LPV/BMI controllers introduced in Chapter 2 are an example of the type of controllers that can be used in this architecture. $\diamond$

7.5.3 Controller Selection Algorithm

Having described the architecture and the design procedure of the MMAC/SVO, we propose an algorithm to select the appropriate controller at each sampling time. In reference to Fig. 7.10, this subsection is devoted to the description of the behavior of the block entitled *Logic*.

Several approaches can be used to tackle this decision problem. At this point, we adopt the simple solution depicted in Fig. 7.4, for SVO-based model falsification. This strategy takes into account the fact that, whenever $\rho \in \Omega_i$, the SVO $\#i$ does never fail, *i.e.*, $\hat{X}_i(k)$ is never empty. On the other hand, if $\rho \notin \Omega_i$, then it can happen that, for some t_0, we have $\hat{X}_i(k) = \emptyset$, for all $k \geq t_0$.

Thus, Fig. 7.12 depicts the algorithm suggested for selecting the controllers. In summary, the main idea is to start by using any controller in the initial set of plausible controllers, and then remove from the loop controllers whose corresponding plant models have been disqualified. For the sake of simplicity, the controllers are selected in a sequential fashion, in this case, *i.e.*, if model #1 is invalidated, we switch to controller #2, while if model #2 is invalidated, we switch to controller #3, and so on. However, other algorithms can be used, as long as the selected controller does never correspond to a previously falsified model.

We stress that the main advantage of this algorithm is that, under mild assumptions, we can guarantee robust stability and performance for the closed-loop, as shown in the sequel.

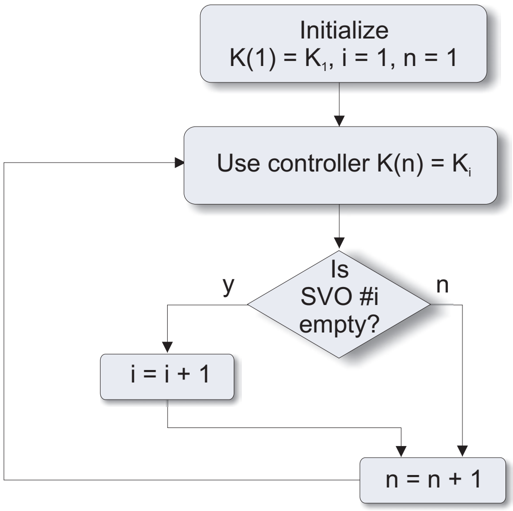


Figure 7.12: Algorithm for the *Logic* block of the MMAC/SVO architecture for time-invariant systems.

Remark 7.3: Although the dynamics of the disturbances need not be explicitly taken into account in the design of the SVOs, they can be key in the selection of the appropriate model of the plant. Indeed, augmenting the dynamic model of the plant with that of the disturbances, may result in faster invalidation of models which would have been plausible, had not the dynamics of the disturbances been known a priori. An example illustrating this effect is presented in Chapter 8. ◇

RMS Analysis

Since the MMAC/SVO can be seen as a *worst case* approach, a controller is not invalidated unless the input/output sequences cannot be explained by the dynamics of the closed-loop system. Hence, an enhancement of the MMAC/SVO architecture is now proposed, based on the fact that, on the one hand, we cannot "exclude" unfalsified controllers, and that, on the other hand, if a given controller is unlikely to be the *correct* one, then one should try another one, if available.

As an example, the Root Mean Square (RMS) of the output error of the closed-loop system, provides an additional method to assess whether or not we are using the *correct* controller at each time. This idea was suggested by Professor Athans, and can be summarized as follows. Since we can compute a priori the maximum RMS of the closed-loop system, given a certain bound on the intensity of the disturbances, this maximum RMS value can be used to aid in the invalidation of dynamic models. More specifically, if controller $\#i$ is connected to the loop, and the RMS of the output of the system is greater than the upper bound on the RMS for region $\#i$ – which was computed offline –, then clearly model $\#i$ is not the *correct* one (or the bounds on the disturbances were violated), if the steady state was already attained.

The main advantage of this method is that we can potentially discard regions of uncertainty faster than simply using the SVOs. However, as a shortcoming, the RMS values computed are only valid for steady state, which may not be trivial to assess in practice.

The integration of other decision schemes with the MMAC/SVO architecture can be done in a similar way to what was done with the Stability Overlay (SO) – see Chapter 4. In order to guarantee the properties of the MMAC/SVO architecture in Fig. 7.10, a model can only be excluded from the set of eligible ones if it is invalidated by the SVOs. Nevertheless, whenever this set has more than a single element, we are free to select a controller which has not violated the aforementioned RMS condition. The resulting algorithm is depicted in Fig. 7.13, where RMS_i is the value of the current RMS of a window of the output, and $\overline{\mathrm{RMS}}_i$ is the a priori bound on that RMS value.

The set S contains all the controllers which have not violated the a priori closed-loop RMS assumption, whenever connected to the loop. If this set is empty, then we can select any controller for which the corresponding model has not been falsified by the SVOs. Otherwise, we can select LNARC $\#i$, as long as $i \in S$ and SVO$\#i \neq \emptyset$. If none of the indexes in S satisfies SVO$\#i \neq \emptyset$, then we can select any LNARC $\#i$, for which SVO$\#i \neq \emptyset$.

7.5.4 Guaranteed Stability and Performance

Preliminary results on the closed-loop stability and performance guarantees provided by the MMAC/SVO architecture were presented in [142]. However, the assumptions in [142], required to guarantee performance, are seldom easy to be checked in prac-

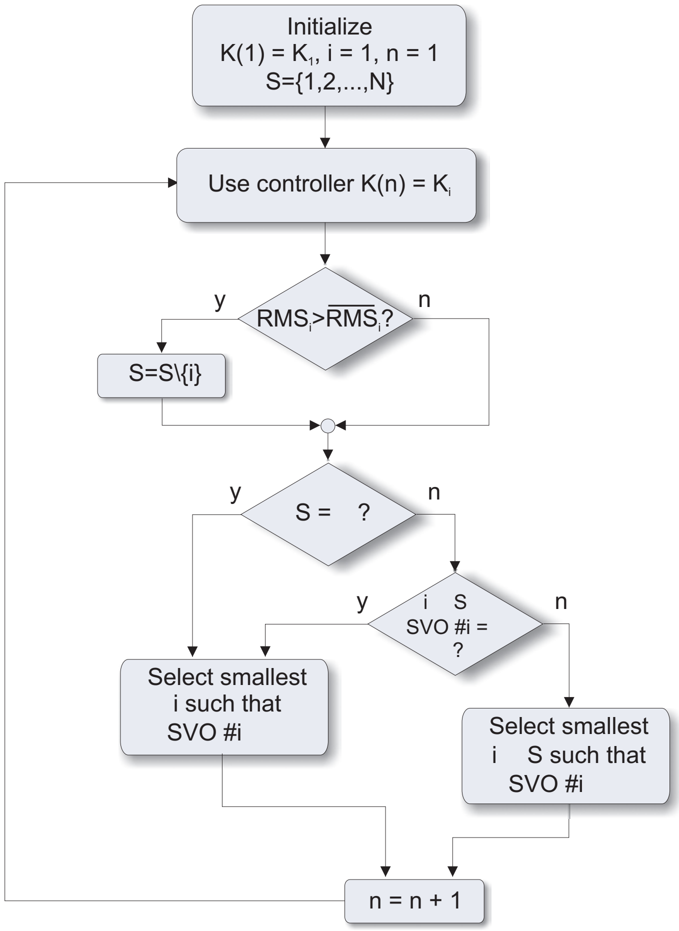


Figure 7.13: MMAC/SVO-RMS algorithm.

tice. Since these assumptions essentially amount for the eventual selection of the appropriate controller, it is natural to use the concept of absolute input distinguishability, introduced in Chapter 5, to pose these requirements.

Despite of that, the MMAC/SVO approach guarantees a stable closed-loop under much less restrictive assumptions. Indeed, we first show that, for plants with no *drifting* parameters (see Definition 4.1), the MMAC/SVO is able to provide stability guarantees, by demonstrating that a stabilizing controller is eventually connected to the loop.

The proof for stability is based upon the fact that the SVOs are non-conservative, *i.e.*, if $\hat{X}_i(k) \neq \emptyset$, then the output of the plant, $y(k)$, can be explained by the previous and current inputs and outputs, and for some $\rho \in \Omega_i$. This statement will be explained in a more formal manner in the sequel. Once again, for the sake of simplicity, we use a plant with only one uncertain parameter.

Consider a system described by

$$\left\{ \begin{array}{rcl} x(k+1) & = & A(k)x(k) + B(k)u(k) + L(k)d(k) \\ y(k) & = & C(k)x(k) + N(k)n(k), \end{array} \right. \tag{7.6}$$

where $x(0) = x_o$, $x_o \in X(0)$, $x(k) \in \mathbb{R}^n$, $d(k) \in \mathbb{R}^{n_d}$, $n(k) \in \mathbb{R}^{n_n}$, and $u(k) \in \mathbb{R}^{n_u}$, for $k \geq 0$. Moreover, $|d| \leq 1$, and $|n| \leq 1$, and $d(\cdot), n(\cdot) \in \ell_2$. We note that the system matrices are equivalently described in an LPV fashion as

$$\left\{ \begin{array}{rcl} A(k) & := & A(\rho(k)), \\ B(k) & := & B(\rho(k)), \\ L(k) & := & L(\rho(k)), \\ C(k) & := & C(\rho(k)), \\ N(k) & := & N(\rho(k)), \end{array} \right.$$

where $\rho(k)$ is a vector of parameters for each $k \geq 0$. Consider a partitioning of the uncertainty set, Ω, as described in the previous subsection ($\Omega = \Omega_1 \bigcup \Omega_2 \bigcup \cdots \bigcup \Omega_N$). Moreover, we posit the following assumptions.

Assumption 7.3: For each of the uncertainty subsets, $\Omega_i, i = \{1, \cdots, N\}$, there is at least one LNARC, referred to as $C_i(\cdot)$, that is able to asymptotically stabilize any plant with model (7.6) and $\rho \in \Omega_i$. □

Let

$$\tilde{X}(y(k),k) = \{x : y(k) = C(k)x + n, |n| \leq 1\},$$

and

$$\bar{X}_i(u(k),k) = \{x : x = A(k)w + B(k)u(k) + L(k)d, w \in X_i(k-1), |d| \leq 1\}.$$

Assumption 7.4: The solution of the SVO $\#i$ is given by

$$X_i(y(k),u(k),k) = \bar{X}_i(u(k),k) \cap \tilde{X}(y(k),k).$$

In words, the solutions of the SVOs are non-conservative. □

Assumption 7.5: There exists $i^\star \in \{1,\cdots,N\}$ such that

$$x(k) \in X_{i^\star}(y(k),u(k),k),$$

for all $k \geq 0$. □

Notice that Assumption 7.5 guarantees that the *true* plant model belongs to the family of *legal* models of at least one of the SVOs.

Assumption 7.6: The closed-loop system with any of the N eligible controllers does not have a finite escape time. □

We stress that Assumption 7.6 is automatically satisfied if, for instance, all the N controllers and the plant are Linear Time-Invariant (LTI) systems – see Chapter 4.

Theorem 7.2. *Suppose Assumptions 7.3–7.6 are satisfied. Then, the closed-loop system with the MMAC/SVO scheme is input/output stable.*

Proof. We first show that the number of switchings is finite. Then, by contradiction, we prove that the closed-loop system is input/output stable.

If $X_i(y,u,k) = \emptyset$, then $y(k)$ cannot be explained by the uncertain plant model used by SVO $\#i$. Thus, we switch to a different controller. According to Assumption 7.5, at least for one value of $j \in \{1,\cdots,N\}$, $x(k) \in X_j(y,u,k), \forall_k$. Hence, the number of switchings is finite and smaller than N. In other words, for some large enough t_o, the controller selected at time instant $k \geq t_o$ is always the same.

Next, suppose that $|y(k)| \to \infty$ as $k \to \infty$. Let $C_j(\cdot)$ be the controller selected for $k \geq t_o$. According to Assumption 7.4, there is a sequence $(d(k), n(k))$, with $|n| \leq 1$ and $|d| \leq 1$, such that $y(k)$ can be obtained with model (7.6) with $\rho \in \Omega_j$. However, according to Assumption 7.3, controller $C_j(\cdot)$ is able to stabilize any plant with $\rho \in \Omega_j$. Since $|d|$ and $|n|$ are bounded, and according to Assumption 7.6, there cannot exist a sequence $(d(k), n(k))$ such that $|y(k)| \to \infty$, which is a contradiction. □

Performance Guarantees

Finally, performance guarantees are provided for the closed-loop system using the MMAC/SVO scheme. It should be noticed that this was the primary reason for the discussion in Chapter 5, *i.e.*, the quest for a decision subsystem which guarantees the selection of the appropriate controller, in a finite number of iterations.

For the sake of clarity, we introduce the following notation to represent the closed-loop system with a given controller. Let the plant be described by (7.6), with $\rho(k) \in \Omega_{i^\star}$ for all $k \geq 0$ and some $i^\star \in \{1, \cdots, N\}$. Denote by $CL_{ij}(\cdot)$ the closed-loop system obtained by interconnecting a plant described by (7.6), where $\rho(k) \in \Omega_i$, with the LNARC $C_j(\cdot)$. Further define

$$\xi(k) := \begin{bmatrix} d(k) \\ n(k) \end{bmatrix},$$

and let $z(\cdot)$ denote a performance output of system (7.6), defined as

$$z(k) := C_z(\rho(k))x(k),$$

for $k \geq 0$. Moreover, suppose that the following assumption is satisfied:

Assumption 7.7: There exist $\gamma, \lambda, \sigma > 0$, such that, for each $i \in \{1, \cdots, N\}$, there is at least one LNARC, referred to as $C_i(\cdot)$, such that, if $C_i(\cdot)$ is interconnected with a dynamic system described by (7.6) with $\rho(k) \in \Omega_i$, for $k \geq k_o$, then

$$\|z(k)\| \leq \gamma\|\xi(k)\| + \lambda e^{-\sigma(k-k_o)}x(k_o),$$

and the dynamics of the closed-loop are asymptotically stable. □

Remark 7.4: The norm, $\|\cdot\|$, in Assumption 7.7 can be, for instance, the ℓ_2-norm of signals, *i.e.*,

$$\|z(k)\|_{\ell_2}^2 := \sum_{m=0}^{k} |z(m)|^2.$$

In such circumstances, γ is the so-called ℓ_2-induced norm of system $CL_{ii}(\cdot)$ $\diamond$

Now the only requirement remaining is that the closed-loop system, for the different possible interconnections, is always distinguishable.

Assumption 7.8: Let $i, j, \ell, m \in \{1, \cdots, N\}$, with $i \neq \ell$ and/or $j \neq m$. Then, the systems $CL_{ij}(\cdot)$ and $CL_{\ell m}(\cdot)$ are absolutely $(X^\star, W)$-input distinguishable in N_{CL} measurements. □

Hence, we are now in conditions of stating the following theorem, which provides performance guarantees for the closed-loop system:

Theorem 7.3. *Consider a plant described by* (7.6) *and that Assumptions 7.4–7.8 are satisfied, where $X^\star$ is defined so that $X_i(y(k), u(k), k) \subseteq X^\star$ for all $k \geq 0$ and $i \in \{1, \cdots, N\}$. Then, the closed-loop system with the MMAC/SVO scheme is asymptotically stable and satisfies, for $k > N_{CL}$,*

$$\|z(k)\| \leq \gamma \|\xi(k)\| + \lambda e^{-\sigma(k - N_{CL})} x(N_{CL}).$$

Proof. Similarly to the proof of Theorem 7.2, we can show that the number of switchings is finite. In particular, due to Assumption 7.8, the maximum number of measurements before the last switching is N_{CL}. Hence, if the plant to be controlled is described by (7.6) with $\rho(k) \in \Omega_{i^\star}$ for $k \geq 0$, the controller $C_{i^\star}$ is selected at time $k^\star \leq N_{\mathrm{CL}}$, which, using Assumption 7.7, concludes the proof. □

7.5.5 TV Plants and TV Bound on the Disturbances

Notwithstanding the fact that the architecture proposed in Section 7.5.2, jointly with the algorithm in Section 7.5.3, can be applied to time-varying plants, they do not allow for the dynamics of the system to *drift* from one region of uncertainty to another. Moreover, the proposed algorithms are not robust to variations or large uncertainty on the bounds of the disturbances. In fact, if the bound on the disturbances is increased from one measurement to another, it may happen that the *correct* model of the plant gets disqualified. Therefore, we now extend the previous results to time-varying plants whose dynamics may not remain in the same region for all times $k \geq 0$, and whose bound on the disturbances may also be time-varying.

In the case of time-varying plants, a model shall never be *disqualified* "forever". Indeed, if the dynamics of the plant drifted at a given time instant, then a previously discarded controller might be the appropriate one to be used from that moment on.[1]

In order to proceed with the development of the algorithm for time-varying plants, we posit the following assumption:

Assumption 7.9: There exists $T_{\min} > 0$ such that, if $\rho(t) \in \Omega_j$, then there exist t_1 and t_2 such that

a) $|t_2 - t_1| \geq T_{\min}$;

b) $t_1 \leq t \leq t_2$;

c) $\rho(\tau) \in \Omega_j$ for all $\tau \in [t_1, t_2]$.

□

Hence, we propose the architecture depicted in Fig. 7.14 as an extension of the MMAC/SVO scheme for time-varying plants. The main idea is to have an SVO, referred to as *Global SVO*, which is able to provide a set-valued state estimate for all the admissible time-varying uncertainties of the plant. Therefore, unless none of the N families of models, assuming that the uncertain parameters are time-varying, is able to describe the dynamics of the actual plant, the Global SVO does never provide an empty set as the set-valued estimate of the state.

In order to account for time-variations of the bound on the disturbances acting upon the plant, we increase the bound on the disturbances used by the SVOs, whenever all the models have been falsified twice in less than a given amount of time. This idea is based on Assumption 7.9. Moreover, although we established that the

[1] A similar line of though was used in the extension of the Stability Overlay (SO) for time-varying plants – see Chapter 4.

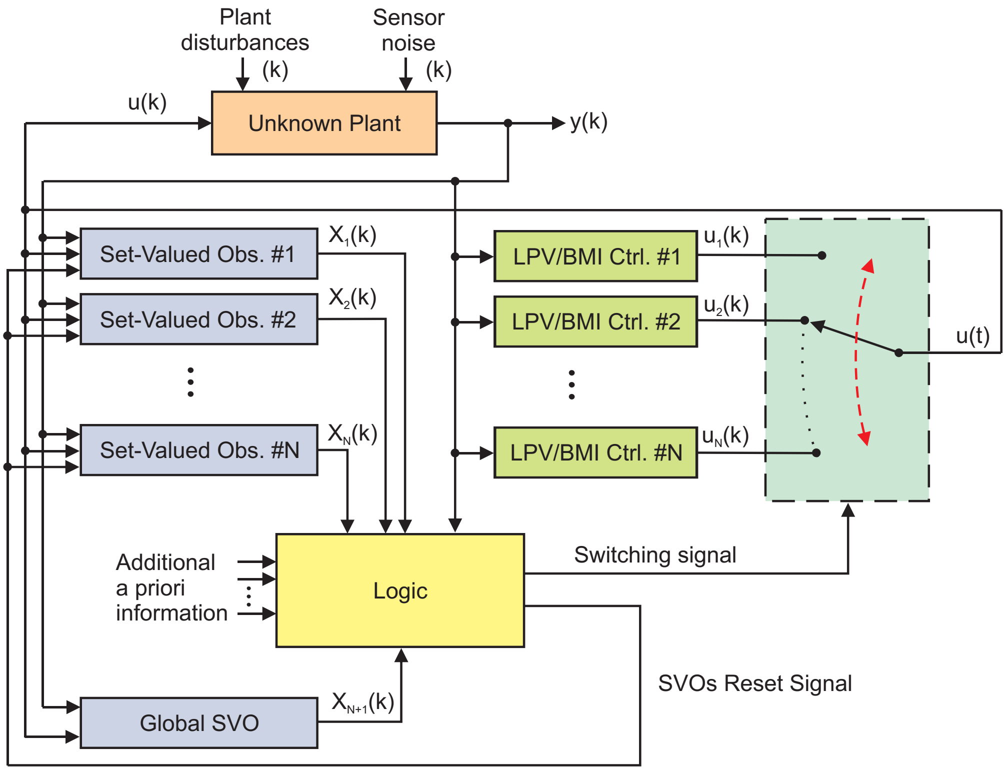


Figure 7.14: MMAC/SVO architecture for time-varying systems.

(tight) bound on the disturbances is unknown, we assume that an (probably very conservative) overbound is known a priori:

Assumption 7.10: The disturbances are bounded by

$$\|d(k)\| \leq \bar{d}, \forall_{k\geq 0},$$

for some known finite constant $\bar{d}$. □

Now suppose that the Global SVO, depicted in Fig. 7.14, is computed assuming a bound on the disturbances given by $\bar{d}$. Then, it is straightforward to conclude that the set-valued state estimate provided by this SVO is never going to be empty.

The algorithm controlling the *Logic* block in Fig. 7.14 is based upon the following reasoning, and is depicted in Fig. 7.15. In the first sampling time, all the SVOs (except the Global SVO) assume that the bound on the disturbances is given by $d_{\max}$, where $d_{\max}$ satisfies

$$d_{\max} \leq \bar{d}.$$

If all but the Global SVO provide empty set-valued state estimates for the plant, it means that none of the N models is able to describe the observed input/output data. Thus, two scenarios have to be considered:

a) the dynamics of the plant have drifted from one region of uncertainty to another;

b) the tentative bound on the disturbances, $d_{\max}$, is smaller than the actual tightest bound on the disturbances.

In order to distinguish between these two scenarios, we resort to Assumption 7.9. If the SVOs are reinitialized twice with the set-valued estimate of the Global SVO, in a time-interval smaller than $T_{\min}$, then we conclude that we have to increase the value of $d_{\max}$. Otherwise, the falsification of the models may be explained by the time-variations of the dynamics of the plant. In both circumstances, all the N SVOs should be reinitialized with the set-valued state estimate of the Global SVO. The remaining parts of the algorithm are similar to the time-invariant case.

In the algorithm depicted in Fig. 7.15, $\Delta T(k)$ stores the amount of time since the estimates of the SVOs have been reinitialized with that of the Global SVO, and γ is a (fixed) parameter, with $\gamma > 1$. Finally, T_s denotes the sampling period of the SVOs, and $\kappa > 1$ is a constant by which each $\overline{\text{RMS}}_i$ is multiplied whenever the algorithm detects that the bound on the magnitude of the disturbances is greater than that used during the design phase. As in the time-invariant case, the RMS of the output is used to potentially accelerate the falsification of the models.

Remark 7.5: If the set-valued state estimate of the Global SVO is bounded, then so do the set-valued state estimates of the remaining SVOs. This is straightforward to conclude from the fact that, at each sampling time, the set-valued state estimate computed by any of the N SVOs is inside the set-valued state estimate of the Global SVO. $\diamond$

Stability Guarantees

Using similar arguments to those of the proof of Theorem 7.2, it is straightforward to prove stability for time-varying plants.

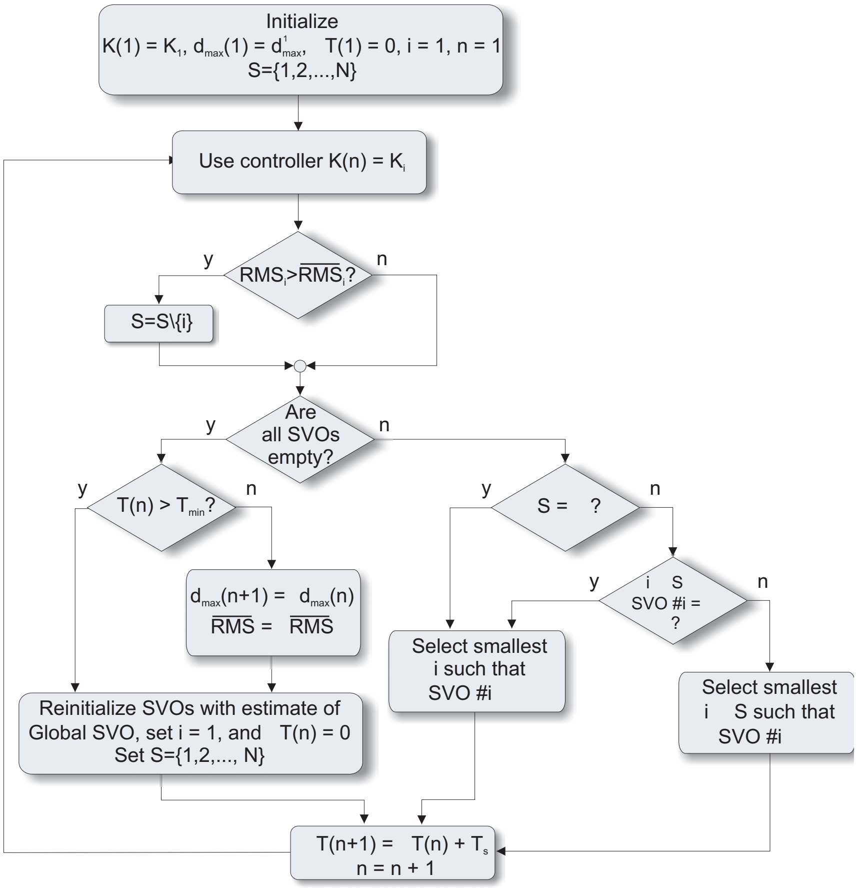


Figure 7.15: Algorithm for the *Logic* block of the MMAC/SVO-RMS architecture for time-varying systems with unknown bound on the disturbances.

Theorem 7.4. *Suppose Assumptions 7.3–7.6, 7.9 and 7.10 are satisfied. Then, the closed-loop system with the MMAC/SVO architecture for time-varying plants is input/output stable, for sufficiently large T_{min}.*

Proof. Using arguments similar to those of the proof of Theorem 7.2 and resorting to the same strategy as in Section 4.5, it is straightforward to show that the MMAC/SVO stabilizes the closed-loop system. □

7.5.6 Simulation: Non-Minimum Phase Plant

We are now going to evaluate the applicability of the MMAC/SVOs scheme through a simple example. Nevertheless, a more detailed evaluation of the proposed adaptive control scheme will be presented in Chapter 8.

Consider the following Non-Minimum Phase (NMP) plant, similar to that examined by other authors of adaptive control studies – see [34, 37]. In particular, let the corresponding transfer function, $G(s)$, be described by

$$G(s) = \frac{1.2\left(\frac{1}{2}s+1\right)\left(-\frac{1}{Z}+1\right)}{\left(\frac{2}{3}s+1\right)\left(\frac{1}{3}s+1\right)\left(\frac{1}{10}s+1\right)} \tag{7.7}$$

where Z is the uncertain non-minimum phase zero, which is assumed to satisfy

$$Z_{\min} := 1 \leq Z \leq 100 =: Z_{\max}.$$

The block diagram of the non-minimum phase plant is depicted in Fig. 7.16, where W_d is described in the sequel. The sensor noise is denoted by $n(\cdot)$ and is a stochastic input, with zero mean and uniform probability distribution.

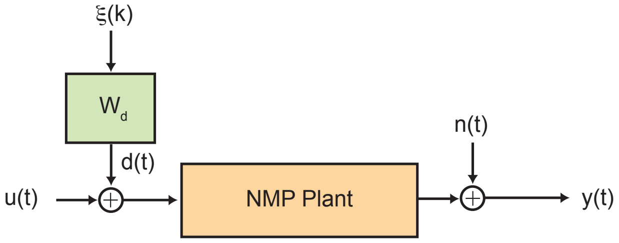


Figure 7.16: Non-minimum phase plant block diagram.

The magnitude Bode plot of the transfer function in (7.7) is depicted in Fig. 7.17.

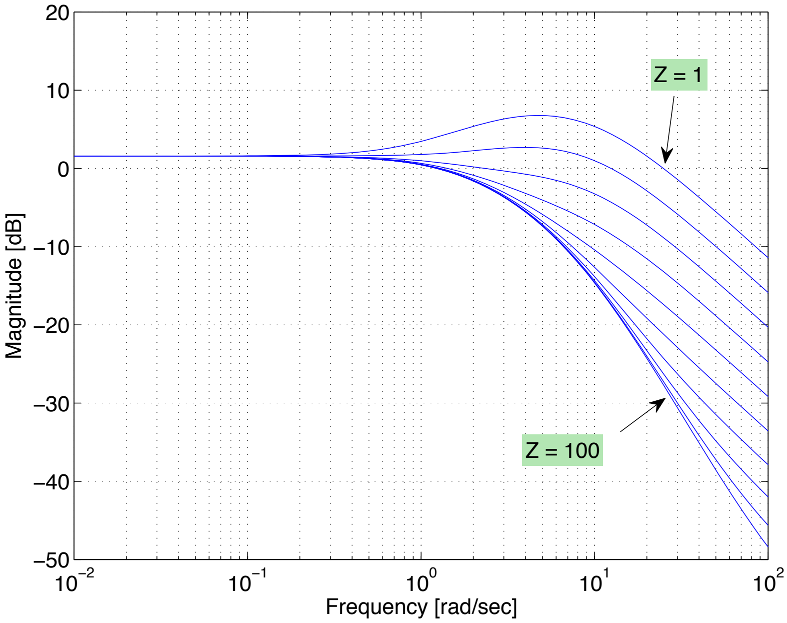


Figure 7.17: Frequency response of the open-loop non-minimum phase plant, for different values of Z.

The plant is subject to a low-frequency disturbance, $d(t)$, obtained by driving a low-pass filter with transfer function $W_d(s)$ described by (7.8), with a uniformly distributed bounded stochastic input, $\xi(t)$, with zero mean.

$$W_d(s) := \frac{d(s)}{\xi(s)} := \frac{\alpha}{s+\alpha} \tag{7.8}$$

The disturbance low-pass filter frequency response is depicted in Fig. 7.18. It should be noticed that the cut-off frequency of this filter is a decade below the smallest value of Z, *i.e.*, $Z = 1$ rad/s. Hence, jointly with the frequency response of Fig. 7.17, we conclude that these disturbances hinder the identification process.

Therefore, the dynamics of the plant can be described in state-space form as

$$\left\{ \begin{array}{rcl} \dot{x}(t) & = & Ax(t) + Bu(t) + L\xi(t), \\ y(t) & = & Cx(t) + n(t), \end{array} \right. \tag{7.9}$$

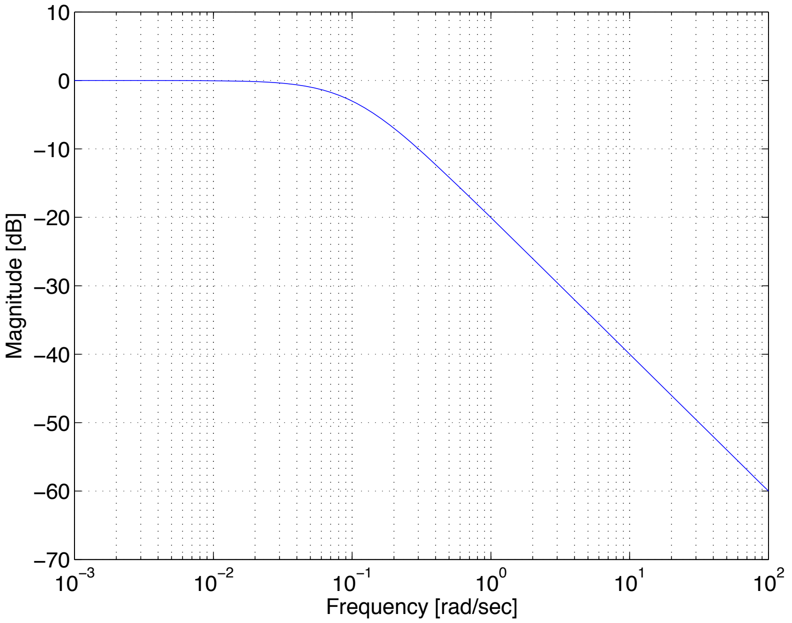


Figure 7.18: Frequency response of the low-pass filter of the disturbances, with transfer function $W_d(s)$.

where

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -45 & -49.5 & -14.5 & 1 \\ 0 & 0 & 0 & -0.1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad L = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0.1 \end{bmatrix},$$
$$C = \begin{bmatrix} 54 & 27 - \frac{54}{Z} & -\frac{27}{Z} & 0 \end{bmatrix}.$$

The output equation in (7.9) can, therefore, be written as

$$y(t) = \bar{C}x(t) + \Delta\tilde{C}x(t) + n(t), \tag{7.10}$$

where

$$\begin{aligned} \bar{C} &= \begin{bmatrix} 54 & 27 & 0 & 0 \end{bmatrix} + \bar{c}\begin{bmatrix} 0 & -54 & -27 & 0 \end{bmatrix}, \quad \bar{c} = \tfrac{Z_{\min}{}^{-1} + Z_{\max}{}^{-1}}{2}, \\ \tilde{C} &= \tilde{c}\begin{bmatrix} 0 & -54 & -27 & 0 \end{bmatrix}, \quad \tilde{c} = \tfrac{Z_{\min}{}^{-1} - Z_{\max}{}^{-1}}{2}, \end{aligned}$$

and where Δ with $|\Delta| \leq 1$ denotes the uncertainty related to the non-minimum phase zero.

Discretizing the dynamics in (7.9) and using the output equation in (7.10), a description of the plant is obtained that is compatible with the structure in Section 6.3.2.

We further considered a non-noisy output of the plant given by

$$\tilde{y}(t) = \bar{C}x(t) + \Delta\tilde{C}x(t),$$

and a performance variable, $z(\cdot)$, described by

$$Z(s) = W_p(s)Y(s),$$

where

$$W_p(s) = A_p\frac{\alpha}{s+\alpha}, \quad \alpha = 0.1 \text{ rad/s.} \tag{7.11}$$

The performance variable is used to evaluate the quality of the disturbance-rejection controllers. The corner frequency of this performance weight is equal to that of the disturbance filter described by $W_d(s)$, which indicates that our goal is to attain high levels of disturbance-rejection in this frequency range.

For the design of the LNARCs, we consider a measurement noise weight of $W_n = 10^{-3}$, while for the control weight we used a filter described by

$$W_u(s) = \frac{10(s+10)}{s+1000}.$$

Finally, we also considered, for controller design purposes, an input uncertain time-delay of, at most, 0.05 secs. This effect can be modeled by a complex uncertainty, denoted by $\Delta_{un}(s)$, and by a weighting function [77]. In this case, we defined

$$W_{un}(s) = \frac{2.45s}{s+40}.$$

Remark 7.6: We stress that, for the sake of simplicity, time-delays are not considered, in simulation and in the design of the SVOs. ◇

Remark 7.7: As mentioned in [160], the presence of low-frequency non-minimum phase zeros constrains the attainable performance of the closed-loop, at low-frequencies. Indeed, in the present case, the achievable low-frequency disturbance-rejection for $Z = Z_{\min} = 1$ rad/sec should be larger than the one achieved for $Z = Z_{\max} = 100$

rad/sec. Nevertheless, this is not a shortcoming of the adaptive control method at hand, but rather a fundamental limitation of this non-minimum phase plant. ◇

Following the design procedure in [34], the large uncertainty region for the non-minimum phase zero is divided into the subintervals

$$\begin{cases} Z_1 & := & [4.10,\ 100.0], \\ Z_2 & := & [2.38,\ 4.10], \\ Z_3 & := & [1.34,\ 2.38], \\ Z_4 & := & [1.00,\ 1.34], \end{cases}$$

so that the attained closed-loop performance is at least 30% of that obtained had we known the exact value of Z. The values of A_p in (7.11), obtained by designing a mixed-μ controller for each of the 4 regions of uncertainty, are summarized in Table 7.3. These mixed-μ controllers provide robust stability and performance guarantees for each region of uncertainty, as long as the plant is time-invariant.

Table 7.3: Values of A_p and uncertainty intervals for each LNARC.

Compensator	Uncertainty Interval	A_p
LNARC #1	$Z_1 := [4.10,\ 100.0]$	298.5
LNARC #2	$Z_2 := [2.38,\ 4.10]$	107.1
LNARC #3	$Z_3 := [1.34,\ 2.38]$	42.6
LNARC #4	$Z_4 := [1.00,\ 1.34]$	26.5

As stressed in Remark 7.7, there are performance limitations that any MMAC scheme is going to inherit from its local non-adaptive controllers. Indeed, as depicted in Fig. 7.19, larger values of Z lead to higher attenuations of the disturbances. Moreover, the controllers synthesized to values of Z closer to 1 rad/sec, guarantee robust performance for a smaller region of uncertainty than the ones designed for values of the non-minimum phase zero closer to 100 rad/sec.

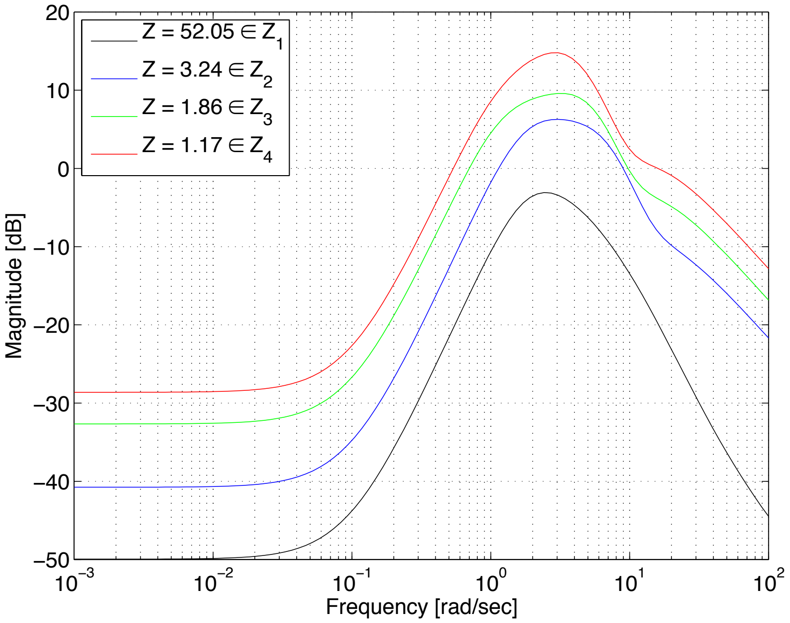


Figure 7.19: Frequency response of the closed-loop system with the corresponding LNARC controller, from the disturbances, $d(\cdot)$, to the output variable, $y(\cdot)$, for different values of the non-minimum phase zero.

Model-Mismatch Instability

As described in [34], the feedback interconnection of the plant with $Z \in Z_i$, with LNARC $\#j$, for $i \neq j$, does not necessarily lead to an unstable closed-loop system. Hence, we summarize, in Table 7.4, the stability properties of the closed-loop for the different possible interconnections.

As an example of interpretation of Table 7.4, if $Z \in Z_1$, all of the 4 LNARCs are able to stabilize the plant, though LNARC #1 is the one that yields the highest levels in terms of performance. Nevertheless, if $Z \in Z_4$, controller #1 leads to an unstable closed-loop system. These phenomena can be interpreted from an intuitive point of view, by understanding that, typically, controllers that lead to higher levels of performance at a given frequency range, have larger loop gains at those frequencies. Hence, LNARC #1 has a low-frequency gain larger than LNARC #4, which leads to a higher level of performance at that frequency range, whenever there does not exist model-mismatch, while making the closed-loop system unstable if the model-mismatch is significant.

Table 7.4: Model-mismatch stability properties of the plant/controller interconnection.

	LNARC			
	#1	#2	#3	#4
$Z \in Z_1$	S	S	S	S
$Z \in Z_2$	CU	S	S	S
$Z \in Z_3$	U	CU	S	S
$Z \in Z_4$	U	U	CU	S

S: stable

U: unstable

CU: conditionally stable, *i.e.*, stable for some values of Z.

This illustrates the benefits of using adaptive control in this case. By properly identifying the region of uncertainty of Z, the performance of the closed-loop system can be significantly increased.

Simulations

We use a sampling time of $T_c = 1$ ms for the controllers, and a sampling time of $T_s = 20$ ms for the SVOs. The discretization of the plant is done based upon the methodology previously described.

In the following simulations, we consider that the initial LNARC selected is #1, that is, the mixed-μ controller designed to guarantee robust performance for the zero uncertainty region Z_1. According to the algorithm introduced in Section 7.5.3, if SVO #1 fails, then we switch to LNARC #2. If SVO #2 also fails, we switch to LNARC #3, and so on.

The disturbances are bounded by $|d(\cdot)| \leq d_{\max} := 10^4$, and the measurement noise is constrained by $|n(\cdot)| \leq 1$. The initial state of the plant is given by

$$x(0) = x_o := \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}},$$

although the only information considered available for the SVOs is that

$$x_o \in X_o := \{x : |x| \leq 20\}.$$

Scenario #1: $Z \in Z_1$

Let us start by considering that $Z = 100$ rad/sec, and thus $Z \in Z_1$. Under these circumstances, the *correct* controller is in fact the first one to be interconnected with the plant, according to the algorithm suggested in Section 7.5.3. Given that the model in (7.7), with $Z \in Z_1$, is a valid description of the dynamics of the plant, this region of uncertainty does never get falsified. As a result, the selected controller is always LNARC #1. The output of the plant in this case is depicted in Fig. 7.20, for a randomly selected Monte-Carlo run.

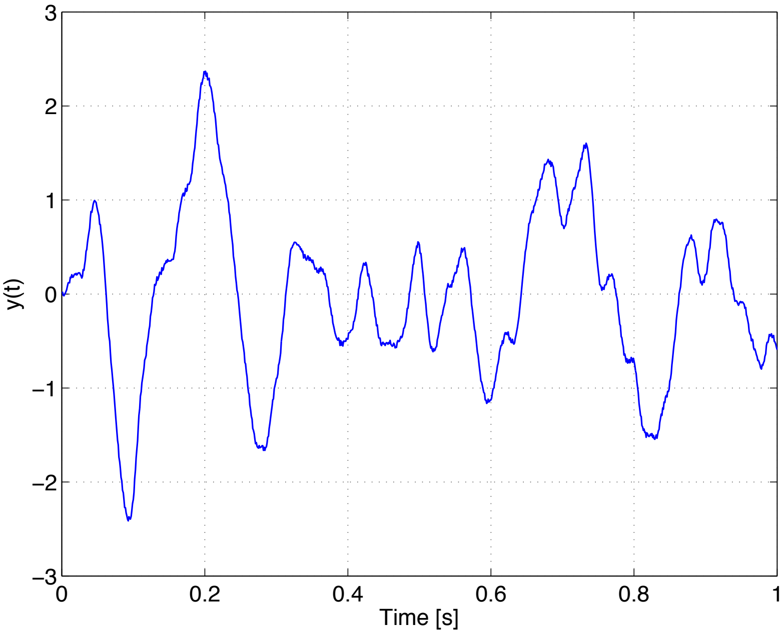


Figure 7.20: Output of the non-minimum phase plant, for Scenario #1, *i.e.*, $Z =$ 100 rad/s $\in Z_1$.

Since we are using a controller that attains a high level of performance, the output of the NMP plant is small, when compared to the maximum allowable amplitude of the disturbances. Consequently, the problem of *distinguishing* between the 4 possible dynamic models becomes harder – see Section 5.3.1 and [148]. This justifies the fact that none of the SVOs, in the 10 Monte-Carlo simulations performed, had the ability to invalidate a model in this scenario.

Scenario #2: $Z \in Z_4$

Now, consider that the uncertain non-minimum phase zero, Z, is on the opposite edge of the uncertainty interval, *i.e.*, $Z = 1$ rad/sec. Under this scenario, LNARC #4 is the *best* controller, from the set of eligible ones. Moreover, the first controller selected is LNARC #1, which, according to Table 7.4, is a destabilizing controller. As shown in Fig. 7.21, this incorrect selection of the controller leads to a transient at the beginning of the simulation.

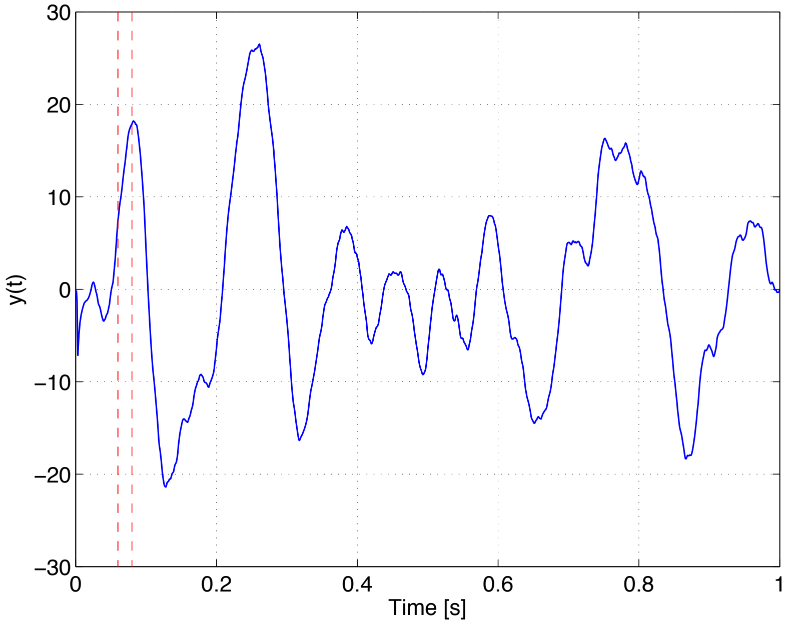


Figure 7.21: Output of the non-minimum phase plant, for Scenario #2, *i.e.*, $Z = 1$ rad/s $\in Z_4$.

Nevertheless, after three measurements, the 2 first regions of uncertainty are invalidated by the SVOs, at once. Thus, at time $t = 60$ ms, LNARC #3 is interconnected with the plant. One measurement later, this region of uncertainty (region #3) is also falsified. Therefore, at time $t = 80$ ms, LNARC #4 is selected. Since the model in (7.7), with $Z \in Z_4$, is a valid description of the dynamics of the plant, this region of uncertainty does never get falsified. Therefore, the *best* controller is eventually selected, providing high levels of closed-loop performance.

It should be noticed that, contrary to the previous scenario, all but one model are invalidated by the SVOs. This can also be explained by the considerations in Section 5.3.1 and [148], which state that, under mild conditions, large amplitude signals promote the distinguishability dynamic models.

Moreover, this leads to the conclusion that the controller selection algorithm in Section 7.5.3 is indeed the adequate heuristics to select the LNARCs, if our goal is to switch as fast as possible to the *best* controller. To see this, consider that the initial controller is LNARC #4. In this case, if $Z \in Z_4$, then the selected controlled would be the adequate one. However, if, for instance, $Z \in Z_1$, then the closed-loop system would still be stable, as described in Table 7.4. Thus, this would hinder the distinguishability of the eligible dynamic models. A similar reasoning can be applied to other selections of initial controllers.

7.5.7 Simulation: MSD Plant

As a second example of the applicability of the MMAC/SVOs scheme, we now consider the Mass-Spring-Dashpot (MSD) plant depicted in Fig. 7.22, where the control input is collocated with the disturbance.

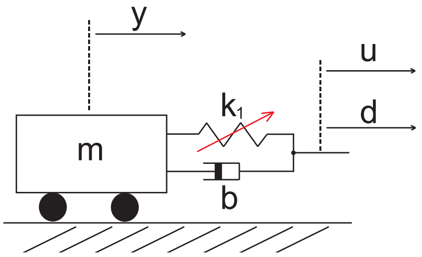


Figure 7.22: Mass-Spring-Dashpot (MSD) plant. The control input is the position of the right part of the spring and dashpot.

The continuous-time model of the MSD plant is derived in the sequel. We denote $y(t)$ as the position of the mass and $u(t)$ the control input, that is, the position of the right-end of the spring and dashpot. The disturbances are not considered, for now, since they act upon the plant in the same way the control input does. Hence, we have

$$m\ddot{y}(t) = k_1 \left(u(t) - y(t)\right) + b \left(\dot{u}(t) - \dot{y}(t)\right). \tag{7.12}$$

where

$$\begin{aligned} m &= 1 \text{ kg}, \\ b &= 0.1 \text{ N/(m/s)}, \\ |d| &\leq 1, \\ |n| &\leq 0.1, \\ k_1 &\in \left[\bar{k}_1 - \tilde{k}_1, \quad \bar{k}_1 + \tilde{k}_1\right] =: K \text{ (N/m)}. \end{aligned}$$

Therefore, using the Laplace transform of $y(t)$ and $u(t)$, denoted by $Y(s)$ and $U(s)$, respectively, (7.12) can be rewritten as

$$ms^2 Y(s) = k_1 \left(U(s) - Y(s)\right) + b \left(sU(s) - sY(s)\right). \tag{7.13}$$

Therefore, (7.12) can be written in state-space form as

$$\begin{cases} \dot{x}(t) &= Ax(t) + Bu(t) + Ld(t), \\ y(t) &= Cx(t) + n(t), \end{cases}$$

where $n(t)$ denotes the measurements noise and

$$\begin{aligned} A &= \begin{bmatrix} -b & -k_1 \\ 1 & 0 \end{bmatrix}, \\ B &= L = \begin{bmatrix} 1 & 0 \end{bmatrix}^T, \\ C &= \begin{bmatrix} b & k_1 \end{bmatrix}. \end{aligned}$$

We use a sampling time of $T_c = 1$ ms for the controllers, and a sampling time of $T_s = 500$ ms for the SVOs. The discretization of the plant is done based upon the methodology previously described.

The uncertainty region considered is $K = [0.25, 9.25]$ N/m. This (large) region was divided into the following 3 regions, K_i, for $i = \{1, 2, 3\}$, and, for each region, a mixed-μ controller that is able to yield a certain level of performance (in this case, 70% of the performance we would obtain, had we known the actual value of the spring constant) was synthesized:

$$\begin{aligned} K_1 &:= [0.25,\, 5.0] \ \text{N/m}, \\ K_2 &:= [5.0,\, 7.0] \ \text{N/m}, \\ K_3 &:= [7.0,\, 9.25] \ \text{N/m}. \end{aligned}$$

In the following simulations, we consider that the initial LNARC selected is $C_1(\cdot)$, that is, the mixed-μ controller designed to guarantee robust performance for the uncertainty region K_1. According to the algorithm introduced in Section 7.5.3, if SVO #1 fails, then we switch to controller $C_2(\cdot)$. If SVO #2 also fails, we switch to controller $C_3(\cdot)$. Notice that at least one of three SVOs cannot fail.

Scenario #1: MMAC/SVO

In this first scenario, we use the MMAC/SVO algorithm described in Fig. 7.12, *i.e.*, we rely exclusively on the SVOs to discard dynamic models of the plant, regardless of the achieved RMS performance.

The first simulation was obtained by using $k_1 \in K_1$. In particular, let

$$k_1 = 1 \ \text{N/m}.$$

Since we start with controller $C_1(\cdot)$, there is no need for switching. The result is depicted in Fig. 7.23.

For the second simulation run, we used

$$k_1 = 6 \ \text{N/m} \ \in K_2.$$

The initial controller is $C_1(\cdot)$, *i.e.*, we assume that the uncertain plant belongs to K_1. However, after 5 measurements, SVO #1 is not able to explain the measured output, and hence we switch to controller $C_2(\cdot)$. Since SVO #2 does not fail, we continue using controller $C_2(\cdot)$, as depicted in Fig. 7.24.

For the last simulation with the MMAC/SVO algorithm, we used a model with

$$k_1 = 9 \ \text{N/m} \ \in K_3.$$

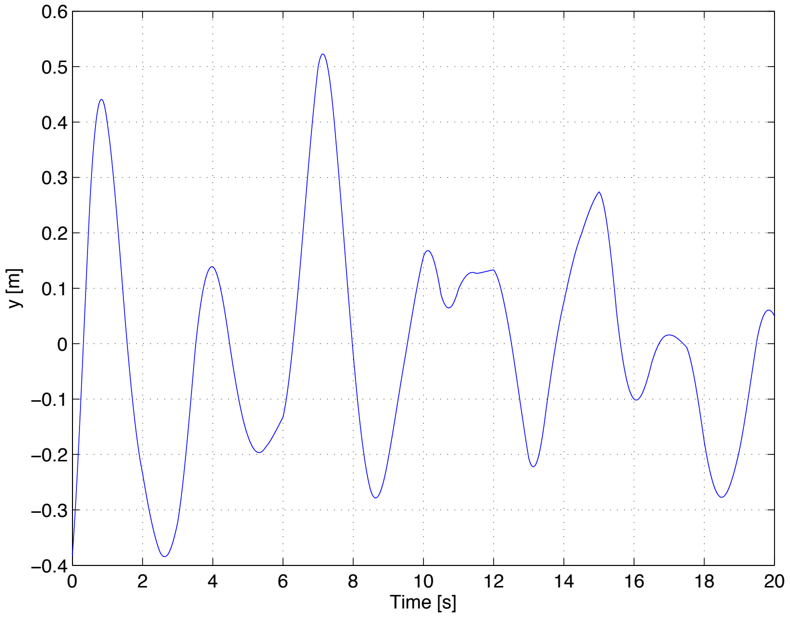


Figure 7.23: Output of the closed-loop for the MSD-plant with $k_1 \in K_1$, for the MMAC/SVO.

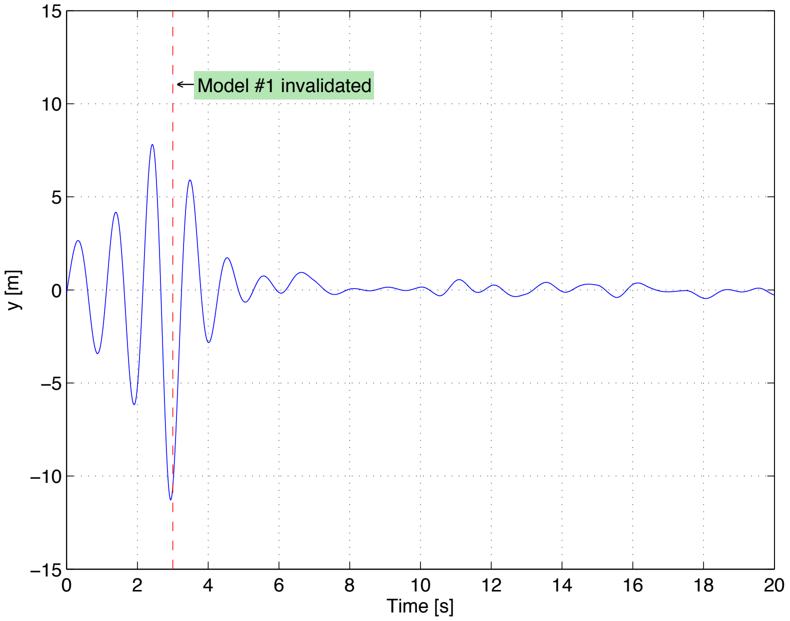


Figure 7.24: Output of the closed-loop for the MSD-plant with $k_1 \in K_2$, for the MMAC/SVO. The red dashed line indicates the time instant at which the SVO #1 failed, and hence the logic switched to LNARC #2.

As depicted in Fig. 7.25, SVOs #1 and #2 fail at times $t = 1.5$ s and $t = 2$ s, respectively. Although $C_2(\cdot)$ is able to stabilize the plant, the algorithm can still decide to switch to controller $C_3(\cdot)$, since the SVO #2 fails.

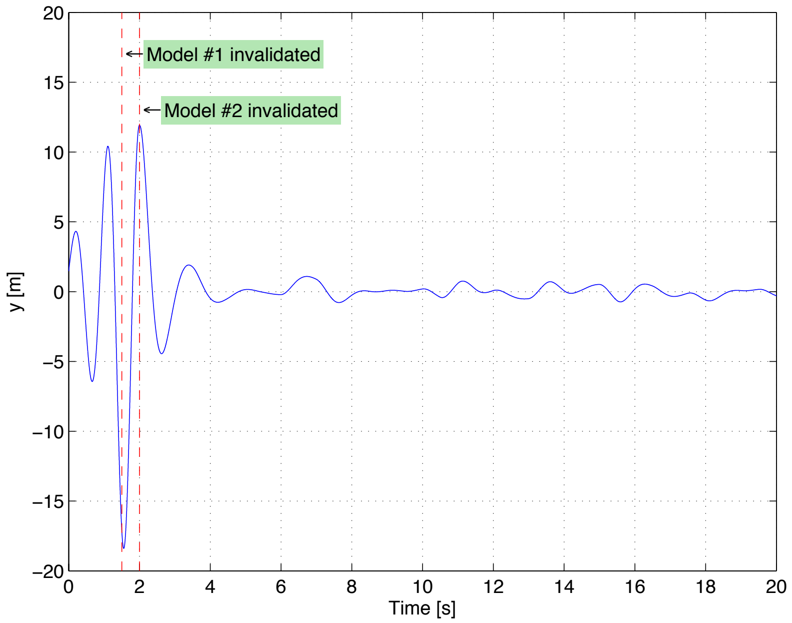


Figure 7.25: Output of the closed-loop for the MSD-plant with $k_1 \in K_3$, for the MMAC/SVO. The red dashed lines indicate the time instants at which the SVO #1 and #2 failed, and hence the logic switched to controller #3.

It should be noted that, the smaller the amplitudes of the input and output signals, the harder it is, in general, to discard an uncertainty region, because the models become less distinguishable. However, we stress that, for the values of the spring constant used, all the controllers were able to stabilize the plant. Nevertheless, the algorithm always picked the controller that provided the performance level we were expecting.

Remark 7.8: For this plant, we have to solve around 10 to 30 linear programs, up to 20 times per iteration, which requires approximately 500 ms or less in a *Core 2 Duo Pentium*$^{\mathrm{TM}}$ processor at 2.0 Ghz. $\diamond$

Scenario #2: MMAC/SVO-RMS

As previously stressed, the MMAC/SVO can be seen as a *worst case* approach, in the sense that models are not invalidated unless the observed input/output sequences cannot be explained by the dynamics of the closed-loop system. In order to accelerate the decision subsystem of the proposed adaptive control scheme, an algorithm was proposed – see Fig. 7.15 – which takes into account the RMS of the closed-loop system. Hence, the following simulations illustrate the goodness of this approach, when compared to the "classical" MMAC/SVO algorithm in Fig. 7.12.

As in the previous scenario, the first simulation was obtained by using

$$k_1 = 1 \text{ N/m } \in K_1.$$

Once again, since we start with controller $C_1(\cdot)$, there is no need for switching, and thus we obtain the results depicted in Fig. 7.23.

For the second simulation run, we considered that

$$k_1 = 6 \text{ N/m } \in K_2.$$

While with the MMAC/SVO we required 5 measurements to invalidate model #1, with the MMAC/SVO-RMS this model is invalidated in a single measurement. Therefore, at time $t = 0.5$ s we switch to controller $C_2(\cdot)$. Since SVO #2 does not fail, we continue using controller $C_2(\cdot)$, as depicted in Fig. 7.26. It should be noted that, since now the amplitude of the output signal is considerably smaller than the one obtained with the classical MMAC/SVO algorithm, the invalidation of dynamic models through the SVOs gets hindered. In particular, in this case, we are not able to invalidate model #2 using the SVOs.

In comparison with the results in Fig. 7.24, the results in Fig. 7.26 clearly show a pronounced increase in terms of performance. Indeed, the transients observed are considerably smaller, since the time required to switch to the correct controller is also smaller, and by a factor of 5.

For the sake of completeness, suppose that

$$k_1 = 9 \text{ N/m } \in K_3.$$

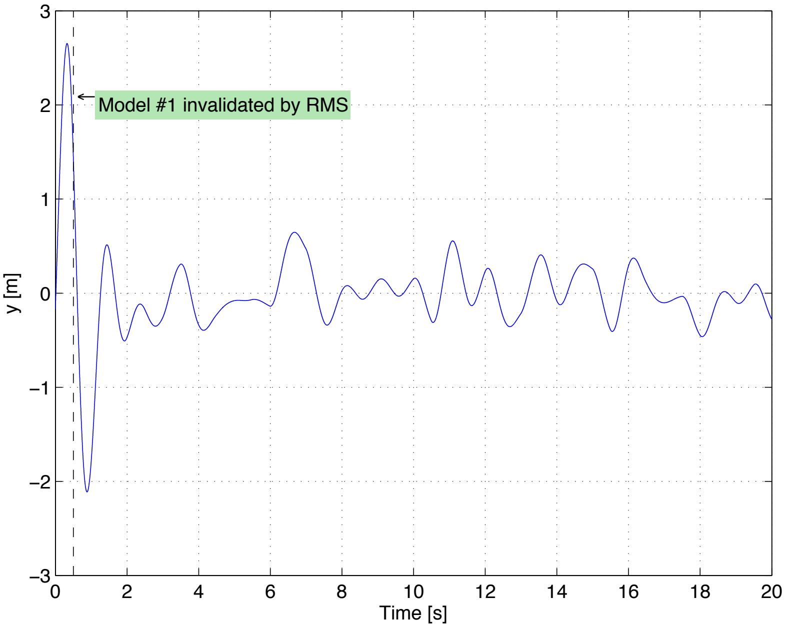


Figure 7.26: Output of the closed-loop for the MSD-plant with $k_1 \in K_2$, for the MMAC/SVO-RMS. The black dashed line indicates the time instant at which model #1 was invalidated due to the RMS of the output, and hence the logic switched to LNARC #2.

As depicted in Fig. 7.27, SVO #1 fails at time $t = 1.5$ s. However, even before that, models #1 and #2 are invalidated through RMS considerations, at times $t = 0.5$ s and $t = 1$ s, respectively. Therefore, by the time SVO #1 invalidates the corresponding dynamic model, the selected controller is already LNARC #3. The benefits in terms of reduced transients, due to the faster response of the MMAC/SVO-RMS algorithm, can be observed by comparing the results depicted in Fig. 7.25 with those in Fig. 7.27.

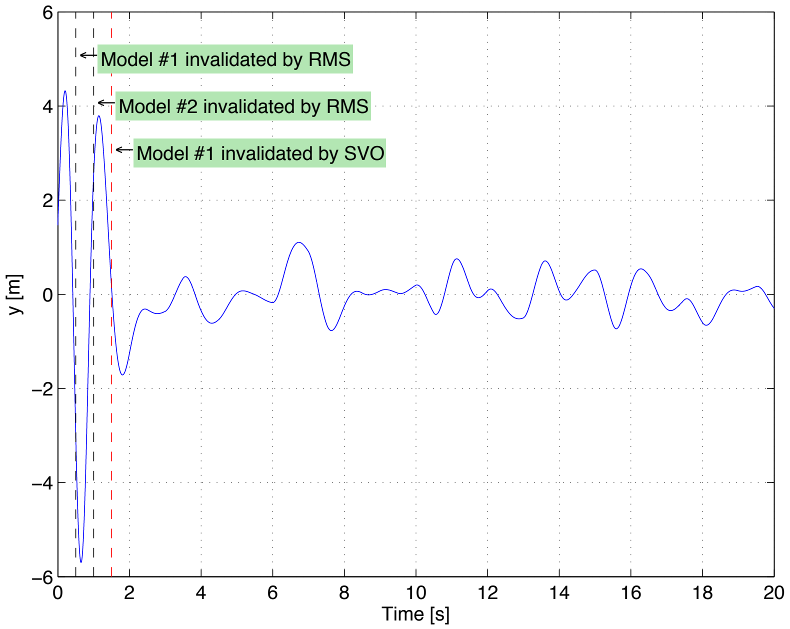


Figure 7.27: Output of the closed-loop for the MSD-plant with $k_1 \in K_3$, for the MMAC/SVO-RMS. The red dashed lines indicate the time instants at which the SVO #1 and #2 failed, and hence the logic switched to controller #3.

A more detailed comparison of the MMAC/SVO with the MMAC/SVO-RMS will be presented in Chapter 8, by using a considerably harder control problem.

7.6 Application to FDI

7.6.1 Introduction

The field of Fault Detection and Isolation (FDI) has been studied since the early 70's [161], and several techniques have, since then, been applied to different types of systems. An FDI device is key in several applications and, in particular, in those that are *safety critical*. Common examples of systems equipped with FDI devices include aircrafts and a wide range of industrial processes such as the ones described in the following references – [128, 131, 134, 136–139, 162, 163]. An FDI system must be able to bear with different types of faults in sensors and/or actuators, which can occur abruptly or slowly in time. Moreover, model uncertainty (such as unmodeled dynamics) and disturbances must never be interpreted as faults. Notwithstanding the hundreds (or maybe thousands!) of papers in the literature concerning this topic, there are still some open questions related to the performance guarantees provided by these devices.

An active deterministic model-based Fault Detection (FD) system (see [137] for a description of the typical FD classes available in the literature) is usually composed of two parts: a filter – see Fig. 7.28 – that generates residuals, which should be *large* under faulty environments; and a decision threshold, which is used to decide whether a fault is present or not – see [127, 129, 136, 137, 161, 164–168] and references therein. The isolation of the fault can, in some cases, be done using a similar approach, *i.e.*, by designing filters for families of faults, and identifying the most likely fault as that associated to the filter with the smallest residuals.

As stressed in Chapter 5, the main idea in such architectures stems from the design of filters that are more sensitive to faults than to disturbances and model uncertainty. This can be achieved, for instance, by using geometric considerations regarding the plant, as in [126–129], or by optimizing a particular norm minimization objective, such as the $\mathcal{H}_\infty$- or l_1-norm – see[130–134]. The later approach provides, in general, important robustness properties, as stressed in [130, 135–137], by explicitly accounting for model uncertainty.

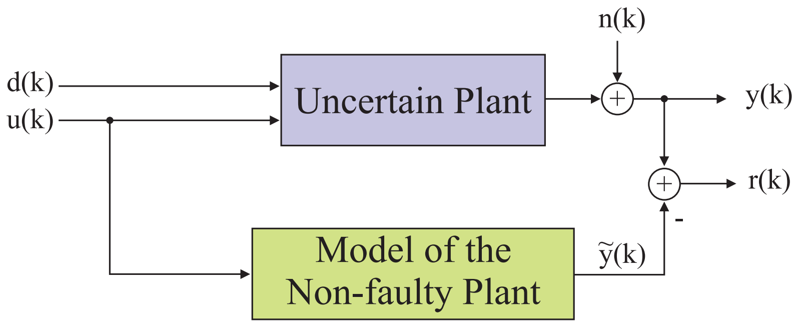


Figure 7.28: Residual generation in a classical fault detection (FD) architecture.

As a caveat, these methodologies are, in general, conservative or can only be applied to a restrict class of systems. Moreover, the thresholds used to declare a fault are typically time-varying and highly dependent on the model uncertainty and on the amplitude of the exogenous disturbances and measurement noise.

The FDI strategy proposed in this section uses a different philosophy. Instead of identifying the most likely model of the faulty plant, we discard models that are not compatible with the observations. As shown in the sequel, this method guarantees that there will not be false alarms, as long as the model of the non-faulty plant remains valid. Moreover, we do not need to compute the decision threshold used to declare whether or not a fault has occurred. To this end, we resort to the model falsification technique introduced in this chapter. In addition, another advantage of the SVO-based methodology presented herein stems from the fact that it is able to deal with linear time-varying uncertain plants.

7.6.2 Types of Faults

We focus our attention in three broad classes of failures that can be found in the literature, namely system dynamics failures, actuator failures, and sensor failures. We present, in the sequel, an outline of the description of each of the aforementioned classes of failures. Moreover, we recall that it is assumed that the model of the plant, whether it is faulty or not, can be represented by

$$\left\{\begin{array}{rcl} x(k+1) & = & A\big(\rho(k)\big)x(k) + B\big(\rho(k)\big)u(k) + L\big(\rho(k)\big)d(k), \\ y(k) & = & C\big(\rho(k)\big)x(k) + N\big(\rho(k)\big)n(k). \end{array}\right. \tag{7.14}$$

System dynamics failures

System dynamics failures can be used to describe significant changes in the model of the plant that can be in general interpreted as changes in one or more elements of matrix A of the dynamic system in (7.14). These changes can be modeled by the parametric uncertainties in (7.14) and, thus, we can use SVOs to detect and isolate such a class of faults, as described in the sequel.

Typical examples of such failures include icing and broken surfaces in aircrafts [137].

Actuators failures

Another important type of failures occurs in the actuators. Such faults can, in general, be described by changes in matrix B in (7.14), and/or changes in the input vectors, $u(\cdot)$. Mathematically, this can be described by

$$\left\{\begin{array}{rcl} x(k+1) & = & A(k)x(k) + A_\Delta(k)x(k) + L_d(k)d(k) + B(k)u(k) + M(k)m(k), \\ y(k) & = & C(k)x(k) + n(k), \end{array}\right. \tag{7.15}$$

where $M(k)m(k)$ accounts for the fault. It should be noticed that this model can describe a broad range of actuator faults, as summarized in Table 7.5 for a single control input plant – see [137] and references therein for further details.

The model described by (7.15) is clearly compatible with (7.14) and, hence, the SVOs can also be used to detect and isolate actuator faults.

Table 7.5: Modeling of common actuator faults.

Type of Fault	Modeling
Hard-over (Saturation)	$M(k) = B(k)$ and $m(k) = -u(k) + u_{\text{sat}}$
Loss-of-Effectiveness	$M(k) = B(k)$ and $m(k) = -u(k) + \alpha u(k), \alpha \in \mathbb{R}$
Floating	$M(k) = B(k)$ and $m(k) = -u(k)$
Bias (Stuck)	$M(k) = B(k)$ and $m(k) = -u(k) + \beta, \beta \in \mathbb{R}$
Noise	$M(k) = B(k)$ and $m(k) =$ noise

Sensors failures

Finally, sensors failures can also be treated in a similar manner. Consider the following description of the plant,

$$\begin{cases} x(k+1) &= A(k)x(k) + A_{\Delta}(k)x(k) + L_d(k)d(k) + B(k)u(k), \\ y(k) &= C(k)x(k) + n(k) + Q(k)q(k), \end{cases} \tag{7.16}$$

where $Q(k)q(k)$ accounts for the sensor fault. Table 7.6 summarizes the most common sensor faults and corresponding models.

Table 7.6: Modeling of common sensor faults.

Type of Fault	Modeling
Dead sensor	$Q(k) = C(k)$ and $q(k) = -x(k)$
Loss-of-Effectiveness	$Q(k) = C(k)$ and $q(k) = -\alpha x(k)$, $\alpha > 0$
Biased sensor	$Q(k) = C(k)$ and $q(k) = \beta$, $\beta \in \mathbb{R}$
Random drift	$Q(k) = C(k)$ and $q(k) =$ random variable

Similarly to what happened in the previous cases, the model described by (7.16) is also compatible with (7.14) and, therefore, we can use the SVOs for fault detection and isolation of sensor failures.

7.6.3 FD/SVO Architecture

In this subsection, an SVO-based method to detect faults is introduced. Since a fault can be defined as a deviation of the plant dynamics from the nominal model, an SVO can be used to detect such a mismatch between the predicted and the actual output of the system. As further stressed in the sequel, this nominal model of the plant must account for unmodeled dynamics, exogenous disturbances and measurement noise.

We use the following proposition to detect faulty behaviors of plants that are modeled by systems represented by (7.14).

Proposition 7.1. *Consider a non-faulty plant described by* (7.14) *and a corresponding SVO, as introduced in Chapter 6. Then, if $\hat{X}(k) = \emptyset$ for some $k \geq 0$, a fault has occurred at some time k_0, where $k_0 \leq k$.*

Proof. The results is straightforward from Lemma 7.1. □

Using Proposition 7.1, the architecture depicted in Fig. 7.29 can be used to tackle the problem of fault detection for discrete-time linear time-varying plants. Notice that the FD filter is composed of an SVO and a *Logic* block, which decides whether a fault is diagnosed or not, according to the emptiness or not of the set-valued estimate of the state at each sampling time.

The sufficient condition in Proposition 7.1 guarantees that there will be no false alarms, unless model (7.14) does not properly describe the nominal plant. It does not, however, provide any guarantees in terms of fault detection, after a certain fault has occurred in the actual system.

In order to guarantee the detection of faults, we pose the following assumption. Let S_1 denote the dynamic model of the non-faulty plant, and S_2 represent a dynamic model of the plant with a given fault, f.

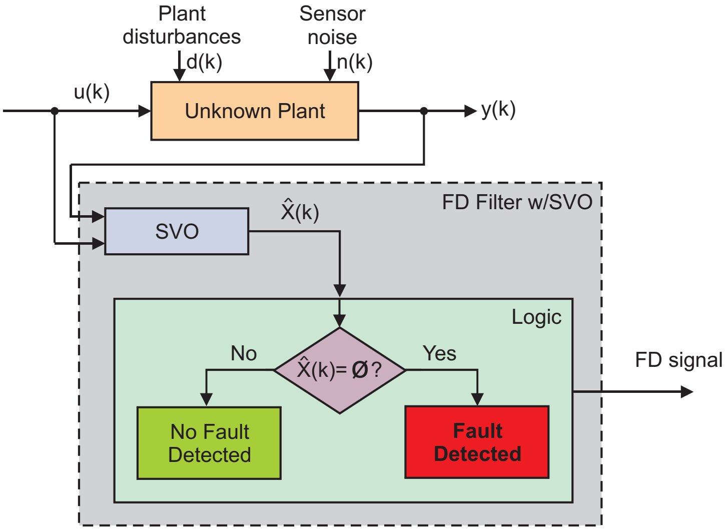


Figure 7.29: Fault Detection (FD) architecture for uncertain plants using a Set-Valued Observer (SVO).

Assumption 7.11: S_1 and S_2 are absolutely (X_o, U, W)-input distinguishable in N sampling times. □

The following result is straightforward to obtain from Theorem 7.1.

Proposition 7.2. *Suppose that Assumptions 7.2 and 7.11 are satisfied. Then, the fault f is detected in at most N iterations after it occurs.*

Hence, using this approach, we have a theoretical tool which provides us, a priori, guarantees of fault detection, while guaranteeing that there are not going to occur any false alarms, under certain assumptions. However, there are also some shortcomings related to this approach:

a) A false alarm can occur if the model described by (7.14) is not a good approximation of the actual dynamics of the plant.

b) Assumption 7.11 may not be satisfied in practice in many examples, although the probability of fault detection may be close to 100%. Notice that this happens because we are using a *worst-case* type of approach.

7.6.4 FDI/SVO Architectures

The fault isolation techniques available in the literature aim to identify a very precise faulty behavior, after a general fault is detected. This means that, unlike an FD filter, an FDI filter should not only be able to detect a faulty behavior of a plant, but also to provide information regarding its location. In particular, FDI filters should be able to separate between the three broad types of failures enumerated in Section 7.6.2.

The proposed SVO-based technique is also suitable for fault isolation, as long as the corresponding model of the fault is considered during the design of the SVOs. The main idea in this case is to resort to model invalidation, as follows. We recover the example of Section 5.2 to illustrate the use of model falsification for fault detection and isolation. Suppose that there are three possible faulty models, $M\#1$, $M\#2$ and $M\#3$, for a given plant, plus a nominal model, $M\#4$. We are interested in deciding which model (if any) is able to justify the input/output data that we are obtaining from the sensors and actuators' command. Therefore, assume that, at a given initial time, t_o, all the four models are plausible, as depicted in Fig. 7.30. Further suppose that, at time t_1, model $M\#4$ is invalidated, *i.e.*, the sensors readings cannot be explained by model $M\#4$. Hence, since this is the nominal model, we conclude that a fault has occurred.

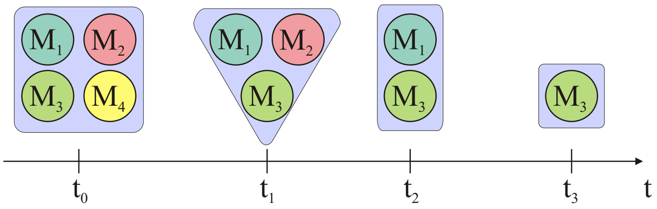


Figure 7.30: Example of the time-evolution of a set of models that are able to describe the input/output behavior of a given plant.

Moreover, consider that, at time t_2, model $M\#2$ is invalidated and that, finally, model $M\#1$ is invalidated at time t_3. Then, we conclude that the only model capable of explaining the input/output time-series generated by the plant is model $M\#3$. Thus, we have properly detected a fault at time t_1, and isolated this fault at time

t_4.

Architecture I

Consider, for instance, a loss-of-effectiveness type of fault in an actuator. As described in Section 7.6.2, this fault can be modeled by multiplying the actuator input by a constant $\lambda \in [0, 1]$. Therefore, consider an SVO, as described in Chapter 6, designed for a plant with this type of uncertainty. Then, such an SVO would validate observations from a model with any value of $\lambda \in [0, 1]$ and, in particular, for $\lambda = 1$, which corresponds to the nominal plant.

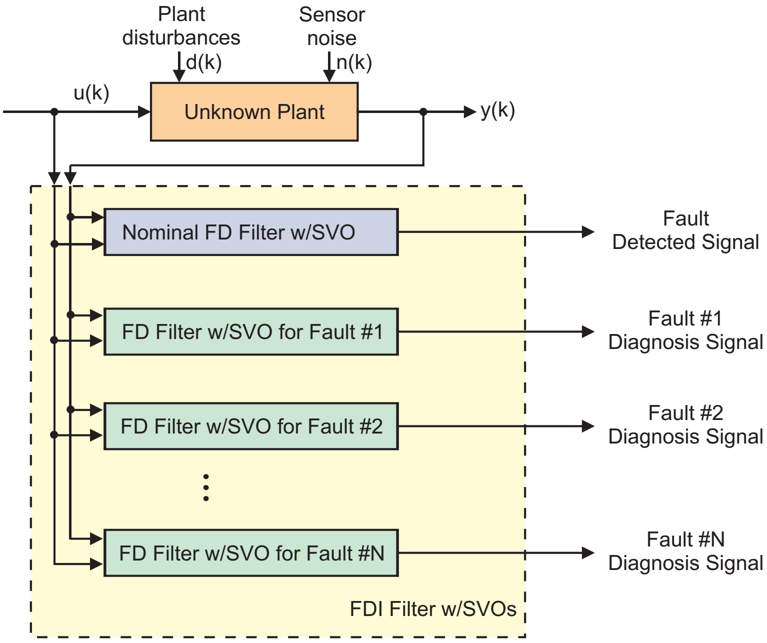


Figure 7.31: Fault Detection and Isolation (FDI) Architecture I for uncertain plants using set-valued observers (SVOs).

Indeed, the architecture described in the sequel, referred to as Architecture I and depicted in Fig. 7.31, assumes that the fault isolation filters provide valid set-valued estimates for the state of plant, not only for the faulty plant, with a specific fault, but also for the non-faulty plant. In addition, one SVO for the non-faulty (probably uncertain and time-varying) plant – referred to as *Nominal SVO* – can be used to detect the fault. As previously mentioned, the set-valued estimate for the state of the plant, obtained from this observer, is non-empty, if the plant does not present a faulty behavior. If the state estimate of the Nominal SVO is the empty set, a fault has occurred. A fault is completely isolated whenever a single fault isolation filter has a non-empty set-valued state estimation.

Remark 7.9: Despite the names of the filters, all the SVOs should be robust against model uncertainty. Thus, the *Nominal SVO* is in fact a robust SVO, as in Chapter 6, but designed for the dynamic model of the non-faulty plant. ◇

However, under certain circumstances, it may not be convenient to assume that the plants modeled by the fault isolation filters include the nominal model of the plant. As an example, suppose that we are interested in detecting faults

$$f_1, f_2, \cdots, f_N.$$

If we use Architecture I, one FD filter and N fault isolation filters are required. Each of these fault isolation filters computes two set-valued estimates for the state of the plant: one for the non-faulty model, $\hat{X}_{\text{nom}}$; and one for the faulty plant, with the corresponding failure, $\hat{X}_{\text{failure }\#i}$. Moreover, the SVOs require these set-valued estimates to be convex regions. Hence, the set-valued estimate of each fault isolation filter can be written as $\hat{X}_i = \text{co}\left\{\hat{X}_{\text{nom}}, \hat{X}_{\text{failure }\#i}\right\}$. This can add conservatism to the solution, and thus the isolation of the faults may become more difficult. A solution to overcome this problem is presented in the sequel.

Architecture II

The architecture described in what follows, referred to as Architecture II, requires two additional SVOs, besides the faults isolation SVOs, namely:

a) one SVO for the non-faulty (probably uncertain and time-varying) plant – referred to as *Nominal SVO*;

b) another SVO – referred to as *Global SVO* – providing set-valued estimates of the state, which are valid not only for the non-faulty plant, but also for the faulty plant, with any of the considered faults.

The *Nominal SVO* is used for fault detection only. If the state estimate of this SVO is the empty set, then a fault has occurred. Hence, the fault isolation SVOs are initialized with the state estimate of the *Global SVO*. A fault is completely isolated whenever a single fault isolation SVO has a non-empty set-valued state estimation.

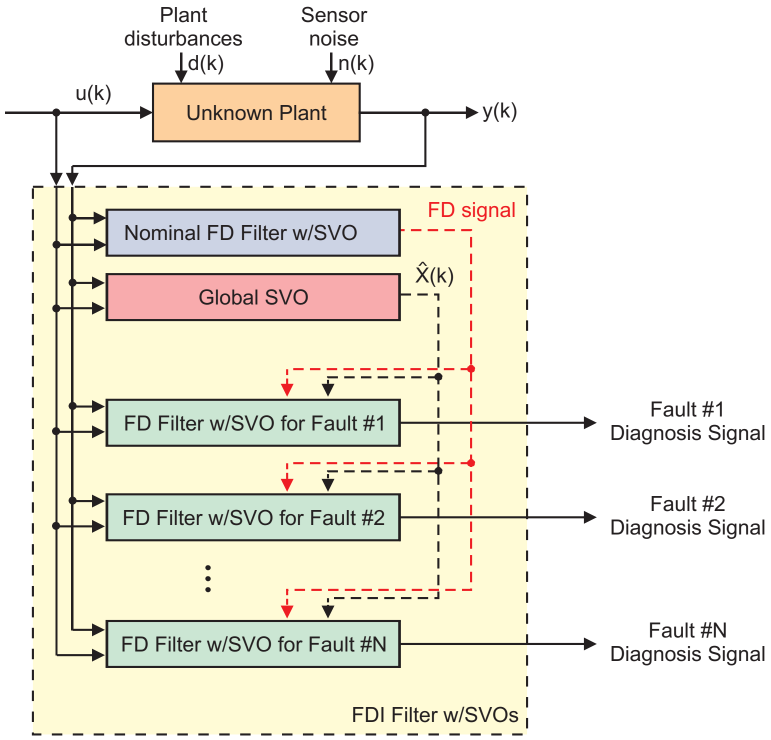

Plant disturbances
d(k)
Sensor noise
n(k)
u(k)
Unknown Plant
y(k)
Nominal FD Filter w/SVO
FD signal
Global SVO
X̂(k)
FD Filter w/SVO for Fault #1
Fault #1 Diagnosis Signal
FD Filter w/SVO for Fault #2
Fault #2 Diagnosis Signal
FD Filter w/SVO for Fault #N
Fault #N Diagnosis Signal
FDI Filter w/SVOs

Figure 7.32: Fault Detection and Isolation (FDI) Architecture II for uncertain plants using Set-Valued Observers (SVOs).

The proposed FDI architecture is depicted in Fig. 7.32. It should be stressed that the FD filters that are designed for specific faults, are only initialized with the set-valued state estimate of the Global SVO when they are signaled by the nominal FD filter that a fault has occurred.

Architecture II is a general approach which, however, may lead to some practical problems, since the uncertainty on the set-valued state estimate of the aforementioned *Global SVO* can be very *large*, which, in turn, can increase the time required to isolate a fault.

7.6.5 FD Example #1: Uncertain System

We are now going to analyze a series of experiments, designed to illustrate the wide applicability of the SVO-based FDI technique introduced in this section.

The first simulation at hand exploits the example described in [134, 169]. Let the linearized discrete-time model of a simplified longitudinal flight control system be described by

$$\left\{\begin{array}{rcl} x(k+1) &=& \left[A + A_\Delta\left(\delta_1, \delta_2, \delta_3, \delta_4, \delta_5, \delta_6\right)\right] x(k) + D_{p,1} w(k) \\ y(k) &=& Cx(k) + D_{p,2} w(k), \end{array}\right. \tag{7.17}$$

where the elements of the state vector

$$x(\cdot) = \begin{bmatrix} \eta_y(\cdot) & \omega_z(\cdot) & \delta_z(\cdot) \end{bmatrix}^{\mathrm{T}}$$

are the normal velocity, $\eta_y(\cdot)$, the pitch rate, $\omega_z(\cdot)$, and the pitch angle, $\delta_z(\cdot)$. The system matrices are

$$A = \begin{bmatrix} 0.8950 & -0.1083 & -0.3872 \\ 0.0015 & 0.8912 & -0.0672 \\ 0 & 0.7368 & 0 \end{bmatrix}, C = I_{3\times 3},$$

$$D_{p,1} = \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.01 \end{bmatrix}, D_{p,2} = 0.1 \times I_{3\times 3},$$

and the map $A_\Delta : \mathbb{R}^6 \to \mathbb{R}^{3\times 3}$ is given by

$$A_\Delta\left(\delta_1, \delta_2, \delta_3, \delta_4, \delta_5, \delta_6\right) = \sum_{i=1}^{6} A_i \delta_i,$$

where

$$A_1 = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, A_2 = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, A_3 = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$A_4 = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, A_5 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, A_6 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix},$$

with

- $|\delta_1| \leq 0.05818$;
- $|\delta_2| \leq 0.00704$;
- $|\delta_3| \leq 0.02517$;
- $|\delta_4| \leq 0.00010$;
- $|\delta_5| \leq 0.05793$
- and $|\delta_6| \leq 0.00440$.

Suppose that, at a given time k, the set-valued estimate of possible states of the plant is given by $X(k)$, where $X(k) := \text{Set}\,(P(k), p(k))$ is a convex polytope. For a given *valid* δ^*, and noticing that $C = I_{3\times 3}$, we can rewrite (7.17) as

a) $x(k+1) = [A + A_\Delta(\delta^*)]\, x(k) + D_{p,1}w(k) = [A + A_\Delta(\delta^*)]\,(y(k) - D_{p,2}w(k)) + D_{p_1}w(k)$;

b) $y(k) = x(k) + D_{p,2}w(k)$;

c) $x(k+1) = [A + A_\Delta(\delta^*)]\, x(k) + D_{p,1}w(k)$.

Therefore, we can write the following restrictions for $x(k+1)$:

a) $x(k+1) = [A + A_\Delta(\delta^*)]\,(y(k) - D_{p,2}w(k)) + D_{p,1}w(k)$;

b) $P(k)y(k) - P(k)D_{p,2}w(k) \leq p(k)$;

c) $P(k)\,[A + A_\Delta(\delta^*)]^{-1}\, x(k+1) - P(k)\,[A + A_\Delta(\delta^*)]^{-1}\, D_{p,1}w(k) \leq p(k)$,

which can be readily implemented as an SVO, since these restrictions define a convex polytope. Moreover, using the results in Chapter 6, we conclude that the convex hull of the set-valued estimates of the state, for the vertices of the hyper-rectangle of the eligible realizations of δ, contains all the values of $x(k+1)$ that are compatible with the observations.

We replicate the simulations in [134] in order to evaluate the behavior of the FD filter with SVOs. The disturbances are given by

$$w(k) = \begin{bmatrix} w_1(k) \\ w_2(k) \\ w_3(k) \end{bmatrix} = \begin{bmatrix} 0.7\sin(kT_s) \\ 0.5\sin(1.5kT_s) \\ 0.5\sin(0.75kT_s) \end{bmatrix}$$

where $T_s = 10$ ms is the sampling period. The bound considered by the SVOs for the disturbances amplitude is $|w(k)| \leq 0.951$. Finally, we assume that the initial state is inside a cube, centered at the origin, and all of whose sides are 20 units long.

The uncertain parameters in vector δ are assigned to their upper bounds. A sensor fault is then added to the system at time $t = 6$ s, by making the reading of the normal velocity, η_y, 0.6 times the actual system velocity. This fault is detected by the FD filter with SVOs in one measurement, as depicted in Fig. 7.33. Thus, at time $t = 6.01$ s, the FD filter with SVOs indicates a faulty behavior of the system.

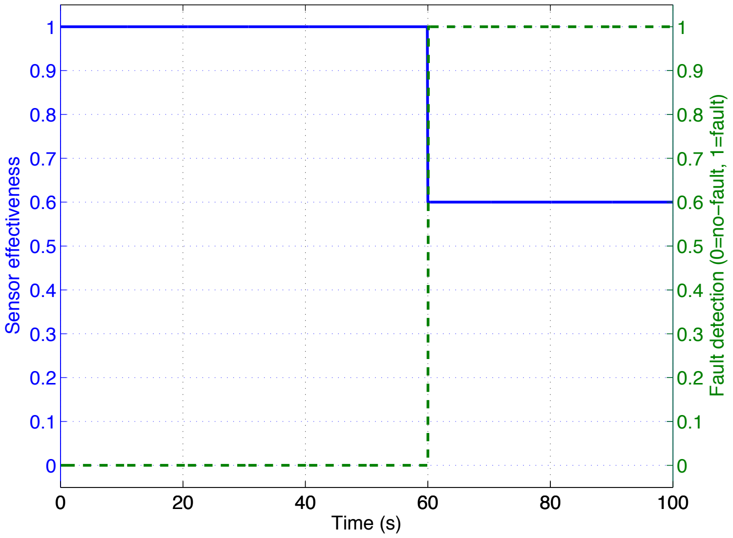


Figure 7.33: Fault detection of a plant with an uncertain dynamic model. The dashed (green) lines illustrate whether the set-valued estimate of the corresponding SVO is empty (1) or not (0).

7.6.6 FD Example #2: Linear Periodic System

As a second illustration of the applicability of the FD filter with SVOs, we use the example of a linear observable 3-periodic discrete-time system in [128], described by

$$\begin{cases} x(k+1) &= A(k)x(k) + B(k)u(k) + M(k)m(k) + L(k)d(k) \\ y(k) &= C(k)x(k), \end{cases} \tag{7.18}$$

where $A(k+3) = A(k)$, $B(k+3) = B(k)$, $C(k+3) = C(k)$, $M(k+3) = M(k)$ and $L(k+3) = L(k)$. The fault vector is represented by $m(k)$ and the disturbances are denoted by $d(k)$. Furthermore,

$$A(0) = \begin{bmatrix} 3 & 0 & -1 \\ -1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}, A(1) = \begin{bmatrix} 1 & 0 & 1 \\ 2 & -1 & 0 \\ 0 & 2 & 3 \end{bmatrix}, A(2) = \begin{bmatrix} 0 & 3 & 1 \\ 1 & 2 & 0 \\ -1 & 0 & 1 \end{bmatrix},$$

$$B(0) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, B(1) = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, B(2) = \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix},$$

$$C(0) = \begin{bmatrix} 2 & 0 & 3 \end{bmatrix}, C(1) = \begin{bmatrix} 0 & 1 & -1 \end{bmatrix}, C(2) = \begin{bmatrix} 1 & 0 & 2 \end{bmatrix},$$

$$M(0) = \begin{bmatrix} 1 \\ 3 \\ 0 \end{bmatrix}, M(1) = \begin{bmatrix} 0 \\ 2 \\ 6 \end{bmatrix}, M(2) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix},$$

$$L(0) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, L(1) = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, L(2) = \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix}.$$

The implementation of an SVO for this system is straightforward from Chapter 6. Notice that the SVO need not know the future system matrices at every sampling time, which allows accounting for time-varying systems in a very natural manner. Hence, a set-valued estimate for the state of a periodic discrete-time system can be generated and used for fault diagnosis.

In this simulation, we consider that the fault $m(k)$ is a step signal, with amplitude 0.3, starting at time $k = 40$. Similarly to what happened in the previous example, the fault is detected in the following sampling time, that is, at time $k = 41$, as illustrated in Fig. 7.34.

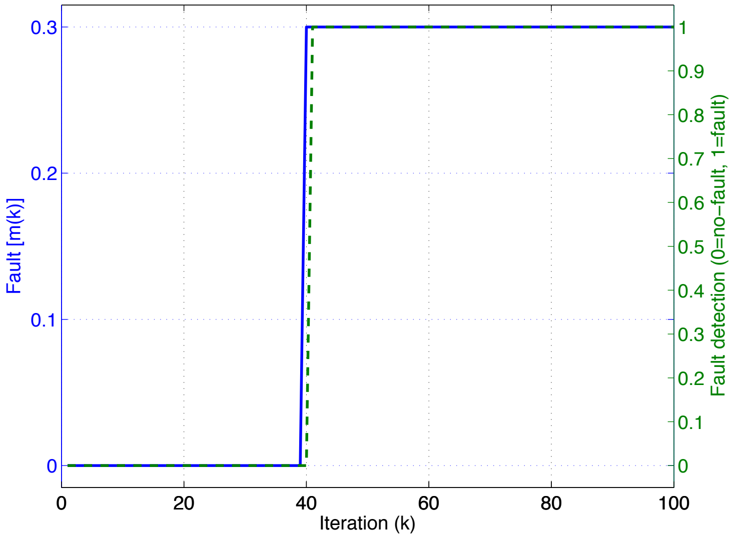


Figure 7.34: Fault detection of a linear periodic plant.

7.6.7 FD Example #3: LTV System

In the following simulation, we illustrate the use of SVOs for FD of a Linear Time-Varying (LTV) plant. We consider the example in [170], for comparison purposes. The LTV system analyzed in this subsection is described by

$$\begin{cases} \dot{x}(t) &= A(t)x(t) + B(t)u(t) + L_d(t)d(t) + B_f(t)f(t) \\ y(t) &= C(t)x(t) + D(t)u(t) + D_d(t)d(t) + D_f(t)f(t) \end{cases} \tag{7.19}$$

where $f(t)$ is the fault signal, and

$$A(t) = \begin{bmatrix} -0.1 & 1 - e^{-t/50} \\ 0 & -0.2 \end{bmatrix}, \qquad L_d(t) = \begin{bmatrix} 0.1 & 0 \\ 0.1 & 0 \end{bmatrix},$$

$$B(t) = \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix}, \qquad B_f(t) = \begin{bmatrix} 0 \\ 5 \end{bmatrix}, \qquad C(t) = \begin{bmatrix} 0.1 & 0.1 \end{bmatrix},$$

$$D(t) = \begin{bmatrix} 0 \end{bmatrix}, \qquad D_d(t) = \begin{bmatrix} 0.1 & 0.1 \end{bmatrix}, \qquad D_f(t) = \begin{bmatrix} 0 \end{bmatrix}.$$

We repeat the simulation in [170], where it is considered that

$$\begin{aligned} u(t) &= 0, \\ d(t) &= \begin{bmatrix} 0.2\sin(0.5t) \\ 0.2\cos(0.5t) \end{bmatrix}, \\ f(t) &= \begin{cases} 0, & t < 20 \text{ s}, \\ 0.1, & t \geq 20 \text{ s}. \end{cases} \end{aligned}$$

For the design of the FD-SVO, we assumed $|d| \leq 0.2$. The system (7.19) was discretized using a sampling period of $T_s = 100$ ms. The simulation results are depicted in Fig. 7.35. The fault starts at $t = 20$ s and is detected 1 s later. In [170], the residuals of the $\mathcal{H}_\infty$ FD filter can only identify a clear mismatch between the output and the plant model after nearly 5 s, which indicates a significant improvement in terms of response of the FD method. Moreover, even if we use conservative bounds for the plant disturbances, the results are not strongly deteriorated. As an example, if we assume that $|d| \leq 0.4$ in the design of the FD-SVO, then the time-interval required to detect the fault would be 1.6 s, and therefore the fault would still be diagnosed faster than with an $\mathcal{H}_\infty$ FD filter.

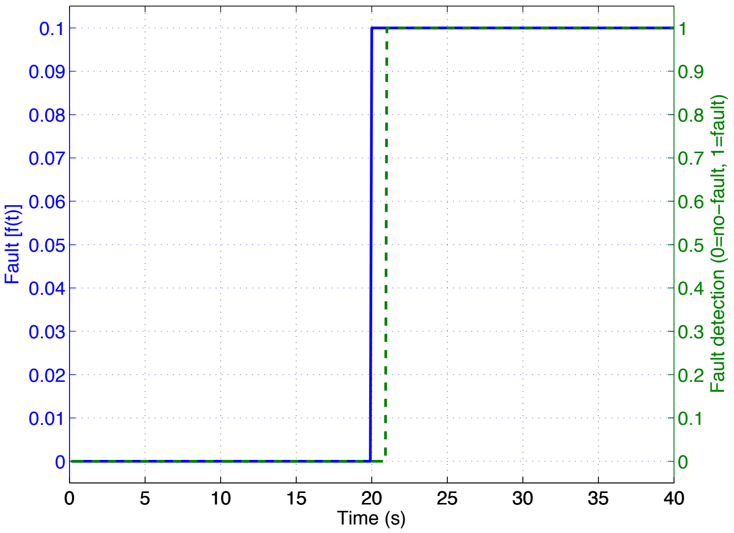


Figure 7.35: Fault detection of a Linear Time-Varying (LTV) plant.

7.6.8 FD Example #4: Nonlinear System

In this subsection, an example is provided showing how, in some special cases, it is possible to design SVOs for nonlinear models. Consider a plant described by the following dynamics:

$$\begin{cases} x(k+1) &= \begin{bmatrix} a(k) & 0.1 \\ 0 & 0.1 \end{bmatrix} + d(k), \\ y(k) &= x(k) + n(k), \end{cases} \tag{7.20}$$

where $|d| \leq 1$, $|n| \leq 1$, $a(k) := f(x(k)) := 0.1 \sin x_1(k)$, and $x_1(k)$ is the first component of vector $x(k)$. We further consider that $d(\cdot)$ and $n(\cdot)$ follow a uniform distribution. Define

$$\bar{a}(k) := \max_{x \in \hat{X}(k)} f(x)$$

and

$$\underline{a}(k) := \min_{x \in \hat{X}(k)} f(x).$$

Consider a system described by

$$\begin{cases} x(k+1) &= \begin{bmatrix} \theta(k) & 0.1 \\ 0 & 0.1 \end{bmatrix} + d(k), \\ y(k) &= x(k) + n(k), \end{cases} \tag{7.21}$$

where $|\Delta| \leq 1$ and $\theta(k) = \frac{\bar{a}(k)+\underline{a}(k)+\Delta(\bar{a}(k)-\underline{a}(k))}{2}$. Notice that

$$a(k) \in \left\{ a : a = \frac{\bar{a}(k) + \underline{a}(k) + \Delta\left(\bar{a}(k) - \underline{a}(k)\right)}{2}, |\Delta| \leq 1 \right\}.$$

Therefore, the solutions to (7.20) are contained in the set of solutions to (7.21). Notwithstanding the conservatism added due to this relaxation, Fig. 7.36 shows that, at least for the present example, the performance of the FD algorithm is not deteriorated. In this case, we assumed that, at $t = 10$ s, the effectiveness of the sensor that measures the first state of the plant is decreased by 80%. The sampling period is $T_s = 100$ ms.

The fault is diagnosed in nearly 200 ms. Hence, describing a nonlinear system by a linear uncertain time-varying dynamic model is a technique that, combined

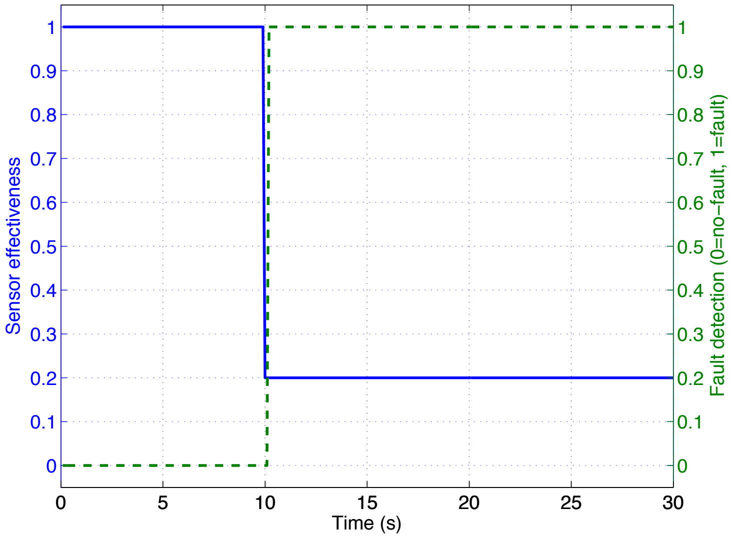


Figure 7.36: Fault detection of a nonlinear plant.

with SVOs, can be used for fault detection, despite the conservatism that may be added to the solution. Having conservative SVOs means that faults may be masked by model uncertainty and thus not properly detected. Nevertheless, we are still able to guarantee that no false alarms will occur.

7.6.9 FDI Example #1: Robotic Arm

We now recover the simulation example of Section 7.4, which was taken from [159], to show an illustration of the use of the SVOs not only for FD, but also for the isolation of the faults. In this example, a single-link robotic arm, with a revolute elastic joint, rotating in a vertical plane, is considered. The dynamics of the system are described by (7.4).

Similarly to [159], we assume that q_1, q_2 and $\dot{q}_2$ are available for measurement.

Thus, the model in (7.4) can be rewritten as

$$\left\{\begin{aligned} \begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\end{bmatrix} &= \begin{bmatrix} 0 & 1 & 0 & 0\\ \frac{-k}{J_\ell} & \frac{-F_\ell}{J_\ell} & \frac{k}{J_\ell} & 0\\ 0 & 0 & 0 & 1\\ \frac{k}{J_m} & 0 & \frac{-k}{J_m} & \frac{-F_m}{J_m}\end{bmatrix} \begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix} + \begin{bmatrix}0\\ \frac{-mg\ell}{J_\ell}\sin x_1\\ 0\\ \frac{u}{J_m}\end{bmatrix} + \begin{bmatrix}0\\ \eta\\ 0\\ 0\end{bmatrix}\\ y &= \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix} + \begin{bmatrix}0\\ d_2\\ d_3\end{bmatrix} + R\theta r(t-T)\end{aligned}\right. \tag{7.22}$$

where $x_1 = q_1$, $x_3 = q_2$, and where $\eta : \mathbb{R}^+ \to \mathbb{R}$ represents the model uncertainty in the derivative of q_1, with the constraint $|\eta| \leq \bar{\eta} = 0.05(mg\ell/J_\ell)\sin x_1$. In addition to the model uncertainty proposed in [159], we also consider that k is uncertain, with the constraint $k \in [1,\ 1.1]$ N/m. Moreover, (7.22) also comprises sensor faults in the term $R\theta r(t - T_o)$. The map $r : \mathbb{R} \to \mathbb{R}$ signals an abrupt fault occurring at time instant T_o, and is described by

$$r(t) = \begin{cases} 0, & t < T_o\\ 1, & t \geq T_o\end{cases}.$$

The parameter $\theta \in [0,\ 0.2]$ is the magnitude of the fault, and R is the corresponding direction. In particular, two sensor faults are considered:

a) Fault #1: sensor bias in the measurement of y_2, represented by $R = R_1 = \begin{bmatrix}0 & 1 & 0\end{bmatrix}^{\mathrm{T}}$,

b) Fault #2: sensor bias in the measurement of y_3, represented by $R = R_2 = \begin{bmatrix}0 & 0 & 1\end{bmatrix}^{\mathrm{T}}$.

As in [159], the unknown measurement noise is modeled by

$$d_2 = 0.01\sin(5t), \text{ and } d_3 = 0.01\sin(2t).$$

For the following simulations, we also considered that $u = 8\sin(t/3)$ and that the initial state of the plant is given by $x(0) = 0$. Moreover, we assumed that $\theta = 0.15$,

$T_o = 5$ s, and $k = k_{\min} = 1$ N/m. For the design of the SVOs, we only assumed that

$$k = k_{\min} + \delta_k,$$

for some $\delta_k \in \Delta_k = [0,\, 0.1]$ N/m.

The aforementioned model was discretized with a sampling period $T_s = 500$ ms and, using the FDI Architecture I, described in Section 7.6.4, *i.e.*, without the Global SVO, three SVOs were designed, namely, one for the non-faulty plant, one for fault #1, and another for fault #2.

Remark 7.10: The sensor faults considered (bias) can easily be accounted for during the design of the SVOs for the fault isolation filters in Fig. 7.31, by considering such faults as increased measurement noise. $\diamond$

The results for the detection and isolation of fault #1, *i.e.*, the bias in y_2, are depicted in Fig. 7.37. In this case, the fault is detected and isolated in one measurement, despite the uncertainties on the model of the plant.

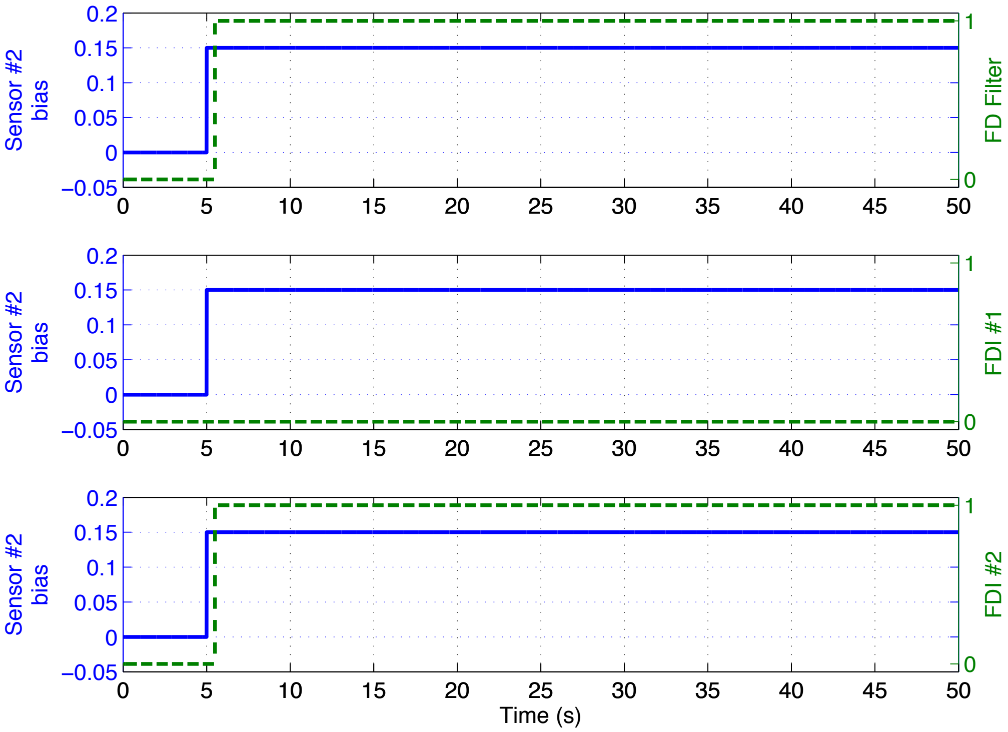


Figure 7.37: Fault #1 detection and isolation using SVOs. The bias in y_2 is detected and isolated in one iteration, *i.e.*, in 500 ms.

The detection of fault #2 requires also a single iteration of the SVOs. Nevertheless, 6 measurements are necessary for the isolation of the fault. Thus, during the first 3 s after the occurrence of the same fault, the input/output sequence of the plant is compatible with the model of the plant used by filter #2.

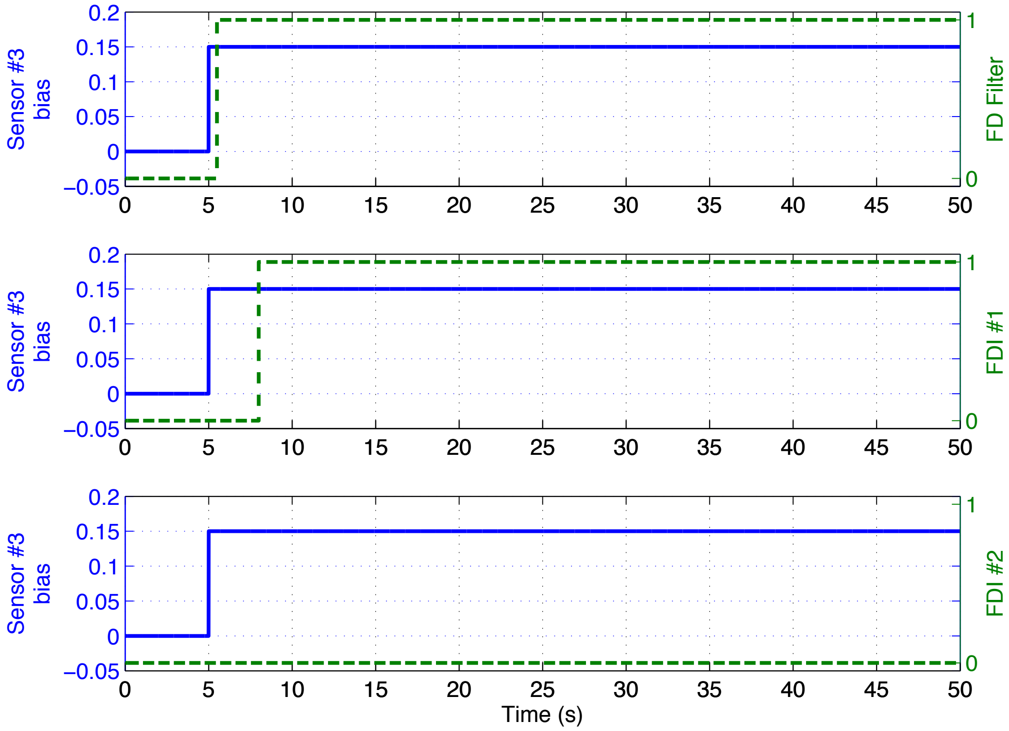


Figure 7.38: Fault #2 detection and isolation using SVOs. The bias in y_3 is detected in one measurement and isolated in six, *i.e.*, in 3 s.

It should be noticed that the uncertainty in k highly increases the difficulty on the FDI process. To see this, we used model (7.22) with an increasingly larger uncertainty in k, to assess the effects of this type of uncertainty on the time required to isolate the faults. The results obtained for 10 Monte-Carlo runs are depicted in Fig. 7.39. As previously, we considered that $k = k_{\min} = 1$ N/m, while for the design of the SVOs, we only assumed that $k = k_{\min} + \delta_k$, for some $\delta_k \in \Delta_k$. As expected, the number of iterations required to isolate the faults increases with the uncertainty in k.

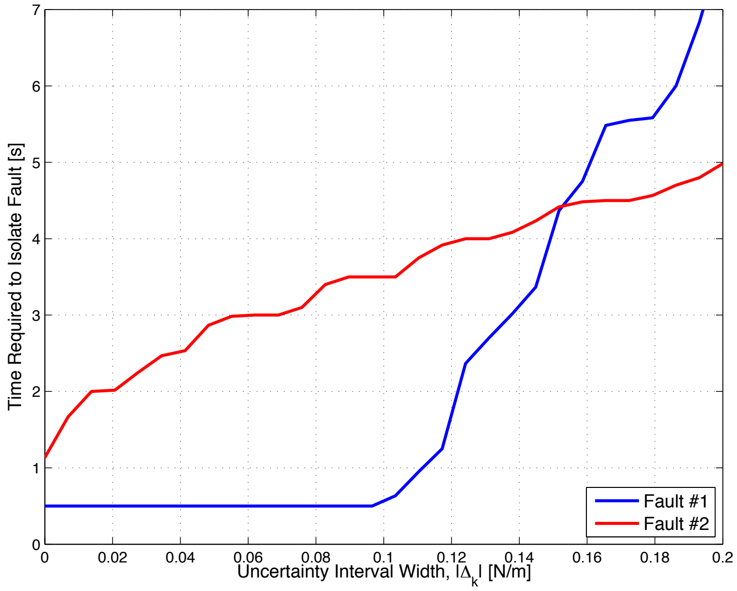


Figure 7.39: Time required to isolate faults #1 and #2 versus the uncertainty in k.

7.6.10 FDI Example #2: Aircraft Longitudinal Dynamics

In this subsection, we study the performance of the SVO-based FDI strategy developed in this chapter, when applied to an aircraft LPV longitudinal model.

The dynamics of an aircraft are highly nonlinear, and depend on several (time-varying) parameters, such as the dynamic pressure and the aerodynamic coefficients. However, in constant altitude steady-state flight, these dynamics are well-described by LPV models, which depend upon the airspeed. In particular, consider the aircraft LPV longitudinal model presented in [171]. This model can be described by the following linearized equations:

$$\begin{aligned}
\tfrac{du}{dt} - X_u u + g\cos\Theta_o\theta &= 0, \\
-Z_u u + V\tfrac{d\alpha}{dt} - Z_\alpha\alpha + (V+Z_q)q + g\sin\Theta_o\theta &= Z_{\delta_e}\delta_e, \\
-M_u u - M_{\dot\alpha}\tfrac{d\alpha}{dt} - M_\alpha\alpha + \tfrac{dq}{dt} - M_q q &= M_{\delta_e}\delta_e, \\
\tfrac{d\theta}{dt} &= q.
\end{aligned} \tag{7.23}$$

The longitudinal states are the forward airspeed, u (which should not be confused with the control input in (7.14)), the pitch angle, θ, the angle-of-attach, α, and the pitch rate, q. The parameters of the model are the stability and control derivatives (described in the sequel), the magnitude of the gravity vector, g, the pitch trimming angle, Θ_o, and the airspeed, V, *i.e.*, the magnitude of the velocity of the aircraft relative to the fluid. Moreover, δ_e is the deviation of the elevator angle.

As explained in detail in [171], by defining $x(t)$ and $y(t)$ as

$$x(t) := \begin{bmatrix} u(t) \\ \theta(t) \\ \alpha(t) \\ q(t) \end{bmatrix}, \quad y(t) := \begin{bmatrix} u(t) \\ \theta(t) \\ \alpha(t) \end{bmatrix},$$

the dynamics in (7.23) can be rewritten as the following continuous-time LPV model:

$$\begin{cases} \frac{d}{dt}x(t) &= A(V,\xi(V))x(t) + B(V,\xi(V))\delta_e(t), \\ y(t) &= Cx(t), \end{cases} \tag{7.24}$$

where

$$\begin{array}{lll} A(V,\xi(V)) = [a_{ij}], & & \\ a_{11} = X_u, & a_{12} = -g\cos\Theta_o, & a_{13} = X_\alpha, \\ a_{14} = 0, & a_{21} = 0, & a_{22} = 0, \\ a_{23} = 0, & a_{24} = 1, & a_{31} = \frac{Z_u}{V}, \\ a_{32} = -\frac{g\sin\Theta_o}{V}, & a_{33} = \frac{Z_\alpha}{V}, & a_{34} = 1 + \frac{Z_q}{V}, \\ a_{41} = M_u + M_{\dot\alpha}\frac{Z_u}{V}, & a_{42} = -M_{\dot\alpha}\frac{g\sin\Theta_o}{V}, & \\ a_{43} = M_\alpha + M_{\dot\alpha}\frac{Z_\alpha}{V}, & a_{44} = M_q + M_{\dot\alpha}\left(1 + \frac{Z_q}{V}\right), & \end{array}$$

$$B(V,\xi(V)) = \begin{bmatrix} 0 \\ 0 \\ \frac{Z_{\delta_e}}{V} \\ M_{\delta_e} + M_{\dot\alpha}\frac{Z_{\delta_e}}{V} \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

The stability and control derivatives (SCDs) are concatenated in vector $\xi(V)$, *i.e.*,

$$\xi(V) = [X_u,\ X_\alpha,\ Z_u,\ Z_\alpha,\ Z_q,\ M_u,\ M_\alpha,\ M_{\dot\alpha},\ M_q,\ Z_{\delta_e},\ M_{\delta_e}].$$

For a nominal airspeed of $V_o = 150$ m/s, the numeric values of the SCDs are summarized in Table 7.7.

Table 7.7: Stability and control derivatives for $V_o = 150$ m/s.

SCD	Value	SCD	Value
X_u	-0.0298	X_α	12.609
Z_u	-0.3065	Z_α	-161.54
Z_q	-1.5464	M_u	0.0013
M_α	-7.9295	M_q	-1.8485
$M_{\dot{\alpha}}$	0.1167	Z_{δ_e}	-11.374
M_{δ_e}	-5.9544		

The LPV model in (7.24) is going to be used in simulation as a realization of the dynamics of the aircraft, and to design the SVOs for the FDI method described in this chapter.

Simulations

We now use the aforementioned aircraft model for the longitudinal axis, discretized with a sampling period $T_s = 200$ ms.

Thereafter, a fault detection filter as in Section 7.6.3 was synthesized in order to diagnose faults in the longitudinal dynamics of the aircraft. For the simulations, both the exogenous disturbances, $d(k)$, and the sensors noise, $n(k)$, were generated using a uniform distribution. Moreover, four fault isolation filters were also designed using the approach in Section 7.6.4, in order to isolate the following failures in the aircraft:

a) FDI #1: loss-of-effectiveness (LOE) in the forward velocity, u, sensor;

b) FDI #2: LOE in the pitch angle, θ, sensor;

c) FDI #3: LOE in the angle-of-attach, α, sensor;

d) FDI #4: LOE in the elevator, δ_e.

The FDI architecture used is depicted in Fig. 7.40. In this case, the FDI filters were designed so that loss-of-effectiveness type of faults can be diagnosed. Thus, as explained in detail in Section 7.6.4, the FDI filters need not be reset whenever a fault is detected by the nominal filter.

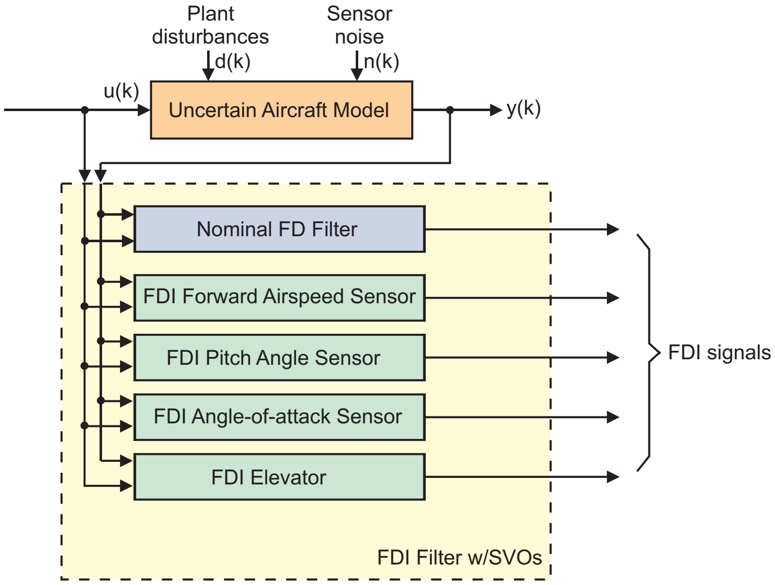


Figure 7.40: FDI architecture using SVOs for the aircraft longitudinal model.

An LQG controller was designed for the aircraft linearized model, around the nominal airspeed of $V_o = 150$ m/s, in order to be able to generate faults in the actuator.

Two different scenarios are going to be analyzed in the sequel. The first one consists in generating abrupt (or hard) faults in the sensors/actuator. We start by considering a hard fault in the elevator. In particular, for $t \geq 20$ s, the elevator becomes stuck at zero, *i.e.*, $\delta_e(t) = 0$ for $t \geq 20$ s. For this configuration, the average results for 5 Monte-Carlo runs are depicted in Fig. 7.41. The faults were detected in less than 3 measurements, *i.e.*, in less than 600 ms, and isolated in less than 1 s.

A similar trial was tested for a hard failure in the forward speed sensor of the aircraft. Suppose that, for $t \geq 20$ s, the effectiveness of the forward speed sensor is decreased by 40%, *i.e.*, the reading acquired from the sensor corresponds to 60% of the *true* forward speed of the aircraft. For this case, the results obtained by averaging 5 Monte-Carlos runs are illustrated in Fig. 7.42. The faults were detected

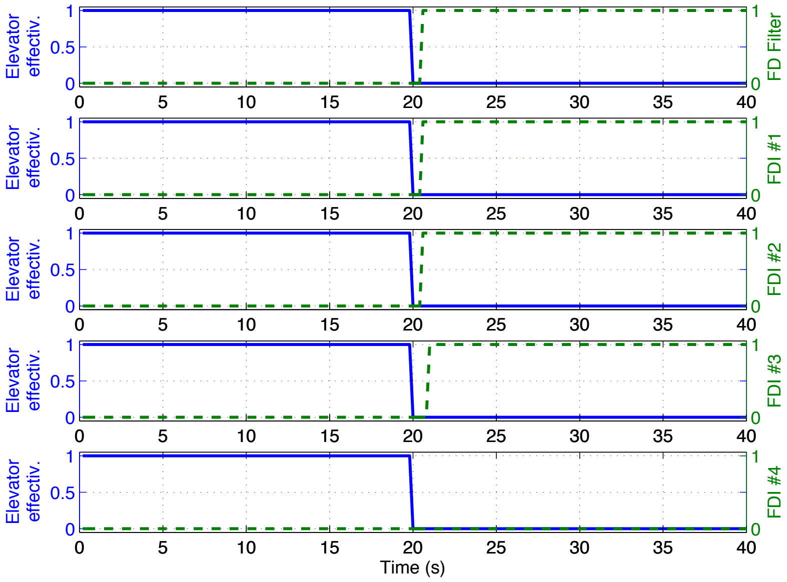


Figure 7.41: Hard fault in the elevator of the aircraft. The results shown were obtained by averaging 5 Monte-Carlo runs. After nearly 1 s, the only FDI filter that is able to explain the observations is FDI #4. Therefore, the fault in the elevator is isolated in nearly 1 s.

and isolated in less than 2 measurements, which is equivalent to 400 ms. In fact, in most cases, only one measurement was required to isolate this fault.

Remark 7.11: In order to give some insight regarding why the failures in the forward speed sensor are, in these simulations, more quickly detected and isolated than the faults in the elevator, we stress that the changes in the forward speed affect not only the state trajectory $u(\cdot)$, but also the dynamics of the model. To see this, notice that $A(\cdot,\cdot)$ and $B(\cdot,\cdot)$ in (7.24) depend upon the airspeed, $V(\cdot)$, which is obviously related to $u(\cdot)$. $\diamond$

It should be noticed, however, that hard faults are, in general, "easier" for detection. Indeed, for the second scenario, we consider smooth (or soft) faults in the sensors/actuator, thus representing more realistic failures.

Suppose that the effectiveness of the elevator suffers the variation depicted in Fig. 7.43, *i.e.*, the effectiveness of the actuator decreases linearly during 2 s. In this case, the FD filter with SVOs takes nearly 600 ms to detect the fault, as shown in Fig. 7.43. Moreover, the fault is isolated in 2 s. In comparison with the previous

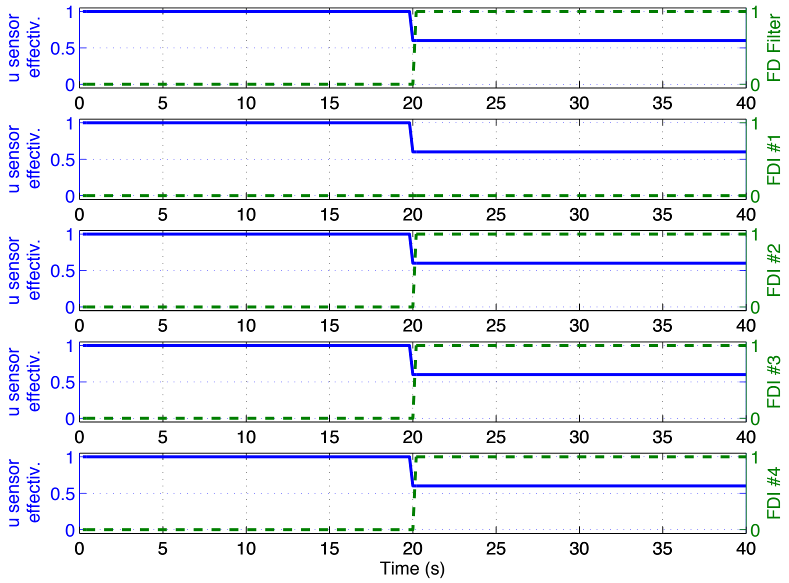


Figure 7.42: Hard fault in the forward speed sensor. The fault in the forward speed sensor is isolated in nearly 400 ms.

scenario (hard failures), the FDI system requires more time to detect and isolate the faults, as expected.

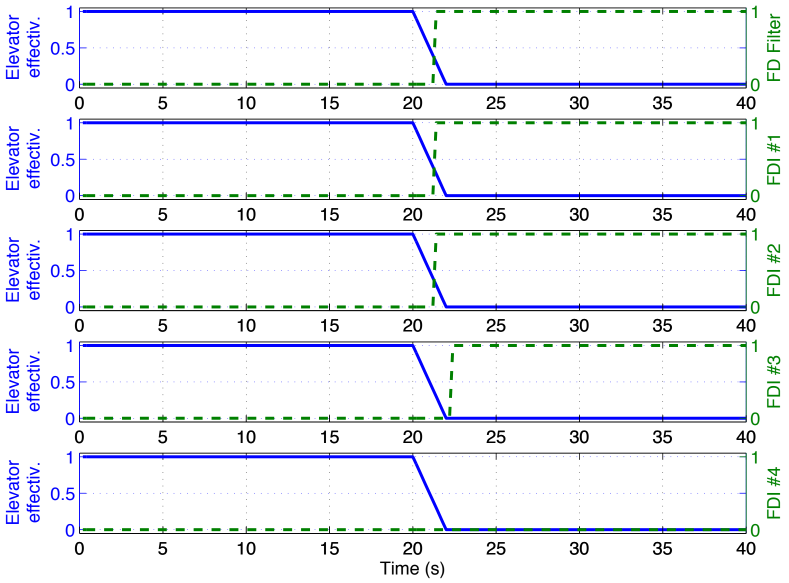


Figure 7.43: Soft fault in the elevator of the aircraft. The fault in the elevator is isolated in nearly 2 s.

The results for a soft fault in the forward speed sensor are depicted in Fig. 7.44.

In this situation, the results are not significantly affected by the smoothness of the fault. Indeed, only 2 measurements are required to isolate the fault. The same reasoning as in Remark 7.11 applies to this case.

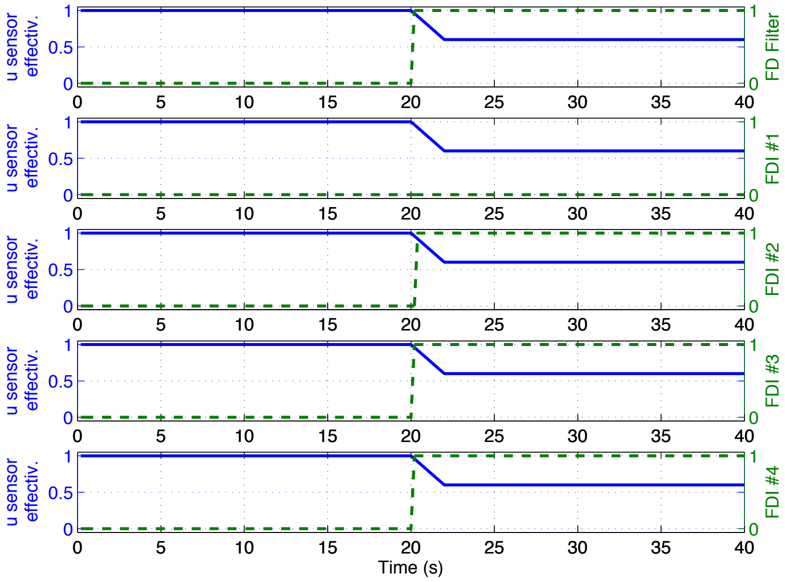


Figure 7.44: Soft fault in the forward speed sensor. The fault in the forward speed sensor is isolated in nearly 400 ms.

7.7 Conclusions

This chapter addressed the problem of *model falsification* using Set-Valued Observers (SVOs), for uncertain Linear Parameter-Varying (LPV) systems. The main goal of this chapter was to develop a series of architectures and algorithms that, under certain conditions, can provide guarantees in terms of model selection. In particular, the model falsification techniques developed in this chapter take advantage of the *absolute input distinguishability* concept, introduced in Chapter 5, to ensure that a single model is selected after a given (finite) number of measurements. Moreover, the results in Chapter 6 are used to guarantee that a *plausible* model of a plant is never invalidated. Unlike some of the solutions available in the literature, the methods suggested to implement this model falsification technique are recursive.

One of the possible applications of the model falsification approach developed in this chapter is related to Multiple-Model Adaptive Control (MMAC). Thus, this chapter also introduced the MMAC/SVO scheme, which yields stability and performance guarantees under several uncertain and time-varying scenarios.

As a second example of the applicability of this model falsification method, this chapter described the use of this technique in Fault Detection and Isolation (FDI). In this case, we are able to guarantee that, under certain circumstances, a fault is always detected and isolated in a finite number of measurements, and that false alarms are avoided. Moreover, all the properties of the SVOs of Chapter 6 are inherited, which means that uncertain LPV models can be handled.

Although several numerical simulations, illustrating the applicability of the aforementioned model falsification technique, were presented in this chapter, the MMAC/SVO scheme was not analyzed in detail. Therefore, the following chapter is devoted to the (thorough) comparison between the MMAC/SVO and RMMAC/BMI schemes, when applied to a double Mass-Spring-Dashpot (MSD) system.

Chapter 8

Comparison of MMAC/SVO vs RMMAC/BMI

8.1 Introduction

In the previous chapter, a novel adaptive control architecture, referred to as Multiple-Model Adaptive Control using Set-Valued Observers (MMAC/SVO), was introduced. Although some simulation results were presented in order to illustrate the behavior of the proposed adaptive control method, it was not clear how the MMAC/SVO compares with other adaptive control schemes. Therefore, this chapter is devoted to an insightful comparison between the MMAC/SVO and another well-known adaptive control architecture, referred to as Robust Multiple-Model Adaptive Control (RMMAC).

As previously mentioned in this thesis – see Chapter 1 –, the RMMAC is a multiple-model approach that computes and uses the posterior probabilities of the uncertain parameters of the process model being in a specific region to switch or blend the outputs of a set of controllers, each of which designed for a given uncertainty region. The estimation part is done by a bank of Kalman filters (KFs), while for the control part a set of mixed-μ controllers is used – see Fig. 8.1, where the Local Non-Adaptive Robust Controllers (LNARCs) are the mixed-μ controllers, in the original RMMAC architecture. For further details, the interested reader is

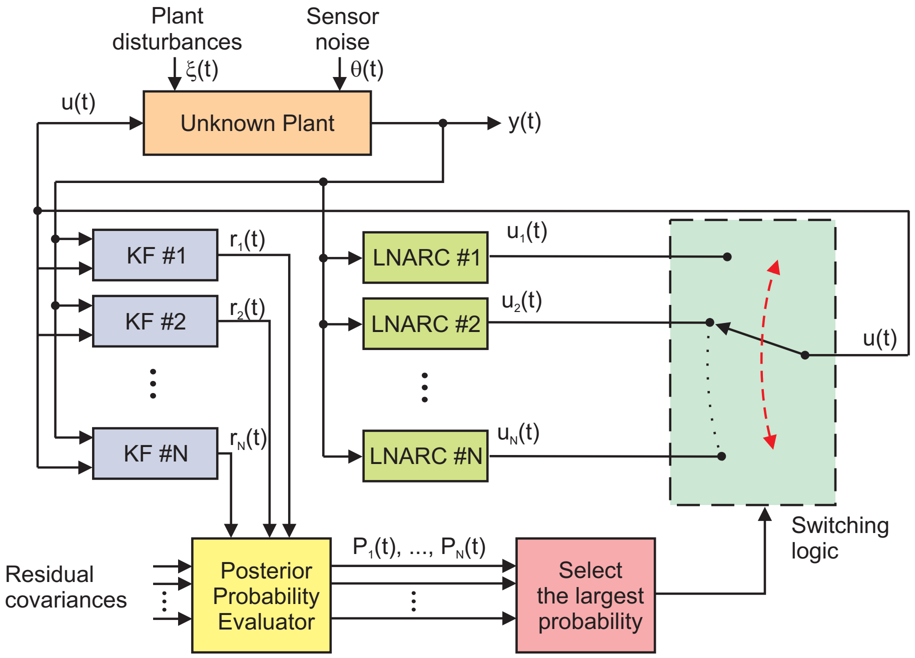


Figure 8.1: RMMAC/BMI architecture.

referred to [57–59].

As shown in Chapter 3, the mixed-μ controllers of the RMMAC can be substituted by LPV/BMI controllers – see Chapter 2 – in order to account for time-variations of the uncertain parameters of the dynamics of the plant. The design specifications of the LPV/BMI controllers are similar to those of the mixed-μ controllers, but assuming nonzero bounds on the rate of variation of the uncertain parameters of the plant. Since the design assumptions are very similar in both cases, except for the time-variation of the parameter, little effort is needed in order to "upgrade" the standard RMMAC method to this novel architecture. In reference to Fig. 8.1, the LNARCs are now LPV/BMI controllers.

8.1.1 Main Contributions and Organization

The key contributions of this chapter are as follows:

a) An insightful practical comparison between the RMMAC/BMI and the MMAC/SVO architectures;

b) The application of the methodology presented in the previous chapter to integrate the MMAC/SVO with another heuristic, using RMS considerations regarding the closed-loop system.

The remainder of this chapter is organized as follows: in Section 8.2, the double Mass-Spring-Dashpot (MSD) plant is revisited; Section 8.3 presents several Monte-Carlo simulations, illustrating the advantages and shortcomings of the RMMAC/BMI architecture, while a method to integrate the MMAC/SVO with closed-loop RMS considerations is used; finally, in Section 8.4, the results obtained are discussed.

8.2 Mass-Spring-Dashpot Plant

In Section 3.2, a time-varying double Mass-Spring-Dashpot (MSD) plant was used in order to evaluate the performance of the RMMAC, with mixed-μ and LPV/BMI controllers. In the remainder of this chapter, an MSD plant is also used, but considering no input time-delay, as depicted in Fig. 8.2.

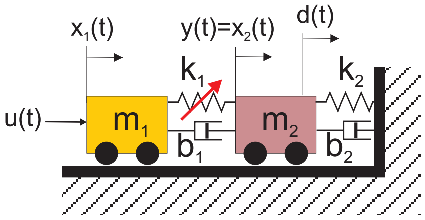


Figure 8.2: Mass-spring-dashpot plant.

In this case, the uncertain parameter is the spring stiffness, $k_1(\cdot)$, which is assumed to be time-varying, but with known bounds on the rate of time-variation. Moreover, it is assumed that

$$\forall_{t\geq 0}, k_1(t) \in K := [0.25 \quad 1.75] \text{ N/m}.$$

The state-space description of the dynamics of the MSD system, excluding the dynamics of the disturbances, is given by

$$\begin{cases} \dot{x}(t) &= A(k_1)x(t) + Bu(t) + Ld(t), \\ y(t) &= Cx(t) + \theta(t), \end{cases}$$

where

$$x^{\mathrm{T}}(t) = \begin{bmatrix} x_1(t) & x_2(t) & x_3(t) & x_4(t) \end{bmatrix}$$

is the state of the plant, and

$$A(k_1) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{k_1}{m_1} & \frac{k_1}{m_1} & -\frac{b_1}{m_1} & \frac{b_1}{m_1} \\ \frac{k_1}{m_2} & -\frac{k_1+k_2}{m_2} & \frac{b_1}{m_2} & -\frac{b_1+b_2}{m_2} \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{m_1} \\ 0 \end{bmatrix}, \quad L = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix},$$

$$C = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}.$$

where

$$m_1 = m_2 = 1 \text{ kg}, \quad k_2 = 0.15 \text{ N/m}, \quad b_1 = b_2 = 0.1 \text{ N/(m/s)}.$$

The disturbance force, $d(\cdot)$, shown in Fig. 8.2 is generated by driving a low-pass filter, with transfer function $W_d(s)$, with continuous-time bounded Gaussian noise $\xi(\cdot)$, with zero mean, bound of $\Gamma\bar{d}$ and intensity of $\Gamma\Xi$, according to

$$d(s) = \operatorname{sat}\left(\frac{\alpha}{s+\alpha}\xi(s), \Gamma\bar{d}\right) = \operatorname{sat}\left(W_d(s)\xi(s), \Gamma\bar{d}\right),$$

where $\Gamma = 1$, $\Xi = 1$, and $\bar{d} = 3$, for the nominal case, and where

$$\operatorname{sat}(x, y) = \begin{cases} x, & \text{if} \quad -y \leq x \leq y, \\ y, & \text{if} \quad x > y, \\ -y, & \text{if} \quad x < -y, \end{cases}$$

for $y \geq 0$. We consider that the sensor noise, $\theta(\cdot)$, is also obtained from a Gaussian distribution, with zero mean and intensity 10^{-6}, saturated by $\bar{n} = 0.003$. The reader is referred to [34, 57–59] for further details on the dynamics of the MSD system.

Following the RMMAC/BMI synthesis methodology and using the same design choices as the ones described in [58, 59], we obtain $N = 4$ LNARCs – which are mixed-μ controllers in the original RMMAC design – in order to achieve at least 70% of the performance we would have obtained, had we known the value of the uncertain parameter, k_1. The 4 regions of uncertainty are summarized in Table 8.1. Then, as explained in [49], the mixed-μ controllers are replaced by BMI/LPV controllers

Table 8.1: Regions of uncertainty for the spring stiffness, k_1.

Region Number	Spring Stiffness Uncertainty [N/m]
#1	[1.02, 1.75]
#2	[0.64, 1.02]
#3	[0.40, 0.64]
#4	[0.25, 0.40]

with similar specifications, but assuming nonzero bounds on the rate of variation of the parameter, k_1, which results in the RMMAC/BMI adaptive control scheme. In this design, we assume a bound of 0.001 (N/m)/s for the slope (time-variation) of the parameter k_1.

One SVO for each region of uncertainty was designed, using a sampling time of $T_s = 100$ ms, and considering a bound on the disturbances of

$$\forall_k,\ |d(k)| \leq d^1_{\max} := 3 \text{ N}.$$

Moreover, a Global SVO was synthesized for the whole region of uncertainty, *i.e.*, for $k_1 = [0.25, 1.75]$ N/m, and assuming no constraints for the time-rate of variation of the uncertain parameter, k_1. Moreover, for the Global SVO, it was considered that the disturbances were bounded by

$$\forall_k,\ |d(k)| \leq \bar{d}_{\text{SVO}} := 30 \text{ N}.$$

Two algorithms for the selection of the controllers are going to be used. In the first case, we consider the simplest MMAC/SVO architecture and algorithm, where the selection of the controllers is performed in a sequential manner, *i.e.*, we start by using LNARC #1 and switch to LNARC $\#i + 1$ (or to model $\#i + j$, for $j > 1$ and as small as possible, if model $\#i + 1$ was previously falsified) whenever model $\#i$ is invalidated. If all the SVOs are invalidated, then we switch to LNARC #1 and reset the set-valued estimates of the SVOs with that of the Global SVO, as illustrated in Fig. 8.3.

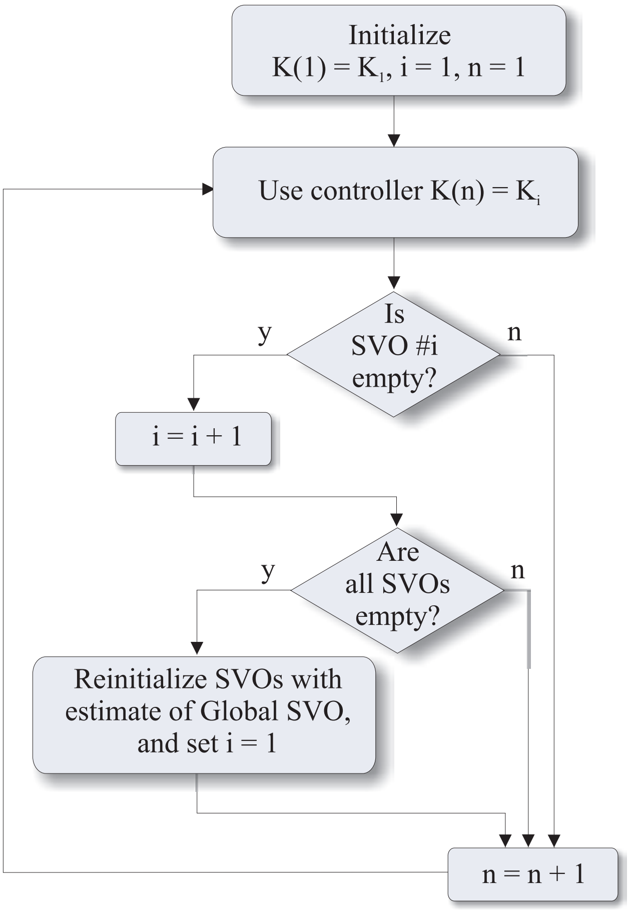


Figure 8.3: Algorithm for the sequential selection of controllers, in the MMAC/SVO architecture for time-varying systems.

The second algorithm that will be used in simulation was described in the previous chapter, and is depicted in Fig. 7.15.

8.3 Simulations

A set of simulations will be presented in what follows, in order to compare the behavior of the MMAC/SVO with that of the RMMAC/BMI, when applied to the MSD plant previously described. Although several Monte-Carlo simulations have been performed, only a representative set will be considered here for analysis.

8.3.1 $k_1 = 0.25$ N - Forced Instability

Consider that $\Gamma = 1$ and let $k_1 = 0.25$ N/m, which means that the dynamics of the plant can be described by model #4. We now analyze the mismatch-stability properties of the MMAC/SVO, when compared to those of the RMMAC/BMI.

This can be achieved by forcing the closed-loop to become unstable, during a small amount of time. In the present example, we force the decision subsystems of both the MMAC/SVO and the RMMAC/BMI to select the LPV/BMI controller synthesized for the region of uncertainty #1, during 100 ms. Notice that, as stressed in [34] for the case where μ-compensators are used, the controller for the region of uncertainty #1 is a destabilizing controller for a model in region #4.

Therefore, consider that, for $t \in [100, 100.1]$ s, the decision subsystems of both architectures select controller #4. The results obtained for a typical Monte-Carlo simulation are depicted in Fig. 8.4.

In this case, the MMAC/SVO performs significantly *worse* than the RMMAC/BMI, due to the large transients observed before the former adaptive control laws selects the appropriate controller. A physical interpretation of this fact is described in Remark 8.2.

The MMAC/SVO is eventually able to switch to controller #4, nearly 14 secs after the forced model-mismatch instability, as depicted in Fig. 8.5. However, before forcing the instability of the system, the MMAC/SVO was not able to falsify region

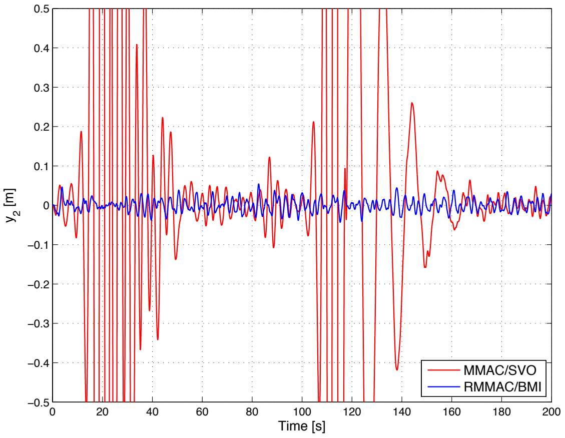


Figure 8.4: Output of the MSD plant, for forced instability and $k_1 = 0.25$ N/m.

of uncertainty #3.

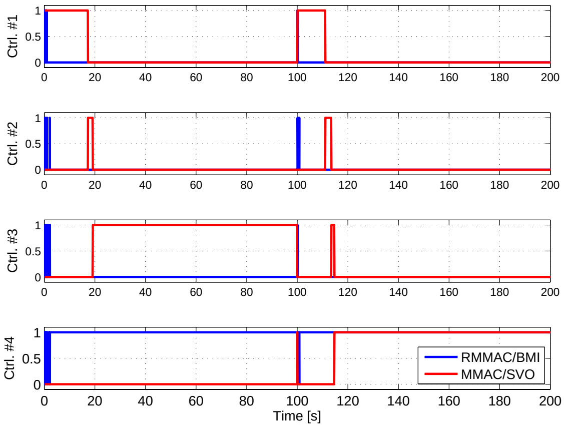


Figure 8.5: Controller selection, for forced instability and $k_1 = 0.25$ N/m.

This example highlights some of the shortcomings of the MMAC/SVO, which are going to be addressed in the sequel.

8.3.2 $k_1 = 1.75$ N - Forced Instability

As happened with the previous case, let $\Gamma = 1$, but assume that $k_1 = 1.75$ N/m, which means that the dynamics of the plant can be described by model #1. Analogously to the previous example, we force the decision subsystems of both the MMAC/SVO and the RMMAC/BMI to select the LPV/BMI controller synthesized for the region of uncertainty #4, during 100 ms. We stress that the controller for the region of uncertainty #4 is a destabilizing controller for a model in region #1.

Hence, consider that, for $t \in [100, 100.1]$ s, the decision subsystems of both architectures select controller #4. The results obtained for a typical Monte-Carlo run are depicted in Fig. 8.6.

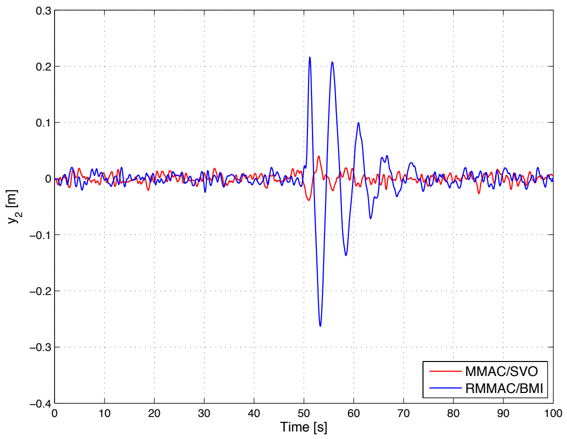


Figure 8.6: Output of the MSD plant, for forced instability and $k_1 = 1.75$ N/m.

The performance of the MMAC/SVO, for this case, is considerably better than that of the RMMAC/BMI. As illustrated in Fig. 8.7, the MMAC/SVO algorithm only requires a single measurement to correctly invalidate model #4, and hence switch back to controller #1.

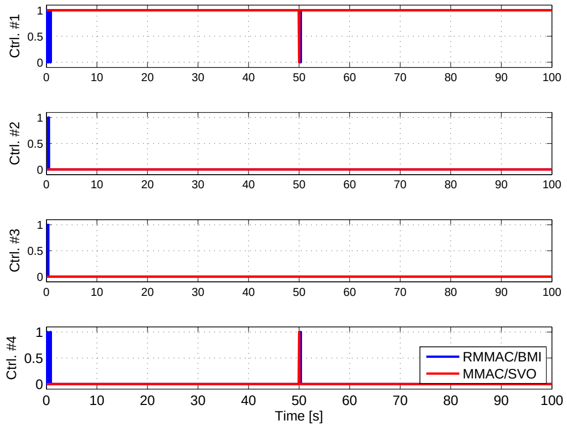


Figure 8.7: Controller selection, for forced instability and $k_1 = 1.75$ N/m.

Remark 8.1: In the MMAC/SVO design, as soon as model #4 is invalidated, controller #1 is selected again, as described in Fig. 8.3. To see this, note that, in order to force the selection of model #4, we have to impose the falsification of models #1, #2 and #3. Hence, if model #4 is also falsified, we reinitialize the SVOs with the set-valued state estimate of the Global SVO, and connect controller #1 to the loop immediately. $\diamond$

Remark 8.2: The large transient observed with the MMAC/SVO architecture, when $k_1 = 0.25$ N/m and the model-mismatch instability is forced, can intuitively be explained by the following physical considerations. We start by recalling that we are only measuring the position of mass m_2. If k_1 is small, in particular, $k_1 = 0.25$ N/m, then the system is considerably slower than that for $k_1 \in [1.02, 1.75]$ N/m. Therefore, for small k_1, the energy transferred by the control input, $u(\cdot)$, to mass m_1, will take longer to impact on mass m_2. Since LPV/BMI controller #4 (designed for $k_1 \in [1.02, 1.75]$ N/m) assumes a fast response of the system, it will try to quickly compensate for the delay observed in the position of mass m_2, as the controller will "blame" the disturbances for this unexpected (slow) behavior of the plant. This over-compensation will naturally lead to large amplitude signals in a small amount of time. $\diamond$

8.3.3 MMAC/SVO-RMS: $k_1 = 0.25$ N/m - Forced Instability

The deterioration in terms of performance of the MMAC/SVO, when compared to the RMMAC/BMI, was largely due to the transients observed whenever a new controller had to be selected. Indeed, since the MMAC/SVO can be seen as a *worst case* approach, a controller is not invalidated unless the input/output sequences cannot be explained by the dynamics of the closed-loop system. Hence, an enhancement of the MMAC/SVO architecture, referred to as MMAC/SVO-RMS, was proposed in the previous chapter, that takes into account the RMS of the output error.

For the case of the MSD plant, depicted in Fig. 8.2, the upper bounds on the RMS of the closed-loop, in steady state, are summarized in Fig. 8.8.

We start by repeating the first simulation, for which we considered $k_1 = 0.25$ N/m, with forced model-mismatch instability at time $t = 100$ secs. The output of the MSD plant, interconnected with what we call the MMAC/SVO-RMS adaptive

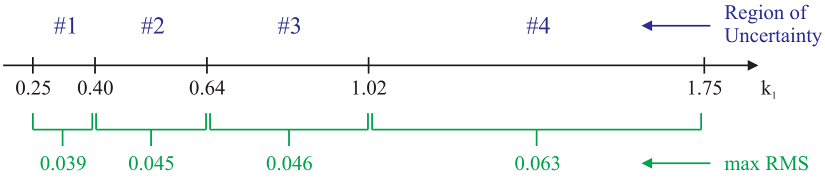


Figure 8.8: Maximum values of the RMS in steady state, for the closed-loop, for each region of the spring stiffness uncertainty.

control scheme, is depicted in Fig. 8.9.

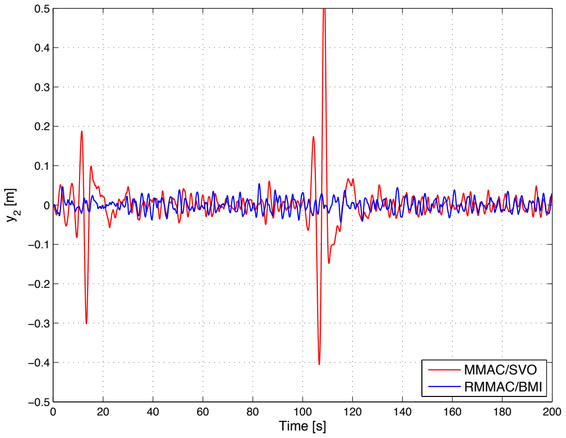


Figure 8.9: Output of the MSD plant, for forced instability and $k_1 = 0.25$ N/m, using the RMMAC/BMI and the MMAC/SVO-RMS control architectures.

The transient of the MMAC/SVO-RMS is still considerably larger than that of the RMMAC/BMI, whenever we force the model-mismatch instability. Nevertheless, we obtain a remarkable improvement when comparing these results with those of the MMAC/SVO in the previous simulations. In fact, not only the MMAC/SVO-RMS is able to select the appropriate controller in about 6 secs, but also it recovers from the forced instability in less than 8 secs.

8.3.4 MMAC/SVO-RMS: $k_1 = 1.75$ N/m - Forced Instability

We now consider that $k_1 = 1.75$ N/m. For this configuration, it was shown that the performance of the MMAC/SVO was larger than that of the RMMAC/BMI. The results obtained using also RMS considerations are depicted in Fig. 8.11 and Fig.

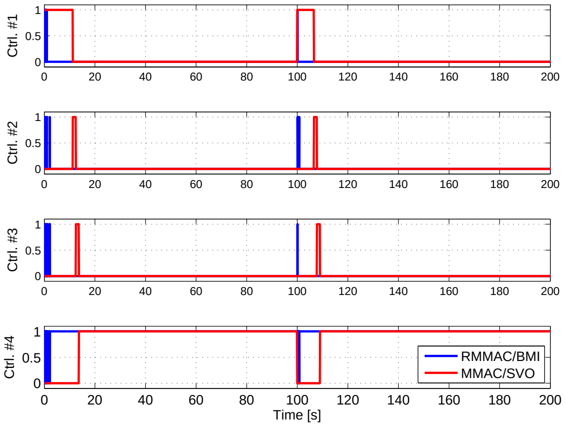


Figure 8.10: Controller selection, for forced instability and $k_1 = 0.25$ N/m.

8.12.

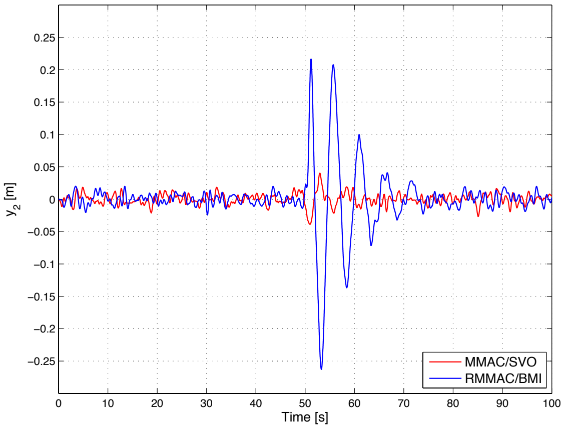


Figure 8.11: Output of the MSD plant, for forced instability and $k_1 = 1.75$ N/m, using the RMMAC/BMI and the MMAC/SVO-RMS control architectures.

The MMAC/SVO-RMS shows a considerably smaller transient when compared to that of the RMMAC/BMI. Indeed, it only requires one measurement to invalidate model #4, as happened with the MMAC/SVO algorithm without RMS considerations.

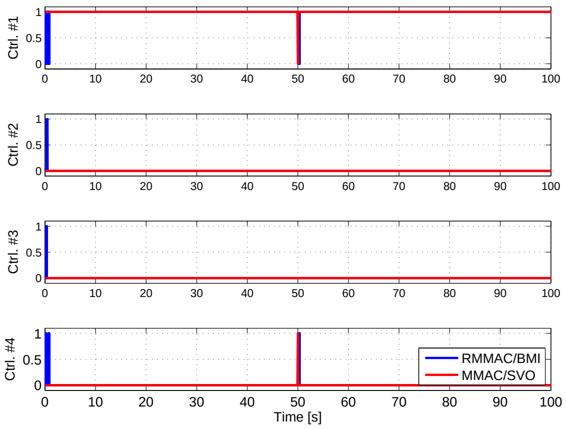


Figure 8.12: Controller selection, for forced instability and $k_1 = 1.75$ N/m.

8.3.5 MMAC/SVO-RMS: Time-Varying k_1

In the following simulation, we consider that the spring stiffness is time-varying and described by

$$k_1(k) = 1.00 + 0.75 \sin(\omega k T_s) \text{ N/m},$$

where

$$\omega = \frac{2\pi}{1000} \text{ rad/s},$$

as illustrated in Fig. 8.13. As previously, consider also that $\Gamma = 1$, which means that we are using the nominal values of the intensity and maximum magnitude of the exogenous disturbances.

Figure 8.14 depicts the results obtained for a typical Monte-Carlo run. In comparison with the RMMAC/BMI, the MMAC/SVO-RMS shows considerably larger transients, whenever the spring stiffness drifts from one region of uncertainty to another.

Figure 8.15 shows the controllers selected by each adaptive control scheme, at each time. The MMAC/SVO-RMS is, in general, slower than the RMMAC/BMI, in the identification of the correct region of uncertainty of k_1. Moreover, the MMAC/SVO was not able, in this MC simulation, to select LNARC #4, when the uncertain parameter was inside the corresponding uncertainty region. This happens because the LPV/BMI controller designed for $k_1 \in [0.4,\, 0.64]$ N/m displays a plausible performance, in terms of expected RMS of the output. Therefore, neither

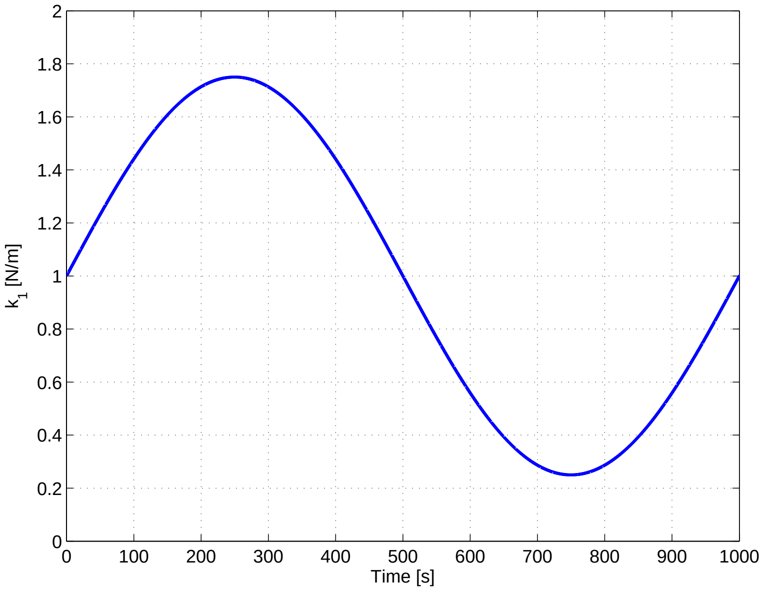


Figure 8.13: Time-varying spring stiffness, $k_1(\cdot)$.

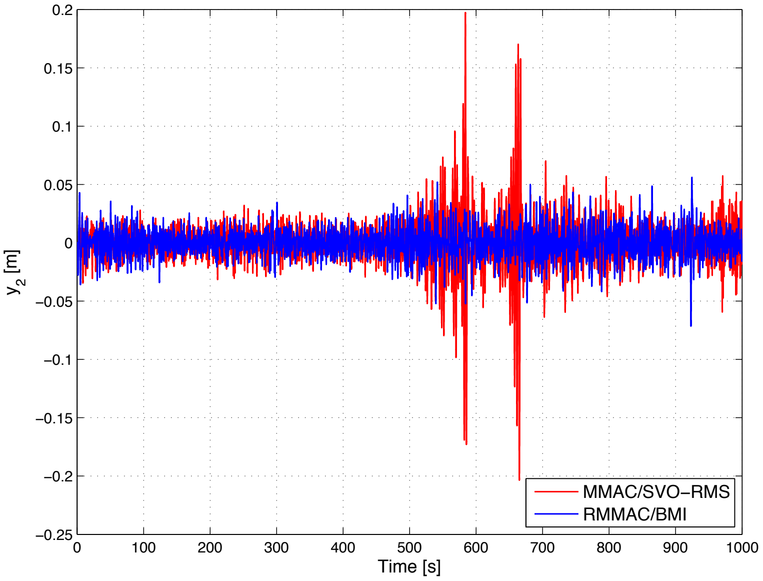


Figure 8.14: Output of the double MSD plant, for the time-varying spring stiffness depicted in Fig. 8.13.

the SVOs, nor the RMS analysis, are able to falsify region of uncertainty #3, and thus the controller designed to this region is used.

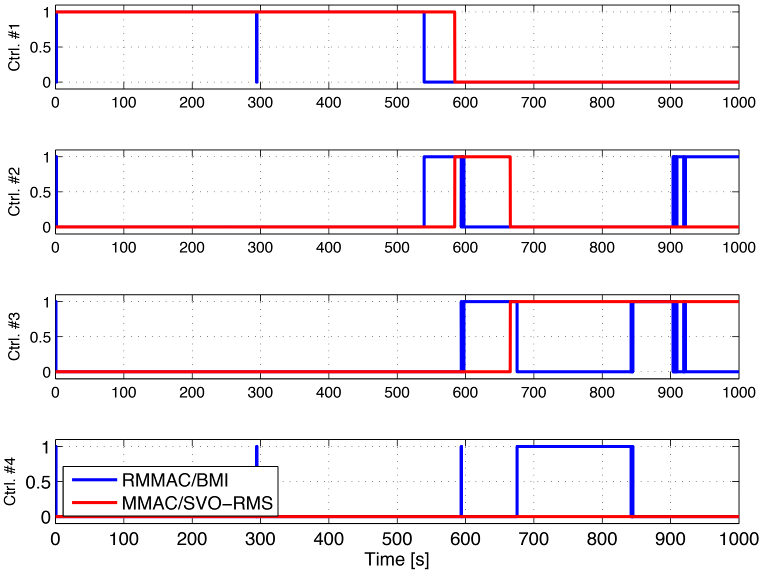


Figure 8.15: MMAC/SVO and RMMAC/BMI controller selection, for the time-varying spring stiffness depicted in Fig. 8.13.

Therefore, this simulation illustrates one of the weaknesses of the MMAC/SVO-RMS, regarding the time required to identify the region of uncertainty of the parameters. In terms of closed-loop behavior, this delay in the decision subsystem translates into large transients at the output of the plant.

8.3.6 MMAC/SVO-RMS: Time-Varying k_1 and Intensity of the Disturbances

Consider that the spring stiffness is time-varying and described by

$$k_1(k) = 1.00 + 0.75\sin(\omega k T_s) \text{ N/m},$$

as illustrated in Fig. 8.13, and where

$$\omega = \frac{2\pi}{1000} \text{ rad/s}.$$

However, let Γ follow the time-evolution depicted in Fig. 8.16.

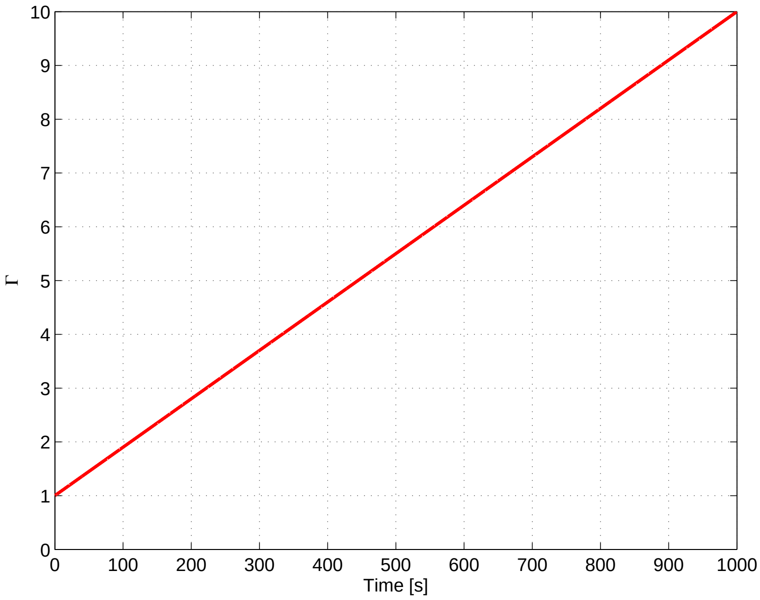


Figure 8.16: Time-varying magnitude of the disturbances.

Figure 8.17 depicts the results obtained for a typical Monte-Carlo run. Once again, the MMAC/SVO-RMS shows some large transients whenever the controllers are switched. Nevertheless, contrary to the RMMAC/BMI, it eventually selects the correct controller, as illustrated in Fig. 8.18. The controller selection of the Kalman filters was omitted, since the decision subsystem of the RMMAC/BMI is not able to converge to any controller for values of Γ greater than 3. Therefore, the RMMAC/BMI keeps switching, every sampling time, between the 4 available controllers.

At $t \approx 200$ secs, the RMS of the output is larger than expected, according to Fig. 8.8. Thus, we switch to the next available controller, which is LNARC #2. At about the same time, regions of uncertainty #3 and #4 are invalidated by the SVOs. Since the RMS of the output obtained with the LPV/BMI controller designed for $k_1 \in [0.4, 0.64]$ N/m – *i.e.*, controller #2 – is also larger than expected, due to the increase of the intensity of the disturbances, we switch back to controller #1.

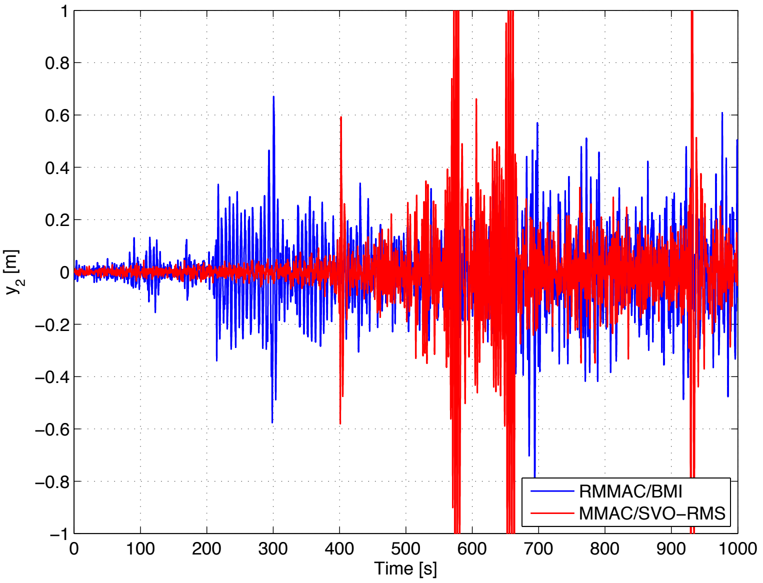


Figure 8.17: Output of the double MSD plant, for the time-varying spring stiffness depicted in Fig. 8.13 and for Γ as in Fig. 8.16.

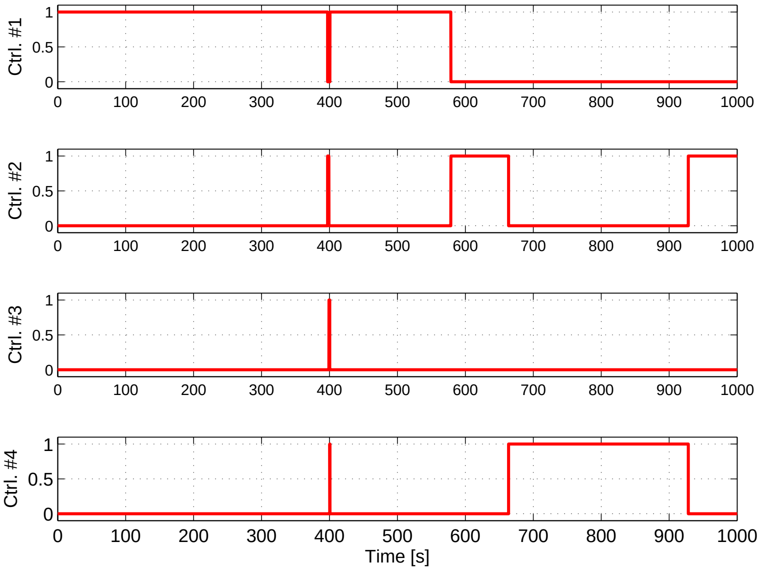


Figure 8.18: MMAC/SVO and RMMAC/BMI controller selection, for the time-varying spring stiffness depicted in Fig. 8.13 and for Γ as in Fig. 8.16.

8.3.7 MMAC/SVO-RMS: TV k_1 and Intensity of the Disturbances – No Disturbances' Model

As a last example, we consider that the model of the disturbances is not available during the design of the SVOs. Therefore, these dynamics are not taken into account by these observers, which intuitively means that they have no information regarding the bandwidth of the exogenous disturbances. It should be noted that this fact pronouncedly toughens the identification problem.

For the simulation, consider that the spring stiffness is time-varying and described by

$$k_1(k) = 1.00 + 0.75\sin(\omega k T_s) \text{ N/m},$$

as depicted in Fig. 8.13, and where

$$\omega = \frac{2\pi}{1000} \text{ rad/s}.$$

Moreover, let Γ follow the time-evolution depicted in Fig. 8.16.

Figure 8.17 depicts the results obtained. As expected, in this case, the MMAC/SVO-RMS shows larger transients than in the case where the model of the disturbances was known. Nevertheless, the MMAC/SVO-RMS is still able to eventually select the appropriate controller, even though the performance is deteriorated.

The controllers selected by the MMAC/SVO-RMS, at each time, are illustrated in 8.20. The controller selection of the Kalman filters was omitted, since the decision subsystem of the RMMAC/BMI is not able to converge to any controller for values of Γ greater than 3.

8.4 Conclusions

In this chapter, several simulations illustrated the applicability of the Multiple-Model Adaptive Control using Set-Valued Observers (MMAC/SVO) scheme to a difficult adaptive control problem, namely, the Mass-Spring-Dashpot (MSD) plant. In order to evaluate the behavior of the MMAC/SVO scheme, the performance

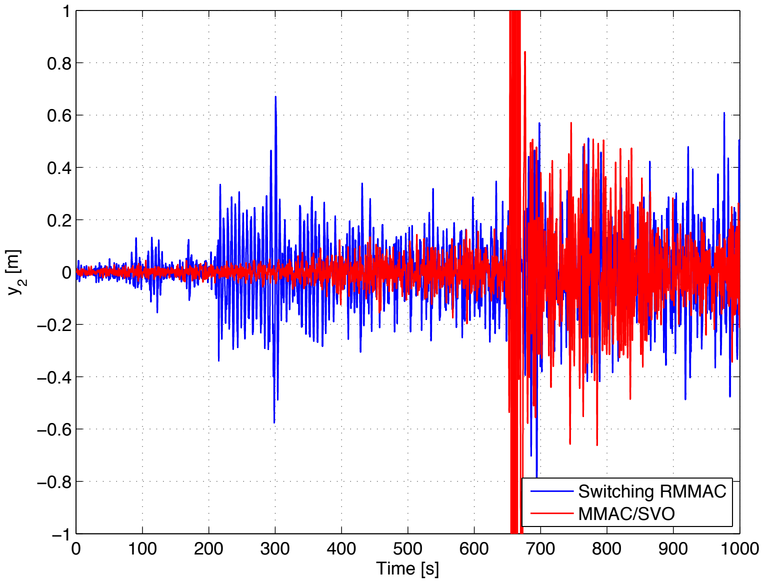


Figure 8.19: Output of the double MSD plant, for the time-varying spring stiffness depicted in Fig. 8.13 and for Γ as in Fig. 8.16, for the case where the model of the disturbances is not known a priori by the SVOs.

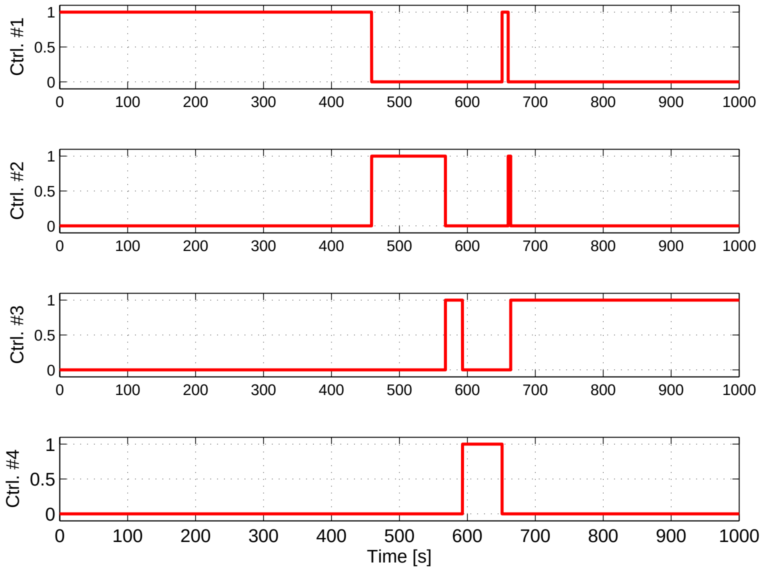


Figure 8.20: MMAC/SVO and RMMAC/BMI controller selection, for the time-varying spring stiffness depicted in Fig. 8.13 and for Γ as in Fig. 8.16, for the case where the model of the disturbances is not known a priori by the SVOs.

obtained with this method was compared to that attained by the Robust Multiple-Model Adaptive Control scheme with LPV/BMI controllers (RMMAC/BMI). However, it is stressed that the RMMAC/BMI and the MMAC/SVO are established under different assumptions. In particular, the RMMAC/BMI considers a stochastic setup, where some information regarding the probability density functions of the exogenous disturbances and measurement noise is available. Furthermore, the RMMAC/BMI assumes that the plant to be controlled is Linear Time-Invariant (LTI) or slowly time-varying. The MMAC/SVO, however, considers a deterministic setup, where no information is available regarding the exogenous disturbances and measurement noise, except that these signals are bounded. Moreover, the MMAC/SVO can handle time-varying plants and provides stability and performance guarantees under mild conditions on the input signals – see Chapter 7.

The results obtained indicate that the RMMAC/BMI typically yields faster responses to time-variations of the uncertain parameters of the plant. In terms of the output of the closed-loop system, this translates into smaller transients.

In order to overcome this problem, the RMS of the output of the closed-loop system was used to potentially invalidate controllers quicker. The Monte-Carlo simulations performed indicate that this method indeed improves significantly the performance of the overall system, leading to smaller transients.

However, using RMS considerations also has its own shortcomings. On the one hand, we can only invalidate controllers using such an approach, if the system has attained a steady-state. On the other hand, if the intensity of the disturbances is not known a priori, then it is not possible to compute the RMS of the output of the closed-loop system.

In summary, the MMAC/SVO architecture is endowed with important stability and performance guarantees, while yielding results in practice which are comparable to those obtained by the RMMAC/BMI, when applied to an MSD plant. Moreover, the Monte-Carlo simulations performed indicate that the MMAC/SVO is robust against variations on the uncertain parameters of the plant and on the intensity of the exogenous disturbances, and that the lack of knowledge regarding the dynamics of the disturbances does not significantly impact on the performance obtained. Nevertheless, the performance of the proposed architecture can be pronouncedly inferior, when compared to that obtained with adaptive control schemes based on stochastic assumptions on the exogenous disturbances, such as the RMMAC/BMI. Therefore, it is possible to improve the overall performance of the MMAC/SVO, by integrating this adaptive control scheme with other heuristics, based on the problem at hand.

Chapter 9

Conclusions and Future Directions

The main goal of this thesis was the development of a methodology to design robust adaptive controllers for uncertain Linear Parameter Varying (LPV) systems, with stability and performance guarantees. The strategy adopted was to divide the problem at hand into the design of a *decision* and a *control* subsystems.

For the control subsystem, a thorough methodology was developed that allows the synthesis of Linear Time-Invariant (LTI) robust controllers for uncertain LPV plants, with performance guarantees at least as good as those provided by the equivalent mixed-μ design, resorting to Bilinear Matrix Inequalities (BMIs). It was shown in simulation that, for systems with parametric real uncertainties, these controllers, referred to as LPV/BMI controllers, are able to significantly reduce the conservatism added by the μ approach. Moreover, the proposed methodology is also able to cope with time-variations of the uncertain parameters of the plant. Nevertheless, in reference to controller design, the LPV/BMI approach is considerably more demanding in terms of (off-line) computations, than the μ synthesis.

Regarding the decision subsystem, two solutions were proposed. The first one is referred to as the Stability Overlay (SO) for adaptive control of a class of nonlinear time-varying plants. The SO can be implemented in parallel with a wide range of "performance-based" adaptive control laws, *i.e.*, adaptive control laws that seek to improve closed-loop performance, but may be susceptible to instability in the presence of unaccounted model uncertainty. In this architecture, the performance-based adaptive control law designates candidate controllers based on performance considerations, while the SO supervises this selection based upon online robust stability considerations. A particular selection of a performance-based adaptive control law need not be specified. Rather, this selection can be from a wide range of adaptive control schemes. As an example, it was shown how to integrate the SO with the so-called Robust Multiple-Model Adaptive Control (RMMAC) method.

The second solution proposed for the decision subsystem is based upon *model falsification*. In particular, the work on Set-Valued Observers (SVOs) was extended to uncertain LPV plants, with special attention to the development of solutions to the main associated numerical and computational issues. The main idea in model falsification using SVOs is to invalidate dynamic uncertain LPV models, based on: a) the control input signals; b) the measured output of the plant; c) the a priori known bounds on the magnitude of the exogenous disturbances and measurement noise. The chief advantage of this approach is that the SVOs are able to perform such a task in an iterative manner. However, the on-line computational requirements of this technique can become prohibitive for plants with a large number of states.

The proposed model falsification approach using SVOs was applied both to robust Multiple-Model Adaptive Control (MMAC) and to Fault Detection and Isolation (FDI). In terms of robust control using SVOs, the main advantages are that:

a) we are able to provide guarantees, under mild assumptions, in terms of robust stability and performance of the closed-loop;

b) the model of the dynamics of the plant to be controlled can contain uncertain time-varying terms.

The architecture developed is referred to as MMAC/SVO.

Regarding FDI, the proposed methodology has the following advantages:

a) we are able to provide guarantees, under mild assumptions, that a given fault is *always* detected, and that false alarms are avoided;

b) the model of the dynamics of the plant to be diagnosed can contain uncertain time-varying terms.

The main disadvantage of this model falsification approach is the on-line computational burden associated with the SVOs.

9.1 Future Directions

Several issues related to the problem of robust adaptive control of uncertain LPV systems are yet to be addressed. In particular, some future research directions are enumerated in what follows:

a) The LPV/BMI controllers described in this thesis consider that the dynamics of the plant depend upon time-varying uncertain (real) parameters. Moreover, they also account for unmodeled dynamics, exogenous disturbances, and measurement noise. Nevertheless, it may happen that the plant also depends upon parameters which are *measured on-line.* Such information could be used to schedule the controller, in a similar manner to what is done in classical LPV control. Thus, a possible future direction of research is to extend the LPV/BMI controllers to the case where some of the plants' time-varying parameters are uncertain, while others are measured on-line.

b) An extension of the SO to handle nonlinear plants with finite escape time would be of interest, since it is clearly important to detect instability as soon as possible. Not only such an approach could be applied to a much larger class of dynamic systems, but also it would enhance the performance of the closed-loop system. A simple solution to this problem would be to "disqualify" controllers that lead the output of the plant to large values in a small amount of time. Such a formulation, however, may not be trivial, since we would have to guarantee that the sampling frequency is sufficiently high to detect such fast variations in the output.

c) If the bound on the amplitude of the disturbances is unknown, the performance of the closed-loop with the MMAC/SVO architecture may become considerably deteriorated. Therefore, it would be interesting to extend the

MMAC/SVO architecture to the case where the a priori bound on the magnitude of the disturbances can be decreased, in the light of the strategy used by the SO, *i.e.*, if the magnitude of the performance output is small enough, then we allow for the reuse of SVOs with smaller bounds on the magnitude of the disturbances.

d) The guarantees of convergence of the SVOs provided in this thesis require that a certain, potentially large, number of previous measurements is used. This fact may be responsible for a pronounced hike in terms of computations. However, for LTI systems, it is well known that we can describe the corresponding dynamics through a *coprime factorization*, with several interesting properties. In particular, under certain mild conditions, we can write the discrete LTI system G as

$$G = MN^{-1},$$

where M and N are also (discrete) LTI systems, but with arbitrary pole location. If the poles of these systems are placed near the origin, then we can design SVOs for M and N with typically faster convergence rates. As a shortcoming, instead of having one SVO for system G, we would have two, *i.e.*, one SVO for system M and another for system N. Nonetheless, the development of this design technique, in particular for time-varying systems, may lead to significant improvements in terms of computational requirements.

e) Further research on the design and implementation of SVOs is also needed. Indeed, it would be interesting to compare the MMAC/SVO using different types of SVOs. In particular, a comparison between the SVOs developed in this thesis and the ellipsoid-approach is needed, in order to evaluate the degree of conservatism added by each technique. It should be noticed that, on the one hand, the bounding ellipsoids typically require a much smaller computational burden, when compared to that of the SVOs developed in this thesis. On the other hand, these SVOs are potentially less conservative than the bounding ellipsoids.

f) Similarly to what was done with the SO, the MMAC/SVO architecture can be integrated with other adaptive control algorithms. Such an approach potentially leads to higher levels of performance, while guaranteeing stability and performance even in a worst-case scenario. Therefore, due to its *model falsification nature*, the MMAC/SVO architecture can be seen as a "safety net" for other adaptive control methodologies. However, in this case we would be able to guarantee not only stability, but also performance.

g) The SVOs considered in this thesis do not account for unknown time-delays in the input of the plant. Two possible solutions for this shortcoming are as follows. A simple approach is to consider that these time-delays can be treated as uncertainty in the input. Indeed, for small time-delays, we can add uncertainty on the input signal in order to model these effects. The second solution, which is possibly more interesting but also harder to implement, is to have several SVOs, each of which "tuned" for a particular time-delay. Thereafter, model falsification methods could be used not only to handle the time-delay, but also to identify it.

Bibliography

[1] S. Hergarten and F. Jansen. On the separation of timescales in spring-block earthquake models. *Nonlinear Processes in Geophysics*, 12:83–88, January 2005.

[2] N. Mitsui and K. Hirahara. Simple spring-mass model simulation of earthquake cycle along the nankai trough in southwest japan. *Pure & Applied Geophysics PAGEOPH*, 161(11-12), December 2004.

[3] M. Setareh. Floor vibration control using semi-active tuned mass sampers. *Canadian Journal of Civil Engineering*, 29(1):76–84, February 2002.

[4] S.J. Kim and J.W. Choi. Parametric uncertainty in controlling the vibration of a building. In *Proceedings of the 39th SICE Annual Conference*, pages 107–112, 2000.

[5] H. Nishimura and A. Kojima. Robust vibration isolation control for a multi-degree-of-freedom structure. In *Proceedings of the 1998 IEEE Conference on Control Applications (CCA)*, Trieste, Italy, September 1998.

[6] P.-Y. Sun and H. Chen. Multi-objective output-feedback suspension control on a half-car model. In *Proceedings of the 2003 IEEE Conference on Control Applications (CCA)*, volume 1, pages 290–295, June 2003.

[7] P. Gaspar, I. Szaszi, and J. Bokor. Design of robust controllers for active vehicle suspension using the mixed μ synthesis. *Vehicle System Dynamics*, 40(4):193–228, October 2003.

[8] A. W. Burton, A. J. Truscott, and P. E. Wellstead. Analysis, modeling and control of an advanced automotive self-leveling suspension system. In *IEE proceedings. Control theory and applications*, volume 142, pages 129–139, March 1995.

[9] M. Yamashita, K. Fujimori, C. Uhlik, R. Kawatani, and H. Kimura. h_∞ control of an automotive active suspension. In *Proceedings on the 29th Conference on Decision and Control (CDC)*, Honolulu, Hawaii, December 1990.

[10] D.A. Kienholz. Simulation of the zero-gravity environment for dynamic testing of structures. In *Proceedings of the 19th Space Simulation Conference*, October 1996.

[11] G.J. Balas, P.M. Young, and J.C. Doyle. μ based control design as applied to a large space structure: Control design for the minimast facility. Technical report, NASA CSI/GI final report, 1992.

[12] P.Y. Chu, B. Wie, B. Gretz, and C. Plescia. Approach to large space structure control system design using traditional tools. *Journal of Guidance, Control, and Dynamics*, 13(5):874–880, 1990.

[13] M.E. Campbell and S. Grocott. Parametric uncertainty model for control design and analysis. *IEEE Transactions on Control Systems Technology*, 7(1):85–96, January 1999.

[14] J. How, R. Glaese, S. Grocott, and D. Miller. Finite element model-based robust controllers for the middeck active control experiment (mace). *IEEE Transactions on Control Systems Technology*, 5(1):110–118, January 1997.

[15] J. How, S.R. Hall, and W.M. Haddad. Robust controllers for the middeck active control experiment using popov controller synthesis. *IEEE Transactions on Control Systems Technology*, 2(2):73–87, June 1994.

[16] M. Athans *et al.* The stochastic control of the F-8C aircraft using the multiple

model adaptive contro (MMAC) method – Part I: Equilibirum flight. *IEEE Trans. on Automatic Control*, 22(5):768–780, 1977.

[17] P. Maybeck and R. Stevens. Reconfigurable flight control via multiple model adaptive control methods. *IEEE Trans. on Aerospace and Electronic Systems*, 27(3):470–480, 1991.

[18] G. Schiller and P. Maybeck. Control of a large space structure using mmae/mmac techniques. *IEEE Trans. on Aerospace and Electronic Systems*, 33(4):1122–1130, 1997.

[19] P. Maybeck and J. Griffin. MMAE/MMAC control for bending with multiple uncertain parameters. *IEEE Trans. on Aerospace and Electronic Systems*, 33(3):903–912, 1997.

[20] B. Chaudhuri *et al.* Application of multiple-model adaptive control strategy for robust damping of interarea oscillations in power systems. *IEEE Trans. on Control Systems Technology*, 12(5):727–736, 2004.

[21] H. Tjabyadi *et al.* Vibration control of a cantilever beam using multiple model adaptive control. In *Proceedings of the American Control Conference*, pages 2907–2908, June/July 2004.

[22] R. N. Silva, J. M. Lemos, and L. M. Rato. Variable sampling adaptive control of a distributed collector solar field. *IEEE Trans. on Control Systems Technology*, 11(5):765–772, 2003.

[23] T. O. Santos, R. B. Caetano, J. M. Lemos, and F. J. Coito. Multipredictive adaptive control of arc welding trailing centerline temperature. *IEEE Trans. on Control Systems Technology*, 8(1):159–169, 2000.

[24] A. Karimi and I. D. Landau. Robust adaptive control of a flexible transmission system using multiple models. *IEEE Trans. on Control Systems Technology*, 8(2):321–331, 2000.

[25] J. Krause *et al.* Robustness studies in adaptive control. In *Proceedings of the IEEE Conference on Decision and Control*, pages 977–981, December 1983.

[26] S. Sastry and J. Sun. *Adaptive Control: Stability, Convergence and Robustness.* Prentice-Hall, NJ, USA, Englewood Cliffs, 1989.

[27] P. Ioannou and J. Sun. *Robust Adaptive Control.* Prentice Hall, Inc, 1996.

[28] I. Landau. *Adaptive Control: The Model Reference Approach.* Marcel Dekker, NY, USA, 1979.

[29] K. Narendra and A. Annaswamy. *Stable Adaptive Systems.* Prentice Hall, Inc, NJ, USA, 1988.

[30] K. Åström and B. Wittenmark. *Adaptive Control, 2nd ed.* Addison-Wesley, MA, USA, 1995.

[31] J. P. Hespanha, D. Liberzon, A. S. Morse, B. Anderson, T. Brinsmead, and F. de Bruyne. Multiple model adaptive control, part 2: switching. *Int. J. of Robust and Nonlinear Control,* Special Issue on Hybrid Systems in Control, 11(5):479–496, April 2001.

[32] P. Maybeck. *Stochastic Models, Estimation and Control*, volume I and II. Academic Press, USA, 1979 and 1982.

[33] B. Anderson and J. Moore. *Optimal Filtering.* Prentice-Hall, NJ, USA, 1979.

[34] Sajjad Fekri. *Robust adaptive MIMO control using multiple-model hypothesis testing and mixed-μ synthesis.* PhD thesis, Instituto Superior Tecnico, Lisbon, Portugal, December 2005.

[35] D. Lainiotis. Optimal adaptive estimation: Structure and parameter adaptation. *IEEE Trans. on Automatic Control*, 16(2):160–170, 1971.

[36] B. Anderson *et al.* Multiple model adaptive control, part 1: Finite controller coverings. *Int. J. of Robust and Nonlinear Control*, 10(11-12):909–929, 2000.

[37] B. Anderson *et al.* Multiple model adaptive control with safe switching. *Int. J. of of Adaptive Control and Signal Processing*, 15(5):445–470, 2001.

[38] L. Rato. *Controlo Comutado Baseado em Modelos Múltiplos.* PhD thesis, IST, Lisbon, Portugal, 2002.

[39] C. E. Rohrs, L. Valavani, M. Athans, and G. Stein. Robustness of continuous-time adaptive control algorithms in the presence of unmodeled dynamics. *IEEE Transactions on Automatic Control*, 30(9)(9), September 1985.

[40] A. Morse. Supervisory control of families of linear set-point controllers – part 2: Robustness. *IEEE Trans. on Automatic Control*, 42(11):1500–1515, 1997.

[41] M. Kuipers and P. Ioannou. Multiple model adaptive control with mixing. *IEEE Transactions on Automatic Control*, 55(8):1822–1836, February 2010.

[42] T. Hägglund and K. Åström. Supervision of adaptive control algorithms. *Automatica*, 36:1171–1180, 2000.

[43] R. Middleton *et al.* Design issues in adaptive control. *IEEE Trans. on Automatic Control*, 33(1):50–58, 1988.

[44] Peter M. Young, Matthew P. Newlin, and John C. Doyle. Practical computation of the mixed μ problem. In *American Control Conference*, pages 2190 –2194, jun. 1992.

[45] P.M. Young, M.P. Newlin, and J.C. Doyle. Computing bounds for the mixed-μ problem. *Int. J. of Robust and Nonlinear Control*, 5:573–590, 1995.

[46] G.J. Balas, J.C. Doyle, K. Glover, A. Packard, and R. Smith. *μ-Analysis and Synthesis Toolbox.* The Mathworks, June 2004.

[47] A. Packard and J. Doyle. The complex structured singular value. *Automatica*, 29(1):71–109, 1993.

[48] G.J. Balas and A. Packard. The structured singular value (μ) framework. In *The Control Handbook*, chapter 42, pages 671–687. CRC Press, 1996.

[49] P. Rosa, G. Balas, C.J. Silvestre, and M. Athans. On the synthesis of robust multiple-model adaptive controllers (RMMAC) using BMI/LPV controllers. In *Proceedings of the 2009 European Control Conference*, pages 1644–1649, 2009.

[50] M.G. Safonov and Tung-Ching Tsao. The unfalsified control concept and learning. *IEEE Transactions on Automatic Control*, 42(6):843–847, June 1997.

[51] D. Wang, A. Paul, M. Stefanovic, and M. Safonov. Cost-detectability and stability of adaptive control systems. In *Proceeding of the 44th IEEE International Conference on Decision and Control*, pages 3584–3589, December 2005.

[52] M. Jun and M. Safonov. Automatic pid tuning: An application of unfalsified control. In *Proceedings of the IEEE CCA/CACSD*, pages 328–333, 1999.

[53] S. Baldi, G. Battistelli, E. Mosca, and P. Tesi. Multi-model unfalsified adaptive switching supervisory control. *Automatica*, 46(2):249–259, 2010.

[54] D. Angeli and E. Mosca. Lyapunov-based switching supervisory control of nonlinear uncertain systems. *IEEE Transactions on Automatic Control*, 47:500–505, 2002.

[55] A. Morse. Supervisory control of families of linear set-point controllers – part 1: Exact matching. *IEEE Trans. on Automatic Control*, 41(10):1413–1431, 1996.

[56] J. Valente de Oliveira and J. M. Lemos. A comparsion of some adaptive-predictive fuzzy-control strategies. *IEEE Trans. on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, 30(1):138–145, 2000.

[57] M. Athans, S. Fekri, and A. Pascoal. Issues on robust adaptive feedback control. In *Preprints of the 16th IFAC World Congress, pp. 9-39*, 2005.

[58] S. Fekri, M. Athans, and A. Pascoal. Issues, progress and new results in robust adaptive control. *Int. Journal of Adaptive Control and Signal Processing*, 20:519–579, 2006.

[59] S. Fekri, M. Athans, and A. Pascoal. Robust multiple model adaptive control (RMMAC): A case study. *Int. Journal of Adaptive Control and Signal Processing*, 21(1):1–30, 2006.

[60] P. Rosa, M. Athans, S. Fekri, and C. Silvestre. Further evaluation of the RMMAC method with time-varying parameters. In *Proc. of the Mediterranean Conference on Automation and Control, MED07, Athens, Greece*, pages 1–6, June 2007.

[61] P. Rosa, M. Athans, S. Fekri, and C. Silvestre. Evaluation of the RMMAC/XI method with time-varying parameters and disturbance statistics. In *Proceedings of the Mediterranean Conference on Automation and Control, MED07, Athens, Greece*, June 2007.

[62] J.S. Shamma. *Analysis and Design of Gain Scheduled Control Systems.* PhD thesis, MIT, Mass., USA, 1988.

[63] C. Scherer and S. Weiland. *Course on Linear Matrix Inequalities in Control.* Dutch Institute of Systems and Control (DISC), 2005.

[64] J. VanAntwerp and R. Braatz. A tutorial on linear and bilinear matrix inequalities. *Journal of Process Control*, 10:363–385, 2000.

[65] J. VanAntwerp, R. Braatz, and N. Sahinidis. Globally optimal robust control for systems with time-varying nonlinear perturbations. *Computers and Chemical Engineering, Supplement 1*, 21:125–130, 1997.

[66] S. Kanev, C. Scherer, M. Verhaegen, and B. Schutter. Robust output-feedback controller design via local BMI optimization. In *Automatica 40*, 2004.

[67] S. Kanev. *Robust Fault-Tolerant Control.* PhD thesis, University of Twente - The Netherlands, 2004.

[68] I. Al-Shyoukh and J.S. Shamma. Switching supervisory control using calibrated forecasts. In *46th IEEE Conference on Decision and Control*, pages 705–716, December 2007.

[69] P. Rosa, J.S. Shamma, C.J. Silvestre, and M. Athans. Stability overlay for adaptive control laws applied to linear time-invariant systems. In *Proceedings of the 2009 American Control Conference*, pages 1934–1939, June 2009.

[70] P. Rosa, J.S. Shamma, C.J. Silvestre, and M. Athans. Stability overlay for linear and nonlinear time-varying plants. In *Proceedings of the 48th IEEE Conference on Decision and Control*, pages 2435–2440, December 2009.

[71] K. Poolla, P. Khargonekar, A. Tikku, J. Krause, and K. Nagpal. A time-domain approach to model validation. *IEEE Trans. on Automatic Control*, 39(5):951–959, 1994.

[72] Jean-Pierre Aubin and Hélène Frankowska. *Set-Valued Analysis (Modern Birkhäuser Classics)*. Birkhäuser Boston, 2008.

[73] Georgi Smirnov. *Introduction to the Theory of Differential Inclusions (Graduate Studies in Mathematics)*. American Mathematical Society, 1st edition, 2001.

[74] F. Blanchini. Set invariance in control. *Automatica*, 35:1747–1767, 1999.

[75] H. Lin, G. Zhai, and P. J. Antsaklis. Set-valued observer design for a class of uncertain linear systems with persistent disturbance and measurement noise. *Int. J. Control*, 76(16):1644–1653, 2003.

[76] J.S. Shamma and Kuang-Yang Tu. Set-valued observers and optimal disturbance rejection. *IEEE Transactions on Automatic Control*, 44(2):253–264, 1999.

[77] K. Zhou, J.C. Doyle, and K. Glover. *Robust Optimal Control.* Prentice Hall, 1996.

[78] S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control: Analysis and Design, 2nd Ed.* John Wiley and Sons, 2005.

[79] J.S. Shamma and M. Athans. Guaranteed properties of gain scheduled control for linear parameter-varying plants. *Automatica*, 27(3):559 – 564, 1991.

[80] J.S. Shamma and M. Athans. Analysis of gain scheduled control for nonlinear plants. *Automatic Control, IEEE Transactions on*, 35(8):898 –907, aug 1990.

[81] D. Xiong. *Stability Analysis and Controller Synthesis of Linear Parameter Varying Systems.* PhD thesis, The University of Texas at Austin, USA, 1998.

[82] P.M. Young, M.P. Newlin, and J.C. Doyle. μ analysis with real parametric uncertainty. In *Proceedings of the 30th IEEE Conference on Decision and Control*, pages 1251 –1256 vol.2, dec. 1991.

[83] P.M. Young. Controller design with mixed uncertainties. In *American Control Conference*, pages 2333–2337, Baltimore, Maryland, June 1994.

[84] P. Blue and S. Banda. D-K Iteration with optimal scales for systems with time-varying and time-invariant uncertainties. In *Proceedings of the American Control Conference, Albuquerque, New Mexico*, pages 3967–3971, June 1997.

[85] M. James, T. Bas, J.S. Shamma, and D. Xiong. Set-valued methods for linear parameter varying systems. *Automatica*, 35(6):1081–1089, 1999.

[86] F. Wu, X. Yang, A. Packard, and G. Becker. Induced l2 norm control for LPV systems with bounded parameter variation rates. *International Journal of Robust and Nonlinear Control*, 6(9/10):983–998, 1996.

[87] J.S. Shamma. Robust stability with time-varying structured uncertainty. *Automatic Control, IEEE Transactions on*, 39(4):714 –724, apr 1994.

[88] J.S. Shamma. Optimization of the l infin;-induced norm under full state feedback. *Automatic Control, IEEE Transactions on*, 41(4):533 –544, apr 1996.

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

[90] F. Wu. *Control of Linear Parameter Varying Systems.* PhD thesis, University of California at Berkley, 1995.

[91] L. El Ghaoui and S. I. Niculescu, editors. *Advances in Linear Matrix Inequality Methods in Control.* Society for Industrial and Applied Mathematics, SIAM, Philadelphia, PA, 1999.

[92] M. Chilali and P. Gahinet. H_∞ design with pole placement constraints an LMI approach. *IEEE Transactions on Automatic Control*, 41(3):358–367, 1996.

[93] P. Seiler, U. Topcu, A. Packard, and G. Balas. Parameter-dependent lyapunov functions for linear systems with constant uncertainties. *IEEE Transactions on Automatic Control*, 54(10):2410 –2416, oct. 2009.

[94] W.P. Dayawansa and C.F. Martin. A converse Lyapunov theorem for a class of dynamical systems which undergo switching. *IEEE Transactions on Automatic Control*, 44(4):751–760, April 1999.

[95] W.J. Rugh and J.S. Shamma. A survey of research on gain-scheduling. *Automatica*, 36:1401–1425, 2000.

[96] O. Toker and H. Ozbay. On the np-hardness of solving bilinear matrix inequalities and simultaneous stabilization with static output feedback. In *Proceedings of the American Control Conference,Seattle, Washington*, pages 2525–2526, June 1995.

[97] M. Kočvara and M. Stingl. *PENBMI User's Guide (Version 2.1), www.penopt.com*, 2006.

[98] M. Kočvara and M. Stingl. Pennon - a code for convex nonlinear and semidefinite programming. In *Opt. Methods and Software*, 2003.

[99] M.G. Safonov, K.C. Goh, and J.H. Ly. Control system synthesis via bilinear matrix inequalities. In *American Control Conference*, volume 1, pages 45–49, jun. 1994.

[100] M. Kuipers and P. Ioannou. Practical robust adaptive control: Benchmark example. In *Proceedings of the 2008 American Control Conference*, pages 5168–5173, 2008.

[101] B. Wie and D.S. Bernstein. Benchmark problems for robust control design. *Journal of Guidance, Control, and Dynamics*, 15(5):1057–1059, September-October 1992.

[102] J. F. Vasconcelos, M. Athans, S. Fekri, C. Silvestre, and P. Oliveira. Stability- and performance-robustness tradeoffs: Mimo mixed-μ vs. complex-μ design. *International Journal of Robust and Nonlinear Control*, 19:259–294, 2009.

[103] J.M. Lourenço and J.M. Lemos. Predictive adaptive control of plants with online structural changes based on multiple models. *Int. Journal of Adaptive Control and Signal Processing*, 22:774–794, 2008.

[104] B.D.O. Anderson, T.S. Brinsmead, F. De Bruyne, J. Hespanha, D. Liberzon, and A.S. Morse. Multiple model adaptive control. part 1: Finite controller coverings. *International Journal of Robust and Nonlinear Control*, 10(11-12):909–929, 2000.

[105] I. Al-Shyoukh and J.S. Shamma. Switching supervisory control using calibrated forecasts. *IEEE Trans. on Automatic Control*, 54(4):705–716, 2009.

[106] P. Rosa, J.S. Shamma, C. Silvestre, and M. Athans. Stability overlay for adaptive control laws. *Automatica*, 47(5):1007 – 1014, May 2011.

[107] M.G. Safonov and T.-C. Tsao. *The unfalsified control concept: A direct path from experiment to controller. In B.A. Francis and A.R. Tannenbaum, eds.,* Feedback Control, Nonlinear Systems and Complexity. Springer-Verlag, Berlin, 1995.

[108] R. Wang, A. Paul, M. Stefanovic, and M.G. Safonov. Cost-detectability and stability of adaptive control systems. *Int. J. Robust and Nonlinear Control*, 17(5–6):549–561, 2007.

[109] C. Manuelli, S.G. Cheong, E. Mosca, and M.G. Safonov. Stability of unfalsified adaptive control with non scli controllers and related performance under different

prior knowledge. In *Proc. of the 2007 European Control Conference*, pages 702–708, Kos, Greece, July 2007.

[110] G. Manuelli, E. Mosca, M.G. Safonov, and P. Tesi. Unfalsified virtual reference adaptive switching control of plants with persistent disturbances. In *Proc. IFAC World Congress*, pages 8925–8930, Seoul, Korea, July 2008.

[111] M.W. Chang and M.G. Safonov. Unfalsified adaptive control: The benefit of bandpass filters. In *AIAA Guidance, Navigation and Control Conf. and Exhibit*, Honolulu, HI, August 2008.

[112] M. Stefanovic and M.G. Safonov. Safe adaptive switching control: Stability and convergence. *IEEE Trans. on Automatic Control*, 53(9):2012–2021, 2008.

[113] M. Fu and B.R. Barmish. Adaptive stabilization of linear systems via switching control. *IEEE Trans. on Automatic Control*, AC-31(12):1097–1103, 1986.

[114] A.S. Morse, D.Q. Mayne, and G.C. Goodwin. Applications of hysteresis switching in parameter adaptive control. *IEEE Trans. on Automatic Control*, 37(9):1343–1354, 1992.

[115] B. Martenson. The order of any stabilizing regulator is sufficient a priori information for adaptive stabilization. *Systems and Control Letters*, 6(2):87–91, 1985.

[116] Jo ao P. Hespanha and A. Stephen Morse. Switching between stabilizing controllers. *Automatica*, 38(11):1905 – 1917, 2002.

[117] Isaac Kaminer, Antonio M. Pascoal, Pramod P. Khargonekar, and Edward E. Coleman. A velocity algorithm for the implementation of gain-scheduled controllers. *Automatica*, 31(8):1185 – 1191, 1995.

[118] Mayuresh V. Kothare, Peter J. Campo, Manfred Morari, and Carl N. Nett. A unified framework for the study of anti-windup designs. *Automatica*, 30(12):1869 – 1883, 1994.

[119] Matthew C. Turner and Daniel J. Walker. Linear quadratic bumpless transfer. *Automatica*, 36(8):1089 – 1101, 2000.

[120] S.F. Graebe and A.L.B. Ahlen. Dynamic transfer among alternative controllers and its relation to antiwindup controller design. *Control Systems Technology, IEEE Transactions on*, 4(1):92 –99, Jan. 1996.

[121] K. Zhou and J.C. Doyle. *Essentials of Robust Control*. Prentice Hall, September 1997.

[122] J. P. Hespanha, D. Liberzon, D. Angeli, and E. D. Sontag. Nonlinear norm-observability notions and stability of switched systems. *IEEE Transactions on Automatic Control*, 50(2):154–168, February 2005.

[123] R. A. Garcia and J. L. Mancilla-Aguilar. State-norm estimator of switched nonlinear systems. In *2002 American Control Conference*, 2002.

[124] P. Rosa, J.S. Shamma, C.J. Silvestre, and M. Athans. Integration of the stability overlay (SO) with the robust multiple-model adaptive control (RMMAC). In *Proceedings of the Mediterranean Conference on Automation and Control, MED09, Thessaloniki, Greece*, June 2009.

[125] F.D. Bianchi and R.S. Sánchez-Peña. Robust identification/invalidation in an LPV framework. *Int. J. of Robust and Nonlinear Control*, 20:301–312, 2010.

[126] Mohammad-Ali Massoumnia. *A Geometric Approach to Failure Detection and Identification in Linear Systems*. PhD thesis, Massachusetts Institute of Technology, 1986.

[127] M. A. Massoumnia. Geometric approach to the synthesis of failure detection filters. *IEEE Trans. on Automatic Control*, 31(9):839–846, 1986.

[128] S. Longhi and A. Moteriù. Fault detection for linear periodic systems using a geometric approach. *IEEE Trans. on Automatic Control*, 54(7):1637–1643, 2009.

[129] J. Bokor and G. Balas. Detection filter design for LPV systems – a geometric approach. *Automatica*, 40:511–518, 2004.

[130] A. Edelmayer, J. Bokor, and L. Keviczky. An $\mathcal{H}_\infty$ approach to robust detection of failures in dynamical systems. In *Proceedings of the 33rd Conference on Decision and Control, Lake Buena Vista, FL, USA*, volume 3, pages 3037–3039, 1994.

[131] P.M Frank and X. Ding. Survey of robust residual generation and evaluation methods in observer-based fault detection systems. *Journal of Process Control*, 7(6):403–424, 1997.

[132] H. Niemann and J. Stoustrup. Fault diagnosis for non-minimum phase systems using h_∞ optimization. In *Proceedings of the American Control Conference, Arlington, VA, USA*, June 2001.

[133] 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*, 13:105–119, 2005.

[134] E. G. Jr. Collins and S. Tinglun. Robust l_1 estimation using the popov–tsypkin multiplier with application to robust fault detection. *Int. J. Control*, 74(3):303–313, 2001.

[135] R.S. Mangoubi, B.D. Appleby, G.C. Verghese, and W.E. Vander Velde. A robust failure detection and isolation algorithm. In *Proceedings of the 34th Conference on Decision and Control, New Orleans, USA*, volume 3, pages 2377–2382, 1995.

[136] R.J. Patton and J. Chen. Observer-based fault detection and isolation: Robustness and applications. *Control Engineering Practice*, 5(5):671–682, 1997.

[137] Andres Marcos Esteban. *Aircraft Applications of Fault Detection and Isolation Techniques*. PhD thesis, University of Minnesota, 2004.

[138] R. Isermann. Supervision, fault-detection and fault-diagnosis methods - an introduction. *Control Engineering Practice*, 5(5):639–652, 1997.

[139] M. Blanke, R. Izadi-Zamanabadi, S.A. Bogh, and C. Lunau. Fault tolerant control systems – a holistic view. *Control Engineering Practice*, 5(5):693–702, 1997.

[140] R. Vidal, A. Chiuso, and S. Soatto. Observability and identifiability of jump linear systems. In *Proceedings of the 41st IEEE Conference on Decision and Control*, pages 3614–3619, 2002.

[141] H. Lou and P. Si. The distinguishability of linear control systems. *Nonlinear Analysis: Hybrid Systems*, 3:21–38, 2009.

[142] P. Rosa, C.J. Silvestre, J.S. Shamma, and M. Athans. Multiple-model adaptive control with set-valued observers. In *Proceedings of the 48th IEEE Conference on Decision and Control*, December 2009.

[143] P. Rosa, C.J. Silvestre, J.S. Shamma, and M. Athans. Fault detection and isolation of LTV systems using set-valued observers. In *Proceedings of the 49th IEEE Conference on Decision and Control*, December 2010.

[144] M.S. Grewal and K. Glover. Identifiability of linear and nonlinear dynamical systems. *IEEE Trans. on Automatic Control*, 21(6):833–837, 1976.

[145] J.J. DiStefano III. On the relationships between structural identifiability and the controllability, observability properties. *IEEE Trans. on Automatic Control*, 22:652, 1977.

[146] J.J. DiStefano III and C. Cobelli. On parameter and structural identifiability: Nonunique observability/reconstructibility for identifiable systems, other ambiguities and new definitions. *IEEE Trans. on Automatic Control*, 25(4):830–833, 1980.

[147] E. Walter, Y. Lecourtier, and J. Happel. On the structural output distinguishability of parametric models, and its relations with structural identifiability. *IEEE Trans. on Automatic Control*, 29(1):56–57, 1984.

[148] M. Baglietto, G. Battistelli, and L. Scardovi. Active mode observation of switching systems based on set-valued estimation of the continuous state. *Int. J. of Robust and Nonlinear Control*, 19:1521–1540, 2009.

[149] S. Keerthi and E. Gilbert. Computation of minimum-time feedback control laws for discrete-time systems with state-control constraints. *IEEE Transactions on Automatic Control*, 32(5):432–435, 1987.

[150] R.T. Rockefeller. *Convex Analysis.* Princeton University Press, Princeton, 1970.

[151] H. Witsenhausen. Sets of possible states of linear systems given perturbed observations. *IEEE Trans. on Automatic Control*, 13(5):556–558, 1968.

[152] F. Schweppe. Recursive state estimation: Unknown but bounded errors and system inputs. *IEEE Trans. on Automatic Control*, 13(1):22–28, Feb 1968.

[153] F. Schweppe. *Uncertain Dynamic Systems.* Prentice-Hall, USA, 1973.

[154] M. Milanese and A. Vicino. Optimal estimation theory for dynamic systems with set membership uncertainty: An overview. *Automatica*, 27(6):997–1009, 1991.

[155] F. Yang and L. Yongmin. Set-membership filtering for discrete-time systems with nonlinear equality constraints. *Automatic Control, IEEE Transactions on*, 54(10):2480–2486, oct. 2009.

[156] P. Rosa, C.J. Silvestre, J.S. Shamma, and M. Athans. Fault detection and isolation of an aircraft using set-valued observers. In *18th IFAC Symposium on Automatic Control in Aerospace*, September 2010.

[157] H. Le Verge. A note on chernikova's algorithm. Technical report, RR 635, IRISA, IRISA, Campus de Beaulieu, 35042 Rennes Cedex, 1992.

[158] R. Meyer and C. Burros. A unified analysis of multirate and periodically time-varying digital filters. *IEEE Transactions on Circuits and Systems*, CAS-22(3):162–168, 1975.

[159] X. Zhang, T. Parisini, and M.M. Polycarpou. Sensor bias fault isolation in a class of nonlinear systems. *IEEE Trans. on Automatic Control*, 50(3):370–376, March 2005.

[160] J. Freudenberg and D. Looze. Right-half plane zeros and poles and design tradeoffs in feedback systems. *Automatic Control, IEEE Transactions on*, 30(6):555–565, 1985.

[161] A.S. Willsky. A survey of design methods for failure detection in dynamic systems. *Automatica*, 12:601–611, 1976.

[162] M. Blanke, M. Staroswiecki, and N.E. Wu. Concepts and methods in fault-tolerant control. In *Proceedings of the American Control Conference, Arlington, VA, USA*, June 2001.

[163] R. Mattone and A. De Luca. Relaxed fault detection and isolation: An application to a nonlinear case study. *Automatica*, 42:109–116, 2006.

[164] P.M Frank and X. Ding. Frequency domain approach to optimally robust residual generation. *Automatica*, 30(5):789–804, 1994.

[165] G. Besançon. High-gain observation with disturbance attenuation and application to robust fault detection. *Automatica*, 39:1095–1102, 2003.

[166] N. Meskin and K. Khorasani. Fault detection and isolation of discrete-time markovian jump linear systems with application to a network of multi-agent systems having imperfect communication channels. *Automatica*, 45:2032–2040, 2009.

[167] D. Wang, W. Wang, and P. Shi. Robust fault detection for switched linear systems with state delays. *IEEE Trans. on Automatic Control*, 39(3):800–805, 2009.

[168] S. Narasimhan, P. Vachhani, and R. Rengaswamy. New nonlinear residual feedback observer for fault diagnosis in nonlinear systems. *Automatica*, 44:2222–2229, 2008.

[169] J. Chen, R.J. Patton, and H. Zhang. Design of unknown input observers and robust fault detection filters. *Int. J. Control*, 63(1):85–105, 1996.

[170] X. Li and K. Zhou. A time domain approach to robust fault detection of linear time-varying systems. *Automatica*, 45:94–102, 2009.

[171] A. Fujimori and L. Ljung. Model identification of linear parameter varying aircraft systems. *J. of Aerospace Engineering*, 28:337–346, 2006.