ROBUST AND REDUCED ORDER H-INFINITY FILTERING

VIA LMI APPROACH AND ITS APPLICATION TO FAULT DETECTION

A Dissertation by

Young-Man Kim

Master of Science, KyungPook National University, 1999

Bachelor of Arts, KyungHee University, 1995

Submitted to the Department of Electrical and Computer Engineering
and the faculty of the Graduate School of
Wichita State University
in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

May 2006

ROBUST AND REDUCED ORDER H-INFINITY FILTERING

VIA LMI APPROACH AND ITS APPLICATION TO FAULT DETECTION

I have examined the final copy of this Dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirement for the degree of Doctor of Philosophy with a major in Control

_____

John M. Watkins, Committee Chair

We have read this Dissertation
and recommend its acceptance:

_____

M. Edwin Sawan, Committee Member

_____

Larry D. Paarmann, Committee Member

_____

Hyuck M Kwon, Committee Member

_____

Behnam Bahr, Committee Member

Accepted for the College of Engineering

_____

Zulma Toro-Ramos, Dean

Accepted for the Graduate School

_____

Susan Kovar, Dean

ACKNOWLEDGEMENTS

"*For God so loved the world that He gave His only begotten Son, that whoever believes in Him should not perish but have eternal life. (John 3:16)"*

I would like to thank for my advisor, Dr. John M. Watkins, for his thoughtful, patient guidance and support. Thanks are also due to Dr. M. Edwin Sawan. Together their advice and selfless role modeling has contributed to my professional and personal development. I would also like to extend my gratitude to members of my committee, Dr. Larry D. Paarmann, Dr. Hyuck M. Kwon, and Dr. Behnam Bahr for their helpful comments and suggestions on all stages of this research. I also want to thank for my parents and family. Without their encouragement, I could not finish this research. Especially, I am thankful for my wife, Mi-Ok Kim, her continuous prayer and warm love. All the glory and honor to our returning Lord and Savior, Jesus Christ.

"Surely, I am coming quickly, Amen. Even so, come Lord Jesus! (Revelation 22:20)"

ABSTRACT

The objective of this dissertation is to develop a practical methodology for designing full and reduced order H_∞ filter for plants with polytopic model uncertainty. Because the polytopic model description is convex, it is amenable to a Linear Matrix Inequality (LMI) formulation. Reduced order filters are desirable in applications where fast data processing is necessary. To improve robustness to model uncertainties, this dissertation reformulates an H_2 filter design technique as a reduced order H_∞ filter design methodology. Lyapunov functions are replaced with parameter-dependent Lyapunov functions to provide less conservative results. As the problem is formulated as an LMI, an admissible filter with suitable dynamic behavior can be obtained from the solution of a convex optimization problem. The advantages of this approach over earlier approaches are highlighted in a simple computational example.

This filtering technique is used to design a fault detection filter. Robust fault detection filter (RFDF) design is formulated as a multi-objective H_∞ optimization for a polytopic uncertain system. The order of the RFDF is reduced using LMI techniques and the detection performance is compared with the full order filter. An adaptive threshold is used to reduce the number of false alarms. Examples are presented to illustrate effectiveness of the order reduction.

TABLE OF CONTENTS

Chapter Page

1. INTRODUCTION 1
   1.1 Background and Motivation 1
   1.2 Outline 2

2. LITERATURE REVIEW 4
   2.1 Linear Matrix Inequality (LMI) 4
      2.1.1 Definition 5
      2.1.2 Usefulness of LMI 6
      2.1.3 Generic Problem 7
      2.1.4 Convexity in LMI 7
   2.2 Parametrized Linear Matrix Inequality (PLMI) 8
   2.3 Useful Tools 10
      2.3.1 Congruence Transformation 10
      2.3.2 Finsler's Lemma 10
      2.3.3 Linearly Parametrized Matrix Inequality over the Unit Simplex Γ 11
      2.3.4 BRL (Bounded Real Lemma) 12

3. ROBUST AND REDUCED ORDER H-INFINITY FILTERING VIA LMI APPROACH .. 13
   3.1 Preliminaries 13
   3.2 Polytopic Covering Approach 19
   3.3 More Practical Approach 22
   3.4 Example 29

4. ROBUST AND REDUCED ORDER FAULT DETECTION FILTER DESIGN VIA LMI APPROACH 32
   4.1 Introduction 32
   4.2 Problem Formulation 35
      4.2.1 Residual Generation 35
      4.2.2 Residual Evaluation 44
   4.3 Main Results 46
   4.4 Examples 56
      4.4.1 Example 1 56
      4.4.2 Example 2 64

5. SUMMARY AND FUTURE WORK 68
   5.1 Summary 68
   5.2 Future Work 69

TABLE OF CONTENTS (continued)

Chapter Page

LIST OF REFERENCES ……..…… 70

APPENDICES …….. 77
A. Matlab Code to Synthesize the Full Order Filter for Table 3 ……...…… 78
B. Matlab Code to Synthesize the Reduced Order Filter for Table 3 …….. 85
C. Matlab Code to Design the 6^{th} Order RFDF for 4.4.2.Example 2 …..…… 92
D. Simulation Diagram for 4.4.2 Example 2 ……..…….. 104

LIST OF TABLES

TABLE Page

1. Performance Comparison According to Each Uncertainty Type …… 30

2. Filter Synthesis ……..…….…….…… 31

3. Performance Comparison Between [15] and (3.3.32) …….…….…….....……... 31

4. Comparison of the Detection Time of RFDF with Different Orders……...……. 58

LIST OF FIGURES

Figure Page

1. General filtering configuration …..……..……...……..…… 13

2. General structure of robust residual generation and decision making of model-based fault diagnosis …….. 34

3. General structure of residual generator ……..…….. 36

4. Block diagram for residual r(t) ……..…….. 39

5. Block diagram for residual error $\tilde{r}(t)$ with reference model ……..…… 42

6. Block diagram for residual error $\tilde{r}(t)$ with reference model and tracking filter……. 43

7. Full order response, ($\alpha_d = \alpha_f = \alpha_u = 1$) ……. 59

8. Full order response, ($\alpha_d = \alpha_f = 1, \alpha_u = 100$) …….. 60

9. Reduced order (k=3) response, ($\alpha_d = \alpha_f = \alpha_u = 1$) …….. 61

10. Reduced order (k=2) response, ($\alpha_d = \alpha_f = \alpha_u = 1$) ……. 62

11. Reduced order (k=1) response, ($\alpha_d = \alpha_f = \alpha_u = 1$) ……..... 63

12. Full order response (k=8) ……..… 66

13. Reduced order response (k=7) ……... 66

14. Reduced order response (k=6) ……..…… 67

15. Reduced order response (k=5) ……... 67

LIST OF ABBREVIATIONS

ARI	Algebraic Riccati Inequality
ARE	Algebraic Riccati Equation
BRL	Bounded Real Lemma
LMI	Linear Matrix Inequality
PLMI	Parametrized Linear Matrix Inequality
RFDF	Robust Fault Detection Filter
MIMO	Multiple Input Multiple Output

LIST OF SYMBOLS

R	field of real numbers
R^n	real vector with dimension n
$R^{m\times n}$	real matrix with dimension $m\times n$
S^n	symmetric matrix with dimension n
$B^{\perp}$	orthogonal complement of matrix B
B^T	transpose of matrix B for terms that are induced by symmetry, e.g., $(*)\begin{bmatrix} S+(*) & * \\ M & Q \end{bmatrix} K^T \equiv K\begin{bmatrix} S+S^T & M^T \\ M & Q \end{bmatrix} K^T$
$0_{n\times m}$	$n\times m$ matrix with all zero elements
col( )	operator that stacks elements into a column vector
$\bigcup$	union of all elements
Unit simplex	$\Gamma = \left\{ (\alpha_1, \cdots, \alpha_S) : \sum_{i=1}^{S} \alpha_i = 1, \alpha_i \geq 0 \right\}$
N(A)	nullspace of matrix A
RH_∞	subspace of L_∞ with real and rational functions that are analytic and bounded in the open right-half plane
L_∞	set of functions bounded on $j\omega$-axis including at ∞

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

Filters that estimate the state variables of a system are important tools for control & signal processing applications. Early work in the area assumed that the system dynamics were known and external disturbances were white noise with known statistical properties [1]. During the past three decades, Kalman filtering techniques have found widespread application in guidance, navigation and control problems. But recently, H_∞ filtering techniques have received considerable attention. In contrast to traditional Kalman filters, H_∞ filters do not require knowledge of the statistical properties of the noise [1,2]. H_∞ filters are more robust to disturbances [3] and modeling uncertainties than Kalman filters. The objective of an H_∞ filter design is to minimize the energy of the estimation error for the worst case bounded energy of the disturbance. Thus, in practical applications where disturbances may not be known exactly and system uncertainties may appear in modeling, the H_∞ technique is often used [4,5,6].

It is important to consider filter order. Standard Kalman and H_∞ approaches result in filters with the same or higher order than the system model. However, it is well-known that a reduced order filter design, i.e., the design of a filter whose order is lower than that of the system model, is important in many applications where fast data processing is necessary. The reduced order filter is often desirable because it reduces the filter complexity and real time computational burdens in many applications [7,8,9,10].

While a reduced order design decreases the number of computations required to implement the filter, it also results in a loss of performance and robustness [8]. In [11], the reduced order

filter problem was studied in an H_∞ setting, but the H_∞ problem was formulated as distance problem. A reduced order H_∞ filter for discrete-time linear time-varying systems was considered in [12].

Recently, the LMI (Linear Matrix Inequality) technique has received much attention. The importance of LMI in control theory was already recognized in the early 1960's [13]. Because a wide variety of problems arising in system and control theory can be reduced to an optimization problem involving LMIs and LMIs can be solved numerically very efficiently, LMIs have seen extensive investigation in the controls field. Using LMI techniques, many control problems can be solved that have no analytical solutions [14]. Thus, LMIs have emerged as a powerful computational design tools in system and control engineering because of their computational efficiency and flexibility in treating a large class of system design problems [14].

In [15], the robust reduced order filtering problem was studied in an H_2 setting via an LMI approach. In [9], the reduced order H_∞ filtering problem was studied via an LMI approach, but only for a specific linear time invariant plant model without model uncertainties.

In this work, the H_∞ approach for robust and reduced order filtering design via LMIs is used because it is known that the H_∞ approach is more robust to model uncertainties than H_2. Less conservative results can be achieved and a computational example is given to demonstrate this.

1.2 Outline

This dissertation is organized as follows: Chapter 2 contains a review of related literature. The definition of LMI, PLMI and useful tools are introduced. In particular, the lemmas and concepts that have an important role through the whole doctoral research are studied. In chapter 3, I

developed a new approach for designing robust and reduced order H_∞ filters using LMIs. Chapter 3 is divided into two main parts. Section 3.2 discusses the polytopic covering approach. The main result, which uses a practical approaching methodology in an H_∞ setting, is presented in Section 3.3. A numerical example is shown in Section 3.4.

In Chapter 4, I show how the results of Chapter 3 can be used for designing reduced order RFDFs. An introduction to fault detection filter design is described in Section 4.1. The problem is formulated in Section 4.2 using the concept of residual generation and the evaluation. The main result of this chapter, the reduced order RFDF design procedure, is developed in Section 4.3. Two examples are presented in Section 4.4. Chapter 5 contains a summary of this dissertation and a discussion of future work in this area.

CHAPTER 2

LITERATURE REVIEW

2.1 Linear Matrix Inequality (LMI)

The history of LMI in the analysis of dynamical systems goes back more than 100 years [13]. It begins in about 1890, when Lyapunov published his seminal work introducing what we now call Lyapunov theory. In the 1940's, the application of the Lyapunov's methods to real control engineering problems was studied and small LMIs were solved by hand. In the 1960's, the important role of LMIs in control theory was already recognized by Yakubovich. Graphical techniques were developed for solving a family of LMI problems. In the late 1960's, it was observed that the same family of LMIs can be solved by solving an ARE (Algebraic Riccati Equation). In the early 1980's, it was discovered that many LMIs could be solved by computer via convex programming. In the late 1980's, Nesterov and Nemirovski developed interior-point methods that apply directly to convex problems involving LMIs [16]. During the 1990's, LMI techniques have seen more investigation and have emerged as powerful design tools in areas ranging from control engineering to system identification and structural design.

Three factors that make LMI technique attractive are as follows [17]:

(i) A variety of design specifications and constraints can be expressed as LMIs.

(ii) Once formulated in terms of LMIs, a problem can be solved exactly by efficient convex optimization algorithms.

(iii) While most problems with multiple constraints or objectives lack analytical solutions in terms of matrix equations, they often remain tractable in the LMI framework. This

makes LMI-based design a valuable alternative to classical "analytical" methods.

These appealing LMI technique are based on the interior-point algorithm with the LMI providing a convex constraint. Linear inequalities, convex quadratic inequalities, matrix norm inequalities and various constraints from control theory such as Lyapunov and Riccati inequalities can all be written as LMIs.

2.1.1 Definition

A linear matrix inequality (LMI) is an expression of the form

$$F(x) = F_0 + x_1 F_1 + \cdots + x_S F_S = F_0 + \sum_{i=1}^{S} x_i F_i < 0 \qquad (2.1.1.1)$$

where the F_i's, $(i = 1,2,\cdots,s)$ are given real symmetric matrices and the x_i's are the sought scalar decision variables. The inequality (<0) means 'negative definite', i.e., $u^T F(x)u < 0$ for all $u \in R^n,\ u \neq 0$. Equivalently, the biggest eigenvalue of $F(x)$ is negative. This is the more general definition.

In most applications, LMIs do not naturally arise in the canonical form (2.1.1.1), but rather in the form

$$L(X_1,\cdots,X_S) < R(X_1,\cdots,X_S) \qquad (2.1.1.2)$$

where $L(\bullet)$ and $R(\bullet)$ are affine functions of some structured matrix variable $X_1,\cdots,X_S$. For example,

$$AX + XA^T < 0 \qquad (2.1.1.3)$$

is an LMI where X is a symmetric matrix decision variable.

An LMI defines a convex constraint on a decision variable. That is the set, $\Phi = \{x : F(x) < 0\}$, is convex. Indeed, if $x_1, x_2 \in \Phi$ and $\alpha \in (0,1)$ then

$$F(\alpha x_1 + (1-\alpha)x_2) = \alpha F(x_1) + (1-\alpha)F(x_2) < 0 \qquad (2.1.1.4)$$

where in the first equality we used the fact that F is affine. The last inequality follows from the fact that $\alpha \geq 0$ and $(1-\alpha) \geq 0$. This is an important property since powerful numerical solution techniques are available for the problems involving convex solution sets.

2.1.2 Usefulness of LMIs

One useful property of LMIs is the fact that a finite set of LMIs can be written as one single LMI. For example, $F_1(x) < 0, \cdots, F_S(x) < 0$ iff

$$F(x) = \begin{bmatrix} F_1(x) & 0 & \cdots & 0 \\ 0 & F_2(x) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & F_S(x) \end{bmatrix} < 0 \qquad (2.1.2.1)$$

The inequality in (2.1.2.1) makes sense as $F(x)$ is symmetric.

Another useful property of LMIs is the fact that it is easy to convert a nonlinear inequality to a linear inequality. If we partition matrix $M \in R^{n \times n}$ as

$$M = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \qquad (2.1.2.2)$$

where M_{11} has dimension $r \times r$. Assume that M_{11} is nonsingular. The matrix $S \equiv M_{22} - M_{21} M_{11}^{-1} M_{12}$ is called the Schur complement of M_{11} in M. If M is symmetric then we have

$$M < 0 \Leftrightarrow \begin{bmatrix} M_{11} & 0 \\ 0 & S \end{bmatrix} < 0 \qquad (2.1.2.3)$$

$$\Leftrightarrow \left\{ \begin{matrix} M_{11} < 0 \\ S < 0 \end{matrix} \right\} \qquad (2.1.2.4)$$

This result is obtained by observing that $M < 0$ iff $u^T M u < 0$ for all nonzero $u \in R^n$. Let

$F \in R^{r\times(n-r)}$. Then $M < 0$ iff for all $u_1 \in R^r$, $u_2 \in R^{n-r}$ and $F = -M_{11}^{-1}M_{12}$,

$$\begin{bmatrix} u_1 + Fu_2 \\ u_2 \end{bmatrix}^T \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \begin{bmatrix} u_1 + Fu_2 \\ u_2 \end{bmatrix}$$

$$= \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}^T \begin{bmatrix} M_{11} & M_{11}F + M_{12} \\ M_{21} + F^T M_{11} & M_{22} + F^T M_{11}F + F^T M_{12} + M_{21}F \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} < 0$$

$$= \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}^T \begin{bmatrix} M_{11} & 0 \\ 0 & S \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} < 0 \qquad (2.1.2.5)$$

Using this result, the nonlinear matrix inequality, (2.1.2.5), can be converted to an LMI (2.1.2.4).

2.1.3 Generic Problem

There are three generic problems related with LMIs.

(i) Feasibility problem: Finding a solution x to an LMI is called a feasibility problem.

(ii) Minimization problem (or eigenvalue problem): Minimizing a convex objective function under some LMI constraint. This linear objective minimization problem,

$$\min c^T x \text{, subject to } F(x) < 0 \qquad (2.1.3.1)$$

plays an important role in LMI based design.

(iii) Generalized eigenvalue problem: This amounts to minimizing a scalar $\lambda \in R$ subject to

$$\begin{Bmatrix} \lambda F(x) - G(x) < 0 \\ F(x) < 0 \end{Bmatrix} \qquad (2.1.3.2)$$

The algorithm used to solve these problems, the interior-point method, can be found in [16].

2.1.4 Convexity in LMI

Why is the convexity such a desirable property in LMIs? One reason is related to the

absence of local minima.

Definition 2.1.4.1

Let S be a subset of a normed space χ. The function $f: S \to R$ is said to have a local minimum at $x_0 \in S$ if there exist $\varepsilon > 0$ such that

$$f(x_0) \leq f(x) \tag{2.1.4.1}$$

for all $x \in S$ with $\|x - x_0\| < \varepsilon$. It is a global minimum of f if (2.1.4.1) holds for all $x \in S$. ∎

Proposition 2.1.4.2

Suppose that $f: S \to R$ is convex. If f has a local minimum at $x_0 \in S$, then $f(x_0)$ is also global minimum of f. If f is strictly convex, then x_0 is unique. ∎

This emphasizes that it is suffices to compute local minima of a convex function f to actually determine its global minimum. It leads to a straightforward algorithm for computation of optimal solutions [16].

2.2 Parametrized Linear Matrix Inequality (PLMI)

The LMI technique is well-known as a unifying framework for formulating and solving problems in control theory. The main advantage of this technique is that complicated control problems can be solved very efficiently with interior point methods [16]. A simple feasibility problem in semidefinite programming is to seek a solution to the LMI

$$F(x) = F_0 + x_1 F_1 + \cdots + x_S F_S = F_0 + \sum_{i=1}^{S} x_i F_i < 0 \tag{2.2.1}$$

where the F_i's, $(i = 1,2,\cdots,s)$ are given real symmetric matrices and the x_i's are the sought decision variables. A more complicated generalization of problem (2.2.1) is the feasibility problem

$$F_0(\alpha) + x_1(\alpha)F_1(\alpha) + \cdots + x_S(\alpha)F_S(\alpha) < 0 \tag{2.2.2}$$

where $\alpha \equiv [\alpha_1 \quad \cdots \quad \alpha_N]^T$ is an additional parameter allowed to take any value in a compact set Γ of R^N. The compact Γ is typically polytopic. In contrast with (2.2.1), the $F_i(\alpha)$, $(i = 1,2,\cdots,s)$ in (2.2.2) are now symmetric matrix valued functions of α, We are now seeking $x_i(\alpha)$ such that the LMI constraint (2.2.2) holds for any admissible value of α.

The complexity of (2.2.2) is due to the following :

(i) It is infinite-dimensional since the $x_i(\alpha)$ are sought in the infinite-dimensional space of the functions of α.

(ii) This is an infinitely constrained LMI problem for which each constraint corresponds to a given point in the range of α.

A common and practical approach for overcoming the difficulties arising from dimensionality is to select a finite basis of functions for the x_i's and reconsider the problem over the resulting spanned finite-dimensional space. In such cases, problem (2.2.2) simplifies to an LMI problem of the form

$$F_0(\alpha) + x_1F_1(\alpha) + \cdots x_SF_S(\alpha) < 0, \ \forall \alpha \in \Gamma \tag{2.2.3}$$

where $x_1, \cdots, x_S$ are conventional scalar decision variables as in (2.2.1) and Γ is compact set. Such problems are referred to as robust semidefinite programming problems [18]. In [18], robust semidefinite programming problems are defined as convex optimization problems where the data is not specified exactly but is known to belong to a given uncertainty set, and the constraints must hold for all possible values of the data in the given uncertainty set.

They are also called Parametrized Linear Matrix Inequality (PLMI) problems to stress the connection with the LMI control theory literature. PLMI problems have high complexity and are

even known to be NP-hard [18]. NP-hardness is strong evidence that problem is not polynomically solvable, i.e., the problem is difficult because its polynomial time algorithm is unlikely. NP-hardness is not a definite proof that no polynomial time algorithm exists, but it suggests that we stop searching for a polynomial time algorithm [19]. Thus, systematic relaxation techniques are needed to turn a PLMI problem into a standard LMI problem.

A fruitful technique for turning a PLMI problem to an LMI problem is the S-procedure method [13]. With this approach, scaling or multipliers are utilized to eliminate the LMI parameter dependency [20]. This results in additional conservatism in the resulting conditions and extra computational effort.

DCC (Directional Convexity Concept), d.c (difference of convex) convexification, separated convexification and Polytopic Covering approaches have also been used to relax the complex parameter dependency. The Polytopic Covering approach focused specially on polytopic systems. Unfortunately, these approaches result in enlarged LMIs. Thus, a more practical approach will be needed.

2.3 Useful Tools

2.3.1 Congruence Transformation

The matrix M is negative definite (positive definite, respectively) if and only if $T^T MT$ is negative definite (positive definite, respectively) for any nonsingular matrix T of appropriate dimension. The matrix $T^T MT$ is said to be congruent to M via the congruence transformation T.

2.3.2 Finsler's Lemma

Let $x \in R^n$, $Q \in S^n$ and $B \in R^{m \times n}$ such that the rank(B)$< n$. The following statements are equivalent.

(i) $x^T Q x < 0, \ \forall Bx = 0, \ x \neq 0$

(ii) $B^{\perp \mathrm{T}} Q B^{\perp} < 0$

(iii) $\exists \sigma \in R : Q - \sigma B^T B < 0$

(iv) $\exists \chi \in R^{n \times m} : Q + \chi B + B^T \chi^T < 0$

Proof : Refer to [21]. ∎

Finsler's Lemma has been previously used in control literature mainly with the purpose of eliminating design variables in matrix inequalities. Thus, Finsler's Lemma is often referred to as Elimination Lemma. Most applications move from statement (iv) to statement (ii), thus eliminating the variable (multiplier) χ. It is also called the Projection Lemma.

2.3.3 Linearly Parametrized Matrix Inequality over the Unit Simplex Γ

The parametrized inequality

$$\sum_{i=1}^{S} \alpha_i \varphi_i(P) < 0, \ \forall \alpha \in \Gamma \tag{2.3.3.1}$$

is feasible in the decision variable P if and only if the following system of matrix inequality is feasible in P :

$$\varphi_i(P) < 0, \ i = 1, \cdots, s \tag{2.3.3.2}$$

here $\varphi_i(P)$ are arbitrary matrix valued functions of P. This concept was already used in [22] to check the stability of linear systems with large parameter variations. The key result of [22] is that it needs to check the stability only for parameter values at the vertices of the polyhedron in parameter space. This concept can be paraphrased that each uncertain matrix

can be written as an unknown convex combination of s given extreme matrices, $\varphi_1(P),\cdots,\varphi_S(P)$, if and only if, it holds for some $\alpha_1 \geq 0,\cdots,\alpha_S \geq 0$ such that $\alpha_1 + \cdots + \alpha_S = 1$. This significantly reduces computational burden of the optimization since only a finite number of extreme models have to be considered [23].

2.3.4 BRL (Bounded Real Lemma)

Consider a dynamical system

$$\left.\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Dx(t) \end{aligned}\right\} \tag{2.3.4.1}$$

The H_∞ norm of transfer function of the system is less than γ ,where γ is given positive scalar, iff $\gamma^2 I - D^T D > 0$ and there exists a matrix $P = P^T > 0$ such that

$$(A^T P + PA + C^T C) + (PB + C^T D)(\gamma^2 I - D^T D)^{-1}(B^T P + D^T C) < 0 \tag{2.3.4.2}$$

The Schur complement lemma implies that ARI (Alegbraic Riccati Inequality) equation, (2.3.4.2), is equivalent to the existence of $P = P^T > 0$ such that the following LMI holds

$$\begin{bmatrix} A^T P + PA & PB & C^T \\ B^T P & -\gamma I & D^T \\ C & D & -\gamma I \end{bmatrix} < 0 \tag{2.3.4.3}$$

Proof: Refer to [13].

CHAPTER 3

ROBUST AND REDUCED ORDER H-INFINITY FILTERING VIA LMI APPROACH

3.1 Preliminaries

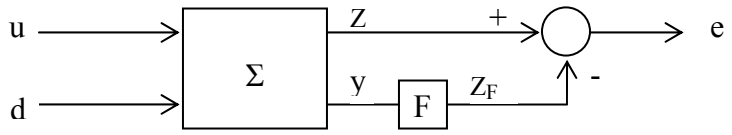


Figure 1. General filtering configuration

The general filtering configuration can be depicted as in Figure 1 where Σ is the plant, F is the filter that will be designed, d is an uncertain disturbance that includes process and measurement noise, z is the signal to be estimated, z_F is the estimate of z, e is the estimation error and y is the measured output.

A linear time invariant plant (Σ) described by:

$$\Sigma : \begin{Bmatrix} \dot{x}(t) = Ax(t) + Bd(t) \\ y(t) = Cx(t) + Dd(t) \\ z(t) = Lx(t) \end{Bmatrix} \tag{3.1.1}$$

will be considered where $x(t) \in R^n$ is the state, $y(t) \in R^p$ is the measured output, $z(t) \in R^q$ is the estimated output, $d(t) \in R^m$ is the disturbance and A, B, C, D and L are the plant matrices of appropriate dimensions. The control input is not included because this input does not affect the filter response when there are no modeling errors. To include plant model uncertainties, we formulate them in the form of a polytopic model as follows:

$$\begin{bmatrix} A & B \\ C & D \\ L & 0 \end{bmatrix} \in \left\{ \begin{bmatrix} A(\alpha) & B(\alpha) \\ C(\alpha) & D(\alpha) \\ L(\alpha) & 0 \end{bmatrix} = \sum_{i=1}^{s} \alpha_i \begin{bmatrix} A_i & B_i \\ C_i & D_i \\ L_i & 0 \end{bmatrix}, \alpha \in \Gamma \right\} \tag{3.1.2}$$

where Γ is a unit simplex such that

$$\Gamma \equiv \left\{ (\alpha_1, \cdots, \alpha_s) : \sum_{i=1}^{s} \alpha_i = 1, \alpha_i \geq 0 \right\} \tag{3.1.3}$$

This formulation is a convex combination, so it is suitable for the LMI approach. This convex bounded polytopic mathematical description of model uncertainty is sufficiently general to include many uncertain system with practical appeal. For instance, only some elements of the model may be known and several elements may depend on the same unknown parameter. These two situations can be easily accommodated by a proper choice of the set of extreme matrices.

The filter F is attached to the system as follows:

$$F : \begin{Bmatrix} \dot{x}_F(t) = A_F x_F(t) + B_F y(t) \\ z_F(t) = L_F x_F(t) \end{Bmatrix} \tag{3.1.4}$$

where $x_F(t) \in R^k$ is the filter state $(k \leq n)$ and $A_F \in R^{k \times k}, B_F \in R^{k \times p}$ and $L_F \in R^{q \times k}$ are the filter matrices that are to be synthesized.

Definition 3.1.1:

The L_2 norm of a vector valued function $f(t)$ is defined as:

$$\| f \|_{L_2} = \sqrt{\int_0^\infty f^T(t) f(t) dt} \tag{3.1.5}$$

The H_∞ optimal filtering problem is to find a filter F to minimize the worst case estimation error energy $\|e\|_{L_2}$ over all bounded energy disturbance d, where $e = z - z_F$, that is

$$\min_F \sup_{d \in L_2 - \{0\}} \frac{\|e\|_{L_2}}{\|d\|_{L_2}} \tag{3.1.6}$$

Hence, the optimal H_∞ filter minimizes the energy gain of the system from disturbance d to estimation error e.

Using the induced L_2-gain property of the H_∞ norm, this problem is equivalent to the following H_∞ norm minimization problem

$$\min_F \left\| T_{ed} \right\|_\infty \tag{3.1.7}$$

where T_{ed} is the transfer function from disturbance d to the estimation error e and the H_∞ norm is defined as the largest gain over all frequencies, i.e.,

$$\left\| T_{ed}(s) \right\|_\infty \equiv \sup_\omega \sigma_{\max}(T_{ed}(j\omega)) \tag{3.1.8}$$

where $\sigma_{\max}$ denotes maximum singular value of given function. The γ-suboptimal H_∞ filtering problem is to find a filter F such that

$$\left\| T_{ed} \right\|_\infty < \gamma \tag{3.1.9}$$

where γ is a given positive scalar. Therefore, the γ-suboptimal H_∞ filtering condition (3.1.9) guarantees that the filtering error energy will be bounded by $\gamma \left\| d \right\|_{L_2}$ for any disturbances d with bounded energy. Obviously, lower values of the H_∞ norm bound γ correspond to a smaller estimation error $\left\| e \right\|_{L_2}$.

To find the transfer function T_{ed}, (3.1.1) and (3.1.4) can be rewritten in augmented form as (3.1.10) and (3.1.11).

$$\dot{x}_{cl} = \begin{bmatrix} \dot{x} \\ \dot{x}_F \end{bmatrix} = \begin{bmatrix} Ax + Bd \\ A_F x_F + B_F y \end{bmatrix}$$

$$= \begin{bmatrix} Ax + Bd \\ A_F x_F + B_F (Cx + Dd) \end{bmatrix}$$

$$= \begin{bmatrix} Ax + Bd \\ A_F x_F + B_F Cx + B_F Dd \end{bmatrix}$$

$$= \begin{bmatrix} A & 0 \\ B_F C & A_F \end{bmatrix} \begin{bmatrix} x \\ x_F \end{bmatrix} + \begin{bmatrix} B \\ B_F D \end{bmatrix} d$$

$$= A_{cl} x_{cl} + B_{cl} d \tag{3.1.10}$$

$$e = z_{cl} = z - z_F = Lx - L_F x_F$$

$$= \begin{bmatrix} L & -L_F \end{bmatrix} \begin{bmatrix} x \\ x_F \end{bmatrix}$$

$$= L_{cl} x_{cl} \tag{3.1.11}$$

where

$$x_{cl} \equiv \begin{bmatrix} x \\ x_F \end{bmatrix},\ A_{cl} \equiv \begin{bmatrix} A & 0 \\ B_F C & A_F \end{bmatrix},\ B_{cl} \equiv \begin{bmatrix} B \\ B_F D \end{bmatrix},\ e \equiv z_{cl} = z - z_F \text{ and } L_{cl} \equiv \begin{bmatrix} L & -L_F \end{bmatrix} \tag{3.1.12}$$

The transfer function of closed system, T_{ed}, can be found as (3.1.13) and (3.1.14):

$$e \equiv z_{cl} = L_{cl} x_{cl}$$

$$= L_{cl} (sI - A_{cl})^{-1} B_{cl} d$$

$$= T_{ed} d \tag{3.1.13}$$

where

$$T_{ed} \equiv L_{cl} (sI - A_{cl})^{-1} B_{cl} \tag{3.1.14}$$

Using the well-known BRL ([24]), the condition in (3.1.9) with (3.1.14) can be described as:

$$\begin{bmatrix} PA_{cl} + A_{cl}^T P & PB_{cl} & L_{cl}^T \\ B_{cl}^T P & -\gamma I & 0 \\ L_{cl} & 0 & -\gamma I \end{bmatrix} < 0 \tag{3.1.15}$$

where A_{cl}, B_{cl} and L_{cl} are given in (3.1.12) and P is a symmetric positive definite matrix variable. Equation (3.1.15) satisfies the design requirement as follows [25]:

(i) internal stability: for d=0, the state vector x_{cl} of closed loop system tends to zero

as time goes to infinity.

(ii) performance: The H_∞ norm, $\|T_{ed}\|_\infty$, is less than specified positive scalar γ.

The statement (i) implies quadratic stability and (ii) implies quadratic performance. Remember that A_{cl}, B_{cl} and L_{cl} depend on the parameter α. Thus, statement (i) should be replaced by quadratic stability in the sense of parameter dependency.

Definition 3.1.2 ([26]):

The system in (3.1.10) and (3.1.11) is said to be parametrically quadratically stable if there exists a positive symmetric Lyapunov matrix P for $\forall \alpha \in \Gamma$ such that

$$A_{cl}^T(\alpha)P + PA_{cl}(\alpha) < 0 \qquad (3.1.16)$$

Statement (ii) is replaced by quadratic performance in the sense of parameter dependency using the following two lemmas, ***Lemma 3.1.3*** and ***Lemma 3.1.4.*** ***Lemma 3.1.3*** notes that the H_∞ norm upper bound, γ, for uncertain linear systems, can be derived based on the Lyapunov function argument. ***Lemma 3.1.4*** concerns the BRL in the case of parameter dependencies.

Lemma 3.1.3 ([13]):

For the system in (3.1.10) and (3.1.11), if there exists a quadratic function $V(x_{cl}) = x_{cl}^T P x_{cl}$, $P > 0$ and $\gamma \geq 0$, such that for all t and all admissible x_{cl} and d,

$$\frac{d}{dt}V(x_{cl}) + e^T e - \gamma^2 d^T d \leq 0 \qquad (3.1.17)$$

then the induced L_2 gain for this system is less than γ.

Lemma 3.1.4 ([26]):

If there exists $\gamma \geq 0$ and $P > 0$ such that

$$\begin{bmatrix} A_{cl}(\alpha)^T P + PA_{cl}(\alpha) & PB_{cl}(\alpha) & L_{cl}(\alpha)^T \\ B_{cl}(\alpha)^T P & -\gamma I & 0 \\ L_{cl}(\alpha) & 0 & -\gamma I \end{bmatrix} < 0, \forall \alpha \in \Gamma \qquad (3.1.18)$$

then, the induced L_2 gain from d to e is less than γ. ∎

Lemma 3.1.3 and ***Lemma 3.1.4*** are small extensions of the well-known BRL lemma for quadratic stability and performance of parameter dependent systems [27].

The validity of single Lyapunov matrix P for a parameter dependent system must be considered, even though it was used successfully for nonparametric systems. The notion of quadratic stability and performance, based on the search for a single quadratic Lyapunov function of the form $V(x) = x^T Px$ for a system, has encountered considerable academic success. However, it has long been known that such an approach often yields overly conservative results for parametric systems [28]. As a consequence, attention also has focused on using Lyapunov functions which explicitly depend on the parameters $\alpha_i, i = 1,2,\cdots,s$. The use of parametric Lyapunov functions can be traced back to the work of [29]. The well-known Popov criterion [30], when specialized to dealing with a single constant uncertain parameter rather than sector-bounded nonlinearity, yields a Lyapunov function that depends on the uncertain parameter affinely. In [28] and [31], a polytopic uncertain system was considered with the so-called generalized Popov criterion to generate parametric Lyapunov functions. It was also studied in [32] by using an affine Lyapunov matrix

$$P(\alpha) = \sum_{i=1}^{s} \alpha_i P_i \qquad (3.1.19)$$

where $P_i, (i = 1,2,\cdots,s)$ are symmetric. In [32], it was shown that finding an affine Lyapunov matrix could be turned into an LMI with variables $P_1,\cdots,P_s$. LMI problems are convex and

efficient polynomial time optimization algorithms are available to solve them [16,17]. Thus, the resulting robust stability test is therefore numerically tractable while always less conservative than quadratic stability. In [32], it was called AQS (affine quadratic stability) to emphasize the affine-dependency of the Lyapunov function on parameters. AQP (affine quadratic performance), a natural extension of the concept of BRL, was also defined. The following two definitions, ***Definition 3.1.5*** and ***Definition 3.1.6,*** were adapted to apply to the error dynamic system of (3.1.10) and (3.1.11).

Definition 3.1.5 ([32]):

The system of (3.1.10) and (3.1.11) is said to have AQS (affine quadratic stability) if there exist a positive symmetric affinely-parameter dependent Lyapunov matrix (3.1.19) such that

$$A_{cl}(\alpha)^T P(\alpha) + P(\alpha)A_{cl}(\alpha) < 0, \forall \alpha \in \Gamma \qquad (3.1.20)$$

Definition 3.1.6 ([32]):

The system of (3.1.10) and (3.1.11) is said to have AQP (affine quadratic H_∞ performance) if there exists a positive symmetric affinely-parameter dependent Lyapunov matrix (3.1.19) such that

$$\begin{bmatrix} A_{cl}(\alpha)^T P(\alpha) + P(\alpha)A_{cl}(\alpha) & P(\alpha)B_{cl}(\alpha) & L_{cl}(\alpha)^T \\ B_{cl}(\alpha)^T P(\alpha) & -\gamma I & 0 \\ L_{cl}(\alpha) & 0 & -\gamma I \end{bmatrix} < 0 \qquad (3.1.21)$$

that holds for all admissible parameter $\alpha_i (i = 1,2,\cdots,s)$.

3.2 Polytopic Covering Approach

In previous sections, we found that a parameter dependent Lyapunov function gave us less conservative results for stability and performance analysis than a single quadratic Lyapunov function. In this section, we will consider the filter synthesis problem equipped with the

previous section's knowledge. We will introduce a related definition.

Definition 3.2.1 ([33])

A convex covering for a bounded region is defined as the union of a set of convex sets, each spanned by a finite number of vertices, that contains the region. All such vertices are called corner points or extreme points of the convex covering. ∎

In this dissertation, we consider a polytopic systems, thus we call it a polytopic covering because a polytope is also convex. The convex covering of the parameter space is used to introduce a new set of parameters that enter affinely in the LMI. These LMI need to hold only at the extreme points of the convex covering. This concept was described at 2.3.3. Here, we try to derive a solvability condition for (3.1.15).

Theorem 3.2.2

Given the system in (3.1.10) and (3.1.11), there exists k_{th} order ($k < n$) filter F satisfying (3.1.15) iff there exist matrices $X(\alpha)$ and $Y(\alpha)$,

$$X(\alpha) = \sum_{i=1}^{s} \alpha_i x_i, \;\; Y(\alpha) = \sum_{i=1}^{s} \alpha_i Y_i \tag{3.2.1}$$

such that the following conditions are satisfied

$$\begin{bmatrix} X(\alpha)A(\alpha) + A(\alpha)^T X(\alpha) & X(\alpha)B(\alpha) \\ B(\alpha)^T X(\alpha) & -\gamma^2 I \end{bmatrix} < 0 \tag{3.2.2}$$

$$\begin{bmatrix} C(\alpha)^T \\ D(\alpha)^T \end{bmatrix}^{\perp} \begin{bmatrix} Y(\alpha)A(\alpha) + A(\alpha)^T Y(\alpha) + L(\alpha)^T L(\alpha) & Y(\alpha)B(\alpha) \\ B(\alpha)^T Y(\alpha) & -\gamma^2 I \end{bmatrix} \begin{bmatrix} C(\alpha)^T \\ D(\alpha)^T \end{bmatrix}^{\perp T} < 0 \tag{3.2.3}$$

$$\begin{bmatrix} Y(\alpha) & I \\ I & X(\alpha)^{-1} \end{bmatrix} \geq 0 \tag{3.2.4}$$

$$rank(X(\alpha) - Y(\alpha)) \leq k \tag{3.2.5}$$

Proof ([24]): The proof is quite standard now and is thus omitted. ∎

But, if we consider the character of our polytopic system, (3.1.10) and (3.1.11), (3.2.2) and (3.2.3) can be differently described as the following ***Corollary 3.2.3***.

Corollary 3.2.3

Let's introduce new parameter $\bar{\alpha}$ such as

$$\bar{\alpha} \equiv col\left\{\alpha_1, \cdots, \alpha_s, \bigcup_{i<j} \alpha_{ij}, {\alpha_1}^2, \cdots, {\alpha_S}^2\right\}, (i, j = 1, 2, \cdots, s) \tag{3.2.6}$$

then, (3.2.2) and (3.2.3) can be described as follows:

$$\begin{bmatrix} Q_1(X, \bar{\alpha}) & Q_2(X, \bar{\alpha}) \\ Q_2^T(X, \bar{\alpha}) & -\gamma^2 I \end{bmatrix} < 0 \tag{3.2.7}$$

$$\begin{bmatrix} R_1(Y, \bar{\alpha}) & R_2(Y, \bar{\alpha}) \\ R_2^T(Y, \bar{\alpha}) & -\gamma^2 I \end{bmatrix} - \sigma \begin{bmatrix} C(\alpha)^T \\ D(\alpha)^T \end{bmatrix} \begin{bmatrix} C(\alpha) & D(\alpha) \end{bmatrix} < 0 \tag{3.2.8}$$

where

$$Q_1(X, \bar{\alpha}) = \sum_i \left\{A_i(\bar{\alpha})^T X_i + X_i A_i(\bar{\alpha})\right\}, \tag{3.2.9}$$

$$Q_2(X, \bar{\alpha}) = \sum_i \left\{X_i B_i(\bar{\alpha})\right\}, \tag{3.2.10}$$

$$R_1(Y, \bar{\alpha}) = \sum_i \left\{Y_i A_i(\bar{\alpha}) + A_i(\bar{\alpha})^T Y_i + L_i(\bar{\alpha})^T L_i(\bar{\alpha})\right\}, \tag{3.2.11}$$

$$R_2(Y, \bar{\alpha}) = \sum_i \left\{Y_i B_i(\bar{\alpha})\right\} \tag{3.2.12}$$

Proof: Substitute (3.2.1) into (3.2.2), and express the results in the new parameter $\bar{\alpha}$. To get (3.2.8), Finsler's Lemma is used. ∎

The LMI obtained are then affine in both the decision variables $X_i, Y_i (i = 1, \cdots, s)$ and the parameter $\bar{\alpha}$.

Remark 3.2.4

The dimension of $\bar{\alpha}$ is $\frac{1}{2}s(s+3)$, so the increase in the number of parameter introduces an enlarged dimension of $\bar{\alpha}$. This can make the problem computationally intractable. ■

Remark 3.2.5

The H_∞ filter synthesis can be easily achieved using a very standard algorithm [24]. ■

Many other approaches were investigated such as DCC (directional convexity concept), d.c. (difference of convex) convexification, separated convexification [34], but these approaches resulted in computationally intractable enlarged LMIs. Furthermore, these approaches could not handle the nonconvex rank constraint (3.2.5). Thus, in Section 3.3 a more practical approach will be considered.

3.3 More Practical Approach

In this section, we will develop an H_∞ filtering approach in LMI framework that is applicable and easy to compute. Let's reconsider equation (3.1.15). The matrix inequality in (3.1.15) is nonlinear because the terms containing the P, A_{cl} and B_{cl} variables multiply the unknown variable P with the unknown that are included in A_{cl} and B_{cl}. Finsler's Lemma is standard tool to separate filter variables from the Lyapunov matrix P [15]. The Reciprocal Finsler's Lemma [35] can be used for H_2 filter design directly, but it cannot be used in the H_∞ case because it cannot separate the filter variables from the Lyapunov matrix variable. Consequently, an alternative LMI formulation that is very useful for solving the robust filtering problem will be used.

Theorem 3.3.1 ([36]):

The LMI

$$P>0,\begin{bmatrix} A^TP+PA & PB & C^T \\ B^TP & Q_{11} & Q_{12} \\ C & Q_{12}^T & Q_{22} \end{bmatrix}<0 \tag{3.3.1}$$

has a feasible decision variable P if and only if, for any choice of $\mu>0$, the following LMI is feasible in the decision variables V and P,

$$\begin{bmatrix} -(V+V)^T & V^TA+P & V^TB & 0 & V^T \\ A^TV+P & -\mu P & 0 & C^T & 0 \\ B^TV & 0 & Q_{11} & Q_{22} & 0 \\ 0 & C & Q_{12}^T & Q_{22} & 0 \\ V & 0 & 0 & 0 & -P/\mu \end{bmatrix}<0 \tag{3.3.2}$$

Proof:

(3.3.2)$\Rightarrow$(3.3.1):

Rewrite (3.3.2) as

$$\begin{bmatrix} 0 & P & 0 & 0 & 0 \\ P & -\mu P & 0 & 0 & 0 \\ 0 & 0 & Q_{11} & Q_{12} & 0 \\ 0 & C & Q_{12}^T & Q_{22} & 0 \\ 0 & 0 & 0 & 0 & -P/\mu \end{bmatrix}+\begin{bmatrix} -I \\ A^T \\ B^T \\ 0 \\ I \end{bmatrix}V\begin{bmatrix} I & 0 & 0 & 0 & 0 \end{bmatrix}+(*)<0 \tag{3.3.3}$$

In order to use Finsler's Lemma, we need to compute nullspaces

$$\mathrm{N}_{[-I\ \ A\ \ B\ \ 0\ \ I]}=\begin{bmatrix} A & B & 0 & I \\ I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix} \tag{3.3.4}$$

$$\mathrm{N}_{[I\ \ 0\ \ 0\ \ 0\ \ 0]} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix} \tag{3.3.5}$$

We apply Finsler's Lemma to (3.3.3) with (3.3.4) and (3.3.5).

$$\mathrm{N}^{T}_{[-I\ \ A\ \ B\ \ 0\ \ I]} \begin{bmatrix} 0 & P & 0 & 0 & 0 \\ P & -\mu P & 0 & C^{T} & 0 \\ 0 & 0 & Q_{11} & Q_{12} & 0 \\ 0 & C & Q_{12}^{T} & Q_{22} & 0 \\ 0 & 0 & 0 & 0 & -P/\mu \end{bmatrix} \mathrm{N}_{[-I\ \ A\ \ B\ \ 0\ \ I]}$$

$$= \begin{bmatrix} A^{T}P + PA - \mu P & PB & C^{T} & P \\ B^{T}P & Q_{11} & Q_{12} & 0 \\ C & Q_{12}^{T} & Q_{22} & 0 \\ P & 0 & 0 & -P/\mu \end{bmatrix} < 0 \tag{3.3.6}$$

and

$$\mathrm{N}^{T}_{[I\ \ 0\ \ 0\ \ 0\ \ 0]} \begin{bmatrix} 0 & P & 0 & 0 & 0 \\ P & -\mu P & 0 & C^{T} & 0 \\ 0 & 0 & Q_{11} & Q_{12} & 0 \\ 0 & C & Q_{12}^{T} & Q_{22} & 0 \\ 0 & 0 & 0 & 0 & -P/\mu \end{bmatrix} \mathrm{N}_{[I\ \ 0\ \ 0\ \ 0\ \ 0]}$$

$$= \begin{bmatrix} -\mu P & 0 & C^{T} & 0 \\ 0 & Q_{11} & Q_{12} & 0 \\ C & Q_{12}^{T} & Q_{22} & 0 \\ 0 & 0 & 0 & -P/\mu \end{bmatrix} < 0 \tag{3.3.7}$$

Now, (3.3.6) is equivalent to (3.3.1) by Schur complement, so this completes sufficiency.

(3.3.1)$\Rightarrow$(3.3.2):

Clearly, for P satisfying (3.3.1), there is a $\mu > 0$ such that (3.3.7) holds. By Finsler's Lemma,

this together with (3.3.6) is sufficient for the existence of V satisfying (3.3.2). This completes necessity part. Thus, it completes the proof. ∎

From ***Theorem 3.3.1***, (3.1.21) can be written as

$$\begin{bmatrix} -(V+V^T) & V^T A_{cl}+P & V^T B_{cl} & 0 & V^T \\ A_{cl}^T V+P & -\mu P & 0 & L_{cl}^T & 0 \\ B_{cl}^T V & 0 & -\gamma I & 0 & 0 \\ 0 & L_{cl} & 0 & -\gamma I & 0 \\ V & 0 & 0 & 0 & -P/\mu \end{bmatrix} < 0 \tag{3.3.8}$$

The parameter α is omitted. The slack variable V has been introduced to separate the variable P from the filter design variables.

To proceed further, we'll need to partition the V and P variables as

$$V = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix} \tag{3.3.9}$$

$$P = \begin{bmatrix} P_1 & P_3^T \\ P_3 & P_2 \end{bmatrix} \tag{3.3.10}$$

Because the reduced order case is being considered, the filter order k, will be less than or equal to the plant order n. The partitioned submatrices in (3.3.9) will be dimensionalized as $V_{11}(n\times n)$, $V_{12}(n\times k)$, $V_{21}(k\times n)$, and $V_{22}(k\times k)$. The matrix inequality (3.3.8) implies that V is nonsingular, and by invoking a small perturbation if necessary, it can be assumed that V_{22} is nonsingular as well [37].

Now, we need to enforce some special structure on V_{21} as

$$V_{21} = \begin{bmatrix} \tilde{V}_{21} & 0_{k\times(n-k)} \end{bmatrix} \tag{3.3.11}$$

where $\tilde{V}_{21}$ is a $k\times k$ matrix. The motivation for this will become clear as we proceed. We can replace $V, P, A_{cl}, B_{cl}, L_{cl}$ in (3.3.8) with (3.1.12), (3.3.9), (3.3.10) and (3.3.11). After that, we

perform a congruence transformation with the transformation matrix

$$diag\begin{bmatrix} I & V_{22}^{-1}\tilde{V}_{21} & I & V_{22}^{-1}\tilde{V}_{21} & I & I & I & V_{22}^{-1}\tilde{V}_{21} \end{bmatrix} \tag{3.3.12}$$

This yields the following LMI (3.3.13)

$$\begin{bmatrix} -(V_{11}+V_{11}^T) & -(S_2^T+\tilde{S}_1^T) & V_{11}^TA+\tilde{B}_FC+\hat{P}_1 & \tilde{A}_F+\hat{P}_3^T & V_{11}^TB+\tilde{B}_FD & 0 & V_{11}^T & \tilde{S}_1^T \\ * & -(S_1+S_1^T) & S_2A+\hat{B}_FC+\hat{P}_3 & \hat{A}_F+\hat{P}_2 & S_2B+\hat{B}_FD & 0 & S_2 & S_1^T \\ * & * & -\mu\hat{P}_1 & -\mu\hat{P}_3^T & 0 & L^T & 0 & 0 \\ * & * & * & -\mu\hat{P}_2 & 0 & -\hat{L}_F^T & 0 & 0 \\ * & * & * & * & -\gamma I & 0 & 0 & 0 \\ * & * & * & * & * & -\gamma I & 0 & 0 \\ * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_1 & -\frac{1}{\mu}\hat{P}_3^T \\ * & * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_2 \end{bmatrix} < 0 \tag{3.3.13}$$

where

$$S_1 = \tilde{V}_{21}^T V_{22}^{-T}\tilde{V}_{21} \tag{3.3.14}$$

$$\tilde{S}_1 = \begin{bmatrix} S_1 & 0_{k\times(n-k)} \end{bmatrix} \tag{3.3.15}$$

$$S_2 = \tilde{V}_{21}^T V_{22}^{-T} V_{12}^T \tag{3.3.16}$$

$$\hat{B}_F = \tilde{V}_{21}^T B_F \tag{3.3.17}$$

$$\tilde{B}_F = \begin{bmatrix} \hat{B}_F \\ 0_{(n-k)\times p} \end{bmatrix} \tag{3.3.18}$$

$$\hat{P}_1 = P_1 \tag{3.3.19}$$

$$\hat{P}_2 = \tilde{V}_{21}^T V_{22}^{-T} P_2 V_{22}^{-1}\tilde{V}_{21} \tag{3.3.20}$$

$$\hat{P}_3^T = P_3^T V_{22}^{-1}\tilde{V}_{21} \tag{3.3.21}$$

$$\hat{A}_F = \tilde{V}_{21}^T A_F V_{22}^{-1}\tilde{V}_{21} \tag{3.3.22}$$

$$\widetilde{A}_F = \begin{bmatrix} \hat{A}_F \\ 0_{(n-k)\times k} \end{bmatrix} \tag{3.3.23}$$

$$\hat{L}_F = L_F V_{22}^{-1} \widetilde{V}_{21} \tag{3.3.24}$$

$$\hat{P} = \begin{bmatrix} \hat{P}_1 & \hat{P}_3^T \\ \hat{P}_3 & \hat{P}_2 \end{bmatrix} \tag{3.3.25}$$

Remark 3.3.2:

The H_∞ filtering solvability condition (3.1.15) is reformulated as feasibility problem of (3.3.13) with respect to $\hat{A}_F, \hat{B}_F, \hat{L}_F, S_1, S_2, V_{11}, \hat{P}, \gamma$ where $\hat{P} > 0$. ■

Remark 3.3.3:

The filter matrices A_F, B_F and L_F can be derived by means of the following procedure.

(i) Compute $V_{22}, \widetilde{V}_{21}$ by solving the factorization problem

$$S_1 = \widetilde{V}_{21}^T V_{22}^{-T} \widetilde{V}_{21} \tag{3.3.26}$$

(ii) Compute A_F, B_F and L_F from

$$A_F = \widetilde{V}_{21}^{-T} \hat{A}_F \widetilde{V}_{21}^{-1} V_{22} \tag{3.3.27}$$

$$B_F = \widetilde{V}_{21}^{-T} \hat{B}_F \tag{3.3.28}$$

$$L_F = \hat{L}_F \widetilde{V}_{21}^{-1} V_{22} \tag{3.3.29}$$ ■

Remark 3.3.4:

The partition in (3.3.11) was necessary to use (3.3.27) for finding A_F. ■

Remark 3.3.5:

A key characteristic of this approach is that the intermediate variables $(\hat{A}_F, \hat{B}_F, \hat{L}_F)$ are independent of the parameter-dependent system data, so that the same approach is valid for systems depending on uncertain parameters. ■

Now we need to remember that we had a polytopic uncertain system. As explained in [15], the parameter dependent Lyapunov matrix $P(\alpha)$, such as

$$P(\alpha) = \sum_{i=1}^{s} \alpha_i P_i(\alpha), \alpha \in \Gamma \tag{3.3.30}$$

is symmetric positive definite for all admissible values of α if and only if this holds for all $P_i's$. Therefore, we need to check the solvability, (3.3.13), only at the vertices, $i = 1,2,\cdots,s$. This gives us the final form (3.3.31)

$$\begin{bmatrix}
-(V_{11}+V_{11}^T) & -(S_2^T+\widetilde{S}_1^T) & V_{11}^T A_i + \widetilde{B}_F C_i + \hat{P}_{1,i} & \widetilde{A}_F + \hat{P}_{3,i}^T & V_{11}^T B_i + \widetilde{B}_F D_i & 0 & V_{11}^T & \widetilde{S}_1^T \\
* & -(S_1+S_1^T) & S_2 A_i + \hat{B}_F C_i + \hat{P}_{3,i} & \hat{A}_F + \hat{P}_{2,i} & S_2 B_i + \hat{B}_F D_i & 0 & S_2 & S_1^T \\
* & * & -\mu \hat{P}_{1,i} & -\mu \hat{P}_{3,i}^T & 0 & L_i^T & 0 & 0 \\
* & * & * & -\mu \hat{P}_{2,i} & 0 & -\hat{L}_F^T & 0 & 0 \\
* & * & * & * & -\gamma I & 0 & 0 & 0 \\
* & * & * & * & * & -\gamma I & 0 & 0 \\
* & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_{1,i} & -\frac{1}{\mu}\hat{P}_{3,i}^T \\
* & * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_{2,i}
\end{bmatrix} < 0$$

$(i = 1,2,\cdots,s)$

(3.3.31)

Consequently, the minimum upper bound γ for the reduced order H_∞ filter can be found by the LMI optimization problem

$$\min_{V_{11},S_1,S_2,\hat{A}_F,\hat{B}_F,\hat{L}_F,\gamma,\hat{P}_i} \{\gamma : (3.3.31)\} \tag{3.3.32}$$

In summary, the γ-suboptimal H_∞ reduced order filter for polytopic uncertain system can be solved if and only if (3.3.31) holds for all vertices, $i = 1,2,\cdots,s$ and the minimum value can be found from (3.3.32). Also, the filter matrices (A_F, B_F, L_F) defining the k_{th} order filter is obtained from (i) and (ii) in ***Remark 3.3.3.***

3.4 Example

In this section, examples are provided that compare the result of this study with the results of earlier research. We handle two cases, the full order and the reduced order filter design, and demonstrate the advantages of this study. We borrowed the example from [15, (79)-(83)]. The example has the plant data

$$\begin{aligned} \dot{x} &= \begin{bmatrix} 0 & -1+0.3\alpha \\ 1 & -0.5 \end{bmatrix} x + \begin{bmatrix} -2 & 0 \\ 1 & 0 \end{bmatrix} w \\ y &= \begin{bmatrix} -100+10\beta & 100 \end{bmatrix} x + \begin{bmatrix} 0 & 1 \end{bmatrix} w \\ z &= \begin{bmatrix} 1 & 0 \end{bmatrix} x \end{aligned} \tag{3.4.1}$$

where the uncertainty parameters, α and β, have four types of uncertainty sets given by

$$|\alpha| \le 1, |\beta| \le 1 \tag{3.4.2}$$

$$|\alpha| \le 1, \beta = \alpha \tag{3.4.3}$$

$$|\alpha| \le 3, |\beta| \le 1 \tag{3.4.4}$$

$$|\alpha| \le 3, \beta = \alpha \tag{3.4.5}$$

Table 1 compares the results for [10,38,39] and (3.3.32) for the four types of uncertainties in (3.4.2)-(3.4.5). In (3.4.2), there are four extreme uncertain matrices for $(\alpha = \beta = 1)$, $(\alpha = \beta = -1)$, $(\alpha = 1, \beta = -1)$ and $(\alpha = -1, \beta = 1)$. For the conservative case, a single Lyapunov matrix P is used and for the nonconservative case, a different Lyapunov matrix P_i is used for each of the four vertices.

The calculations were done with [17]. An arbitrary positive scalar for μ should be chosen and finally, the feasibility of the formulation should be checked.

TABLE 1

PERFORMANCE COMPARISON ACCORDING TO EACH UNCERTAINTY TYPE

Type	Filter order	[38]: $\|T_{ed}\|_2$	[39]: $\|T_{ed}\|_2$	[15]: $\|T_{ed}\|_2$		(3.3.32): $\|T_{ed}\|_\infty$	
				Non conservative	conservative	Non conservative	conservative
$\|\alpha\| \le 1, \|\beta\| \le 1$	full	5.728	4.867	2.382	5.7495	2.965	3.346
	reduced		4.946	3.001	5.8467	3.179	4.304
$\|\alpha\| \le 1, \beta = \alpha$	full	4.819	4.373	2.382	4.8841	2.948	3.165
	reduced		4.556	3.079	5.0704	3.163	4.023
$\|\alpha\| \le 3, \|\beta\| \le 1$	full	$+\infty$	$+\infty$	93.365	$+\infty$	29.106	$+\infty$
	reduced			106.493	$+\infty$	29.679	$+\infty$
$\|\alpha\| \le 3, \beta = \alpha$	full	$+\infty$	$+\infty$	100.963	$+\infty$	28.808	$+\infty$
	reduced			106.517	$+\infty$	29.732	$+\infty$

If it is feasible, the minimum value of (3.3.32) can be found. The feasibility must be checked because the product term of μ and Lyapunov variable P is nonlinear. We need to tune μ until the LMI solver returns a feasible solution [40]. After finding feasible μ, we should find minimum γ according to changing μ. For any choice of $\mu > 0$, if solvable, it gives a unique solution because of the convexity. In our result for type (3.4.2), we have imposed $\mu = 7$ to get $\gamma = 2.965$ for the full order case and $\mu = 1.451$ to get $\gamma = 3.179$ for the reduced order case.

From Table 1, we can see that the optimization in (3.3.32) using parameter-dependent Lyapunov functions does not fail in the uncertainty cases of (3.4.4) and (3.4.5), but the conservative approaches in [38,39] failed for the same cases. In the uncertainty cases of (3.4.4) and (3.4.5) cases, we also found that the conservative application of [15] and our approach in (3.3.32), i.e., the usage of a single parameter-independent Lyapunov function, also all failed.

The filter data of the full and reduced order case in (3.4.2) could be found from (3.3.27)-(3.3.29). The Schur Decomposition method was used to solve factorization problem in (3.4.2). The result is shown in Table 2 for the uncertainty case in (3.4.2).

TABLE 2

FILTER SYNTHESIS

Type	Filter order	A_F	B_F	L_F
$\lvert\alpha\rvert \le 1, \lvert\beta\rvert \le 1$	full	$\begin{bmatrix} -23.5163 & 0.6242 \\ 6.3142 & -3.0403 \end{bmatrix}$	$\begin{bmatrix} 0.0006 \\ -0.0031 \end{bmatrix}$	$\begin{bmatrix} 588.5549 & 6.1988 \end{bmatrix}$
	reduced	-0.8396	0.0022	4.0742

As we already know, our H_∞ approach method, (3.3.32), is more robust to the uncertainties than the H_2 approach in [15]. To show this fact in an analytical sense, we found the filter data of [15] and (3.3.32) and plugged in (3.1.21) and compared $\|T_{ed}\|_\infty$ for the case of the uncertainty of in (3.4.2). Table 3 shows the comparison result of the performance. The Matlab code for full order is in Appendix A and for reduced order in Appendix B. $\|[15]\|_\infty$ means that the filter data was found using H_2 technique and then, the H_∞ norm was found.

TABLE 3

PERFORMANCE COMPARISON BETWEEN [15] AND (3.3.32)

Type	Filter order	$\|[15]\|_\infty$	(3.3.32)
$\lvert\alpha\rvert \le 3, \lvert\beta\rvert \le 1$	Full	11.1939	9.8752
	Reduced	11.3486	10.7081

Therefore, from this example we find that our approach is more robust to model uncertainty than the former approaches, [15,38,39] and gives a non-conservative result.

CHAPTER 4

ROBUST AND REDUCED ORDER FAULT DETECTION FILTER DESIGN VIA LMI APPROACH

4.1 Introduction

Now, we consider an application of robust reduced order filter design to fault detection (FD). As science and technology develops, the reliability and security of complex systems becomes more important. But, in many cases failures are inevitable. Failures and malfunctions in a system can result in unacceptable loss or hazards to man or machinery. Thus, on-line monitoring of faults as they occur during operation of a dynamic system is necessary. Hardware redundancy is often utilized to achieve fault tolerance. However, it requires extra cost and space.

As a consequence, functionally redundant fault detection schemes were developed [41]. These schemes are usually based on state estimation, parameter estimation and adaptive filtering. In this study, estimator based fault detection methods will be the focus. The key to estimator based fault detection is to generate a fault indicating signal (residual) using input and output signals from the monitored system [42][43]. A residual is defined as the signal generated from a consistency check of the variables. Ideally the residual should be zero when the system is normal and should diverge from zero when a fault occurs in the system. Faults are detected by setting a fixed or variable threshold on a residual quantity generated from the difference between real measurements and the estimates of these measurements found using the mathematical model.

However, there is always a model-reality mismatch between plant dynamics and the model

used for the residual generation [44][45][46]. This problem is further complicated by the fact that the true characteristics (e.g., spectral shape, variance, mean, stationarity, etc) of the disturbance and noise signals acting upon the system cannot be known exactly. These modelling uncertainties and disturbance make estimator based fault detecting schemes produce false alarms even when there are no faults [47]. Consequently, the residual generator must be made robust to the modelling uncertainties and disturbances. Thus, the robustness problem in FD is defined as the maximization of the detectability of faults and the minimization of the effect of uncertainties and disturbance on residuals [48].

Most of the studies about the robustness problem in FD have focused on the problem of creating robustness in residual generation [49][50][51]. The robustness of residual depends on its fault sensitivity. The residual should be sensitive to faults but insensitive to the modeling uncertainties and disturbance [52][53]. But, due to inevitable parameter uncertainties and disturbance encountered in a practical application, we will rarely find a situation where the conditions for a perfectly robust residual generation are met. It is therefore necessary to provide sufficient robustness not only in the residual generation stage but also in the decision-making stage [54].

In the decision-making stage, the generated residuals are examined for the likelihood of faults, and a decision rule is then applied to determine if any faults have occurred. The goal of robust decision-making is to minimize the false and missing alarm rates due to the effects that modelling uncertainties and unknown disturbance will have on the residuals. This can be achieved in several ways [54][55]. When a fixed threshold is used, the sensitivity to faults will be intolerably reduced if the threshold is chosen too high, whereas the false alarm rate will be too large when the threshold is chosen too low. The proper choice of the threshold is a delicate

problem. Thus, an adaptive threshold scheme is usually used for the threshold selection [56] [57][58]. In the adaptive threshold approach, the residual thresholds are varied according to the control activity of the process. To enhance the robustness of the adaptive threshold scheme, the proper determination of the functional form of the adaptive threshold law is important. While the decision-making stage often includes the function of fault isolation in the case of multiple fault occurrences, fault isolation is not considered in this study. From the above description, the general structure of robust residual generation and decision-making of model-based fault diagnosis can be depicted in Figure 2 [54].

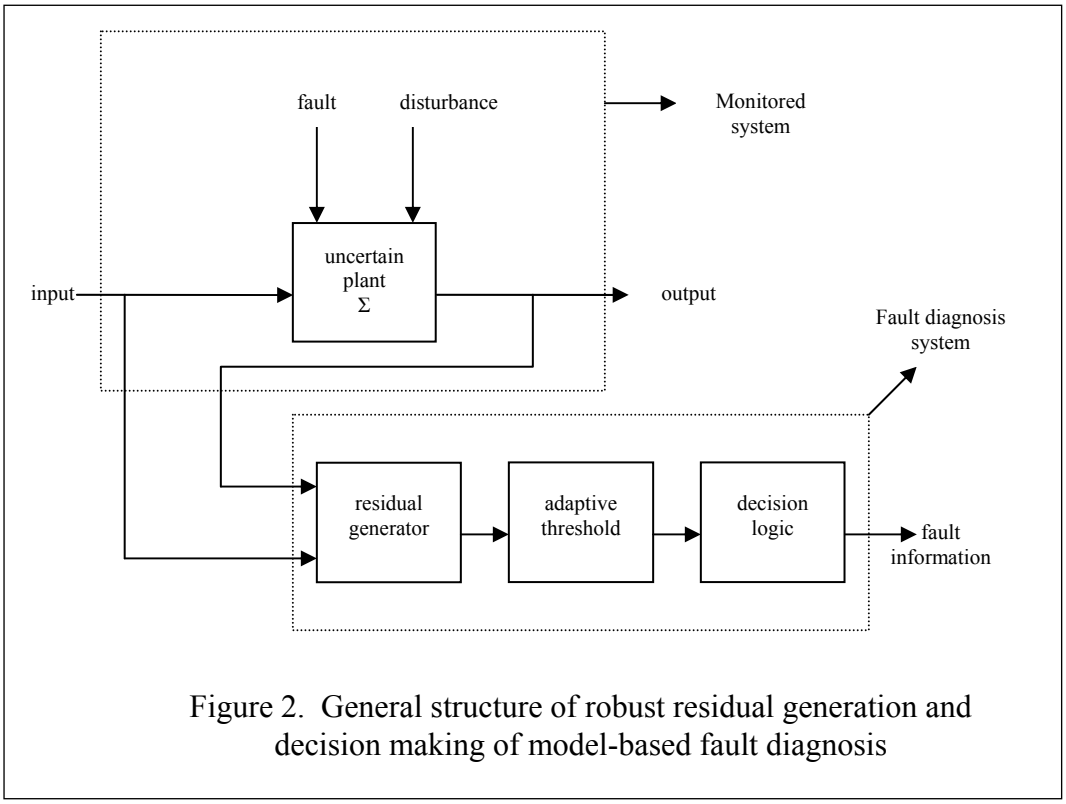


Figure 2. General structure of robust residual generation and decision making of model-based fault diagnosis

4.2 Problem Formulation

4.2.1 Residual Generation

Let's consider the following uncertain continuous time linear system described by

$$\Sigma : \begin{Bmatrix} \dot{x}(t) = Ax(t) + B_u u(t) + B_f f(t) + B_d d(t) \\ y(t) = Cx(t) + D_u u(t) + D_f f(t) + D_d d(t) \end{Bmatrix} \tag{4.2.1.1}$$

where $x(t) \in R^n$ is the state, $y(t) \in R^m$ is the measured output, $f(t) \in R^{n_f}$ is the fault, $d(t) \in R^{n_d}$ is the bounded disturbance and $u(t) \in R^{n_u}$ is the control signal. Actuator and component faults are modeled by $B_f f(t)$ and sensor faults are modeled by $D_f f(t)$. Plant model uncertainties are modeled in the form of a polytopic model as follows:

$$\begin{bmatrix} A & B_u & B_f & B_d \\ C & D_u & D_f & D_d \end{bmatrix} \in \left\{ \begin{bmatrix} A(\alpha) & B_u(\alpha) & B_f(\alpha) & B_d(\alpha) \\ C(\alpha) & D_u(\alpha) & D_f(\alpha) & D_d(\alpha) \end{bmatrix} = \sum_{i=1}^{s} \alpha_i \begin{bmatrix} A_i & B_{ui} & B_{fi} & B_{di} \\ C_i & D_{ui} & D_{fi} & D_{di} \end{bmatrix}, \alpha \in \Gamma \right\} \tag{4.2.1.2}$$

where Γ is a unit simplex such that

$$\Gamma \equiv \left\{ (\alpha_1, \cdots, \alpha_s) : \sum_{i=1}^{s} \alpha_i = 1, \alpha_i \geq 0 \right\} \tag{4.2.1.3}$$

Because this formulation is a convex combination, it is suitable for a LMI approach.

Here we assume that the above polytopic system possesses the affine quadratic stability that was introduced in ***Definition 3.1.5***. Other assumptions that are made for our purpose are that (C, A) is detectable and $\begin{bmatrix} A - j\omega I & B_d \\ C & D_d \end{bmatrix}$ has full rank for all ω. The assumption that (C, A) is detectable is standard. The assumption that $\begin{bmatrix} A - j\omega I & B_d \\ C & D_d \end{bmatrix}$ has full row rank for all ω ensures that $G_{yd} = \begin{bmatrix} A & B_d \\ C & D_d \end{bmatrix}$ has no zeros on the $j\omega$-axis. It is a standard assumption in robust

control theory.

Generally speaking, FD consists of two parts: a residual generation part and a residual evaluation part that includes a threshold and a decision logic unit. The residual generator part is considered first. The model based fault detection system that was introduced in the Section 4.1 uses information known a priori about the characteristics of certain signals and generates a fault indicating signal, a residual, using available input and output information from the monitored system. The residual should be normally zero or close to zero when no fault is present, but distinguishably different from zero when a fault occurs. The algorithm (or processor) used to generate residuals is called a residual generator. Thus, residual generation is a procedure for extracting fault symptoms from the system. The residual should ideally carry only fault information, but in the real world it does not. Consequently, we should make the residual generator as robust to the unknown inputs (model uncertainty, noise, etc) as possible.

The structure of residual generator is depicted in Figure 3 [55].

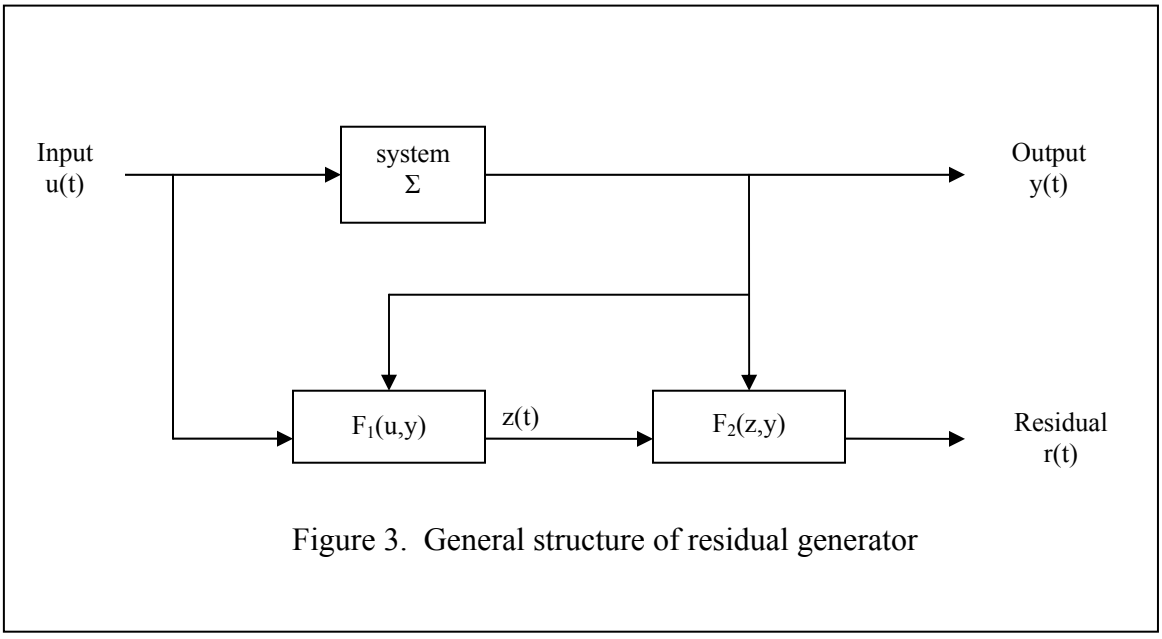


Figure 3. General structure of residual generator

The block $F_1(u,y)$ generates an auxiliary signal z(t) that generates the residual r(t) satisfying the

condition

$$r(t) = F_2(z(t), y(t)) = 0 \tag{4.2.1.4}$$

for fault free case. From this generalized residual generator, a number of residual generation methods can be developed, for example [54],

(i) Observer-based: full-order state observers, generalized observers, unknown input observers, failure detection filters, Kalman filter and modifications

(ii) Parity relation

(iii) Stable factorization of transfer matrices

(iv) Parameter estimation

(v) H_∞ optimization: multi-objective optimization, LMI approach, standard H_∞ filtering

The above methods constitute the main approaches to model based residual generation system. The most important characteristic of residual generator is how close it comes to depending solely and totally on faults. Order reduction in a residual generator is also important because it gives us faster data processing and a reduction in detection filter complexity. Because the generalized observer formulation makes order reduction easier to handle than the state observer formulation, the generalized observer is used in this study [54]. In the fault detection literature this observer is usually called a fault detection filter to emphasize the relationship with the filtering concept.

The proposed fault detection filter (FDF) will have the form

$$F : \begin{Bmatrix} \dot{x}_F(t) = A_F x_F(t) + B_F \begin{bmatrix} y(t) \\ u(t) \end{bmatrix} \\ z(t) = L_F x_F(t) + H_F \begin{bmatrix} y(t) \\ u(t) \end{bmatrix} \end{Bmatrix} \tag{4.2.1.5}$$

The residual $r(t)$ is defined as

$$r(t) \equiv z(t) - y(t) \tag{4.2.1.6}$$

Thus, the robust fault detection filter (RFDF) design problem can be described as designing the filter (A_F, B_F, L_F, H_F) such that the residual $r(t)$ is as sensitive as possible to the fault $f(t)$ and as robust as possible to unknown input $d(t)$, control input $u(t)$ and polytopic model uncertainty, $(A(\alpha), B_u(\alpha), B_f(\alpha), B_d(\alpha), C(\alpha), D_u(\alpha), D_f(\alpha), D_d(\alpha))$.

There are a number of schemes to approach the RFDF problem. In [57] it is formulated as an optimization problem. It is stated as

$$\min_F \frac{\|G_{rd}(s)\|}{\|G_{rf}(s)\|} \tag{4.2.1.7}$$

where different choices of $\|\bullet\|$ are possible and G_{rd} and G_{rf} are the transfer functions from the disturbances (d) and fault (f) to residual (r). Unfortunately, this approach is difficult to extend to systems with modeling uncertainty [53].

Another approach is to formulate the RFDF problem as an H_∞ model–matching problem [59]. It can be stated as

$$\min_F \frac{\|r - f\|}{\|d\|} \tag{4.2.1.8}$$

The idea of this approach is to use the residual as a fault estimator. However, this approach may produce a false alarm in a no fault situation [60]. Thus, we propose a new approach that uses multi-objective H_∞ optimization [61], [62] based on the model-matching formulation. Before proceeding to multi-objective H_∞ optimization, we need to provide more details about the optimization and model-matching problem.

As previously explained, the generated residual should be sensitive to faults, but insensitive to uncertainties and noises. To indicate the residual's sensitivity to faults, a

new H_- index was proposed by Hou and Patton [63]. The H_- index is defined as the minimum nonzero singular value of a transfer matrix. As we know, H_∞ is used to measure the robustness against unknown uncertainties and disturbances. The H_∞ norm is the worst case estimation on the impact of disturbances. If the H_∞ norm of the transfer function from faults to residual is used to check the worst case impact of fault, it cannot measure the best sense of fault occurrence. To express the best sensitivity to the fault, the H_- index was introduced. However, this is not a norm [55]. Thus, optimization problems with this index cannot be solved in a systematic way [64]. Therefore, the model-matching problem needs to be introduced to relate the minimum gain from fault to residual with the residual error [65].

The block diagram for the residual can be depicted as follows:

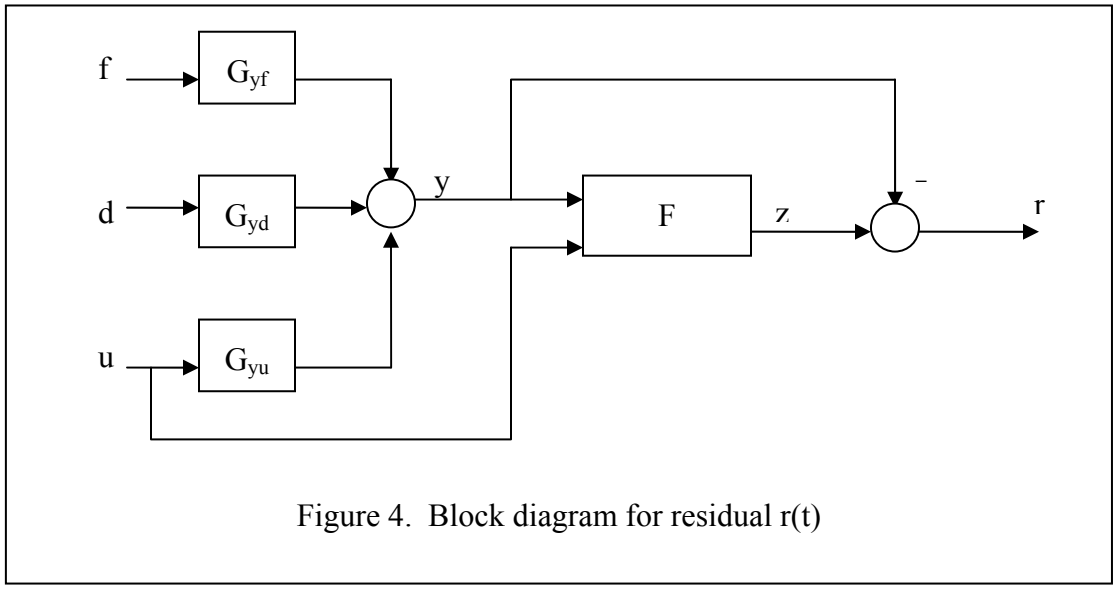


Figure 4. Block diagram for residual r(t)

where $G_{yf} = \begin{bmatrix} A & B_f \\ C & D_f \end{bmatrix}$, $G_{yd} = \begin{bmatrix} A & B_d \\ C & D_d \end{bmatrix}$ and $G_{yu} = \begin{bmatrix} A & B_u \\ C & D_u \end{bmatrix}$ are transfer function matrices defined from (4.2.1.1). The residual r is given by

$$r = z - y$$

$$= L_F x_F + H_F \begin{bmatrix} y \\ u \end{bmatrix} - \{Cx + D_u u + D_f f + D_d d\}$$

$$= L_F (sI - A_F)^{-1} B_F \begin{bmatrix} y \\ u \end{bmatrix} + H_F \begin{bmatrix} y \\ u \end{bmatrix} - \{C(sI - A)^{-1}[B_u u + B_f f + B_d d] + D_u u + D_f f + D_d d\}$$

$$= [L_F (sI - A_F)^{-1} B_F + H_F] \begin{bmatrix} y \\ u \end{bmatrix} - \{G_{yu} u + G_{yf} f + G_{yd} D\}$$

$$= F \begin{bmatrix} G_{yu} u + G_{yf} f + G_{yd} D \\ u \end{bmatrix} - \{G_{yu} u + G_{yf} f + G_{yd} D\}$$

$$= \left\{ F \begin{bmatrix} G_{yf} \\ 0 \end{bmatrix} - G_{yf} \right\} f + \left\{ F \begin{bmatrix} G_{yu} \\ I \end{bmatrix} - G_{yu} \right\} u + \left\{ F \begin{bmatrix} G_{yd} \\ 0 \end{bmatrix} - G_{yd} \right\} d$$

$$\equiv T_{rf} f + T_{ru} u + T_{rd} d \tag{4.2.1.9}$$

where the transfer matrices, T_{rf}, T_{ru} and T_{rd}, are defined as follows:

$$T_{rf} = F \begin{bmatrix} G_{yf} \\ 0 \end{bmatrix} - G_{yf}$$

$$T_{ru} = F \begin{bmatrix} G_{yu} \\ 0 \end{bmatrix} - G_{yu}$$

$$T_{rd} = F \begin{bmatrix} G_{yd} \\ 0 \end{bmatrix} - G_{yd} \tag{4.2.1.10}$$

For r to be sensitive to faults and insensitive to uncertainties (disturbance, modeling errors), $\|T_{rf}\|_-$ should be maximized and $\|T_{ru}\|_\infty$ and $\|T_{rd}\|_\infty$ should be minimized. As already mentioned,

$\|\bullet\|_-$ is not norm so its optimization is difficult. But, if we use the residual error instead of the residual, it will be more convenient for optimizing the detection filter F. The residual error is defined as

$$\tilde{r} = W(s) f - r$$

$$= W(s) f - \left\{ T_{rf}(s) f + T_{ru}(s) u + T_{rd}(s) d \right\}$$

$$= \begin{bmatrix} W(s) - T_{rf}(s) & \vdots & -T_{ru}(s) & \vdots & -T_{rd}(s) \end{bmatrix} \begin{bmatrix} f \\ u \\ d \end{bmatrix} \tag{4.2.1.11}$$

where $W(s)$ is called reference model [66]. The reference model, $W(s)$, is an RH_∞ transfer matrix [55]. The idea of a reference model has successfully been used to describe signal behavior in other fields like controller design [67] and adaptive control [68]. The block diagram for the residual error is depicted in Figure 5.

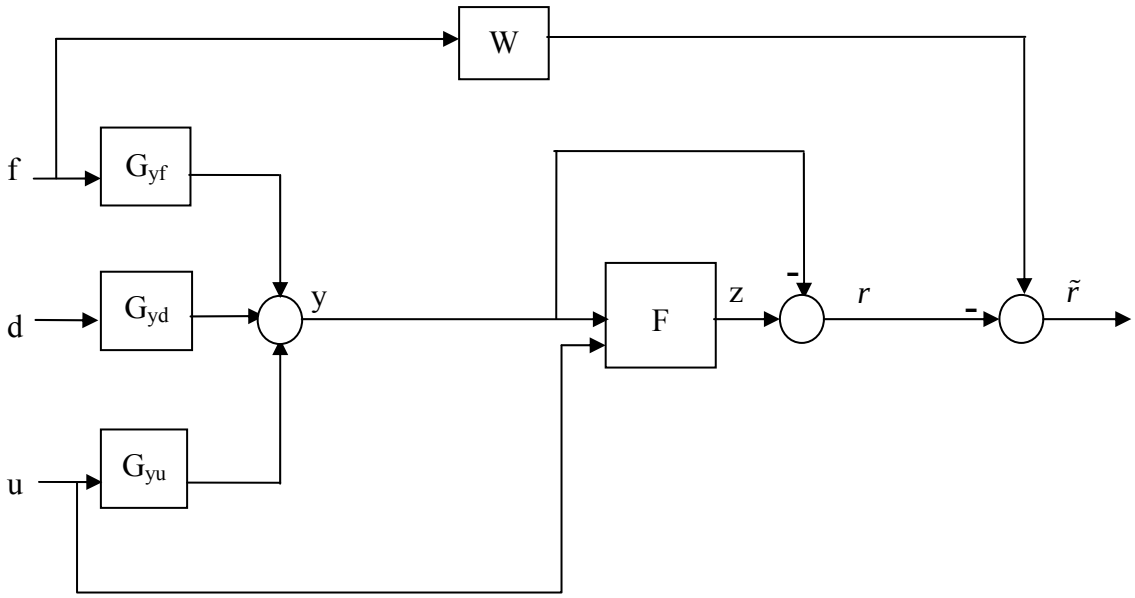


Figure 5. Block diagram for residual error $\tilde{r}$ with reference model

A poorly chosen reference model can result in a residual generator with poor robustness [66]. The suitable choice helps maximize the robustness, keeps the residual sensitive to faults and meets the specified performance. In [66], the residual generator is determined first, and then the reference model $W(s)$ is decided. However, in this study, the FDF cannot be found without the reference model because the optimization is based on (4.2.1.11), which includes the reference model. So, we use the method that was described in [61]. As discussed in [66], the main function of the reference model is to describe the desired behavior of the residual vector r with respect to the faults f. For example, if we desire to design a residual generator such that it responds to low frequency components ($\omega < \omega_l$) of the fault, $W(s)$ could be chosen as [66]

$$W(s) = \frac{\omega_l}{s + \omega_l} \tag{4.2.1.12}$$

If we want to detect faults in the frequency range between 0 and 2 rad/s with a -20dB/dec roll-off at the higher frequencies, the reference model, $W(s)$ is given as

$$W(s) = \frac{2}{s+2} \tag{4.2.1.13}$$

Using the approach in [61], $W(s)$ in (4.2.1.13) is placed between the plant and the filter so that the filter can track the faults with its specific feature. This can increase the robustness of the filter to the specific faults, if the designer can choose $W(s)$ suitably. Thus, the block diagram for the residual error, $\tilde{r}(t)$, with tracking filter $W(s)$ is shown in Figure 6.

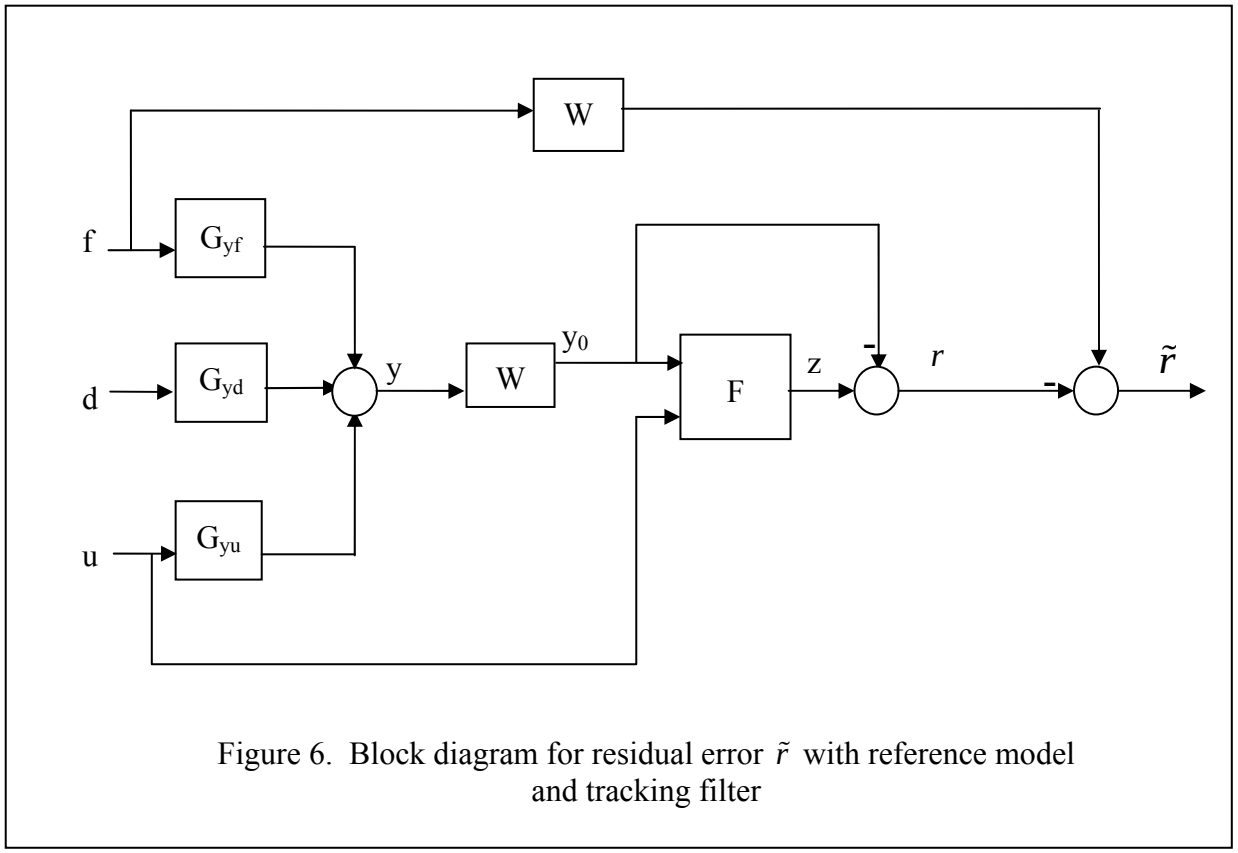


Figure 6. Block diagram for residual error $\tilde{r}$ with reference model and tracking filter

With this background, we can describe the RFDF design problem using multi-objective H_∞ optimization as follows:

Definition 4.2.1.1

Given positive scalars α_d, α_f and α_u, find $F(s)$ such that

$$\min_{F(s)}(\alpha_d * \gamma_d + \alpha_f * \gamma_f + \alpha_u * \gamma_u)$$

$$\max_{\alpha\in\Gamma}\left\| F(s)\begin{bmatrix} G_{y_0 d}(s) \\ 0 \end{bmatrix} - G_{y_0 d}(s)\right\|_\infty \equiv \max_{\alpha\in\Gamma}\left\|T_{rd}^0\right\| \le \gamma_d$$

$$\text{s.t.} \qquad \max_{\alpha\in\Gamma}\left\| W(s) - (F(s)\begin{bmatrix} G_{y_0 f}(s) \\ 0 \end{bmatrix} - G_{y_0 f}(s))\right\|_\infty \equiv \max_{\alpha\in\Gamma}\left\|W(s) - T_{rf}^0\right\| \le \gamma_f$$

$$\max_{\alpha\in\Gamma}\left\| F(s)\begin{bmatrix} G_{y_0 u}(s) \\ I \end{bmatrix} - G_{y_0 u}(s)\right\|_\infty \equiv \max_{\alpha\in\Gamma}\left\|T_{ru}^0\right\| \le \gamma_u \tag{4.2.1.14}$$

where $G_{y_0 f} \equiv WG_{yf}$, $G_{y_0 d} \equiv WG_{yd}$, $G_{y_0 u} \equiv WG_{yu}$, $T_{rf}^0 \equiv F\begin{bmatrix} G_{y_0 f} \\ 0 \end{bmatrix} - G_{y_0 f}$, $T_{rd}^0 \equiv F\begin{bmatrix} G_{y_0 d} \\ 0 \end{bmatrix} - G_{y_0 d}$ and $T_{ru}^0 \equiv F\begin{bmatrix} G_{y_0 u} \\ 0 \end{bmatrix} - G_{y_0 u}$. ■

The positive scalars $(\alpha_d, \alpha_f, \alpha_u)$ are used to weight the relative importance of tracking and filtering performance. Reduction in the order of the detection filter will be explored and its effectiveness will be studied. This will be discussed in the main results.

4.2.2. Residual Evaluation

Another important task for fault detection is the evaluation of the generated residual. Robust residual evaluation, which can keep the false alarm rate small with an acceptable sensitivity to faults, can be accomplished in many ways. Examples include statistical data processing, data reconciliation, correlation, pattern recognition, fuzzy logic or adaptive threshold [58]. An adaptive threshold will be used in this work. The disadvantage of a fixed threshold is that if

we fix the threshold too low, it can increase the rate of false alarms. Thus, the optimal choice of the magnitude of the threshold depends upon the nature of the system uncertainties and varies with the system input. That is called an adaptive threshold.

The first step of residual evaluation is to choose an evaluation function and to determine the corresponding threshold. Among a number of residual evaluation functions, the so-called time windowed root mean square (RMS) is often used. The time windowed RMS is represented as

$$\|r(t)\|_{RMS,T} = \sqrt{\frac{1}{T}\int_{t-T}^{t} r^T(\tau)r(\tau)d\tau} \tag{4.2.2.1}$$

where T is the finite time window. Since an evaluation of residual signal over the whole time range is impratical, the time windowed RMS evaluation method is used in practice to detect faults as early as possible. After selecting the evaluation function, we are able to determine the threshold.

A major requirement for fault detection is to reduce or prevent false alarms. Thus, in the absence of any faults, $\|r(t)\|_{RMS,T}$ should be less than the threshold value J_{th}, i.e.,

$$J_{th} = \sup_{\alpha\in\Gamma,\, f=0} \|r(t)\|_{RMS,T} \tag{4.2.2.2}$$

Under fault-free conditions, (4.2.1.14) can be described as

$$r(s) = T_{ru}^0(s)u(s) + T_{rd}^0(s)d(s) \tag{4.2.2.3}$$

From the Parseval Theorem [69] and the RMS norm relationship [70], the threshold J_{th} can be found as follows:

$$
\begin{aligned}
J_{th} &\equiv \|r\|_{RMS,T,f=0} \\
&= \left\|T_{ru}^{0}u + T_{rd}^{0}d\right\|_{RMS,T} \\
&\le \left\|T_{ru}^{0}\right\|_{\infty}\|u\|_{RMS,T} + \left\|T_{rd}^{0}\right\|_{\infty}\|d\|_{RMS,T} \\
&= \gamma_u\|u\|_{RMS,T} + \gamma_d\|d\|_{RMS,T}
\end{aligned}
\tag{4.2.2.4}
$$

where γ_u and γ_d come from the optimization problem, (4.2.1.14) and $\|d\|_{RMS,T}$ is the worst disturbance acting on the plant. Thus, in (4.2.2.4), $\|d\|_{RMS,T}$ is evaluated off-line while $u(t)$ is assumed to be known and $\|u\|_{RMS,T}$ is calculated on-line.

4.3 Main Results

In this section, LMIs will be used to solve the problem formulated in Section 4.2 for a RFDF. To solve the optimization problem in (4.2.1.14), we need the state-space realization of each transfer matrix. The reference model $W(s)$ can be realized as

$$
W(s) = \begin{bmatrix} A_w & B_w \\ C_w & D_w \end{bmatrix} \tag{4.3.1}
$$

where, in this formulation, the feedthrough term D_w is zero. After some manipulation, the transfer matrices term in (4.2.1.14) can be realized as

$$F(s)\begin{bmatrix} G_{y_0 d}(s) \\ 0 \end{bmatrix} - G_{y_0 d}(s)$$

$$= \left[\begin{array}{cccc:c} A & 0 & 0 & 0 & B_d \\ B_w C & A_w & 0 & 0 & B_w D_d \\ 0 & 0 & A_w & 0 & 0 \\ 0 & B_F \begin{bmatrix} C_w \\ 0 \end{bmatrix} & 0 & A_F & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & C_w - H_F \begin{bmatrix} C_w \\ 0 \end{bmatrix} & C_w & -L_F & -H_F \begin{bmatrix} 0 \\ I \end{bmatrix} \end{array}\right]$$

$$W(s) - (F(s)\begin{bmatrix} G_{y_0 f}(s) \\ 0 \end{bmatrix} - G_{y_0 f}(s))$$

$$= \left[\begin{array}{cccc:c} A & 0 & 0 & 0 & B_f \\ B_w C & A_w & 0 & 0 & B_w D_f \\ 0 & 0 & A_w & 0 & B_w \\ 0 & B_F \begin{bmatrix} C_w \\ 0 \end{bmatrix} & 0 & A_F & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & C_w - H_F \begin{bmatrix} C_w \\ 0 \end{bmatrix} & C_w & -L_F & -H_F \begin{bmatrix} 0 \\ I \end{bmatrix} \end{array}\right]$$

$$F(s)\begin{bmatrix} G_{y_0 u}(s) \\ I \end{bmatrix} - G_{y_0 u}(s)$$

$$= \left[\begin{array}{cccc:c} A & 0 & 0 & 0 & B_u \\ B_w C & A_w & 0 & 0 & B_w D_u \\ 0 & 0 & A_w & 0 & 0 \\ 0 & B_F \begin{bmatrix} C_w \\ 0 \end{bmatrix} & 0 & A_F & B_F \begin{bmatrix} 0 \\ I \end{bmatrix} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & C_w - H_F \begin{bmatrix} C_w \\ 0 \end{bmatrix} & C_w & -L_F & -H_F \begin{bmatrix} 0 \\ I \end{bmatrix} \end{array}\right] \tag{4.3.2}$$

Here, we note that the order of filter is

$$n_F = 2n_w + n \tag{4.3.3}$$

where n_F is the filter order, n_w is the tracking filter order and n is the plant order. To simplify (4.3.2), we let

$$\widetilde{A} \equiv \begin{bmatrix} A & 0 & 0 \\ B_w C & A_w & 0 \\ 0 & 0 & A_w \end{bmatrix}$$

$$\widetilde{C} \equiv \begin{bmatrix} 0 & \begin{bmatrix} C_w \\ 0 \end{bmatrix} & 0 \end{bmatrix}$$

$$\widetilde{B}_d \equiv \begin{bmatrix} B_d \\ B_w D_d \\ 0 \end{bmatrix}$$

$$\widetilde{B}_f \equiv \begin{bmatrix} B_f \\ B_w D_f \\ B_w \end{bmatrix}$$

$$\widetilde{B}_u \equiv \begin{bmatrix} B_u \\ B_w D_u \\ 0 \end{bmatrix}$$

$$\widetilde{L}_F \equiv \begin{bmatrix} 0 & C_w - H_F \begin{bmatrix} C_w \\ 0 \end{bmatrix} & C_w \end{bmatrix} \tag{4.3.4}$$

Using (4.3.4), (4.3.2) can be rewritten as

$$F(s)\begin{bmatrix} G_{y_0 d}(s) \\ 0 \end{bmatrix} - G_{y_0 d}(s)$$

$$= \left[\begin{array}{ccc|c} \widetilde{A} & 0 & \vdots & \widetilde{B}_d \\ B_F\widetilde{C} & A_F & \vdots & 0 \\ \cdots & \cdots & \vdots & \cdots \\ \widetilde{L}_F & -L_F & \vdots & -H_F\begin{bmatrix} 0 \\ I \end{bmatrix} \end{array}\right]$$

$$W(s) - (F(s)\begin{bmatrix} G_{y_0 f}(s) \\ 0 \end{bmatrix} - G_{y_0 f}(s))$$

$$= \left[\begin{array}{ccc|c} \widetilde{A} & 0 & \vdots & \widetilde{B}_f \\ B_F\widetilde{C} & A_F & \vdots & 0 \\ \cdots & \cdots & \vdots & \cdots \\ \widetilde{L}_F & -L_F & \vdots & -H_F\begin{bmatrix} 0 \\ I \end{bmatrix} \end{array}\right]$$

$$F(s)\begin{bmatrix} G_{y_0 u}(s) \\ I \end{bmatrix} - G_{y_0 u}(s)$$

$$= \left[\begin{array}{ccc|c} \widetilde{A} & 0 & \vdots & \widetilde{B}_u \\ B_F\widetilde{C} & A_F & \vdots & B_F\begin{bmatrix} 0 \\ I \end{bmatrix} \\ \cdots & \cdots & \vdots & \cdots \\ \widetilde{L}_F & -L_F & \vdots & -H_F\begin{bmatrix} 0 \\ I \end{bmatrix} \end{array}\right] \tag{4.3.5}$$

Applying the BRL to (4.3.5) results in

$$\begin{bmatrix} \begin{bmatrix} \widetilde{A} & 0 \\ B_F\widetilde{C} & A_F \end{bmatrix}^T P + P\begin{bmatrix} \widetilde{A} & 0 \\ B_F\widetilde{C} & A_F \end{bmatrix} & P\begin{bmatrix} \widetilde{B}_d \\ 0 \end{bmatrix} & \begin{bmatrix} \widetilde{L}_F^T \\ -L_F{}^T \end{bmatrix} \\ * & -\gamma_d I & -H_F\begin{bmatrix} 0 \\ I \end{bmatrix} \\ * & * & -\gamma_d I \end{bmatrix} < 0$$

$$\begin{bmatrix} \begin{bmatrix} \widetilde{A} & 0 \\ B_F\widetilde{C} & A_F \end{bmatrix}^T P + P\begin{bmatrix} \widetilde{A} & 0 \\ B_F\widetilde{C} & A_F \end{bmatrix} & P\begin{bmatrix} \widetilde{B}_f \\ 0 \end{bmatrix} & \begin{bmatrix} \widetilde{L}_F^T \\ -L_F{}^T \end{bmatrix} \\ * & -\gamma_f I & -H_F\begin{bmatrix} 0 \\ I \end{bmatrix} \\ * & * & -\gamma_f I \end{bmatrix} < 0$$

$$\begin{bmatrix} \begin{bmatrix} \widetilde{A} & 0 \\ B_F\widetilde{C} & A_F \end{bmatrix}^T P + P\begin{bmatrix} \widetilde{A} & 0 \\ B_F\widetilde{C} & A_F \end{bmatrix} & P\begin{bmatrix} \widetilde{B}_u \\ B_F\begin{bmatrix} 0 \\ I \end{bmatrix} \end{bmatrix} & \begin{bmatrix} \widetilde{L}_F^T \\ -L_F{}^T \end{bmatrix} \\ * & -\gamma_u I & -H_F\begin{bmatrix} 0 \\ I \end{bmatrix} \\ * & * & -\gamma_u I \end{bmatrix} < 0 \tag{4.3.6}$$

where P is $2n_F \times 2n_F$ Lyapunov function matrix and $P > 0$.

Using the same procedure as in Section 3.3, the matrix inequalities in (4.3.6) can be rewritten as

$$\begin{bmatrix} -(V_{11}+V_{11}^T) & -(S_2^T+S_1^T) & V_{11}^T\widetilde{A}+\hat{B}_F\widetilde{C}+\hat{P}_1 & \hat{A}_F+\hat{P}_3^T & V_{11}^T\widetilde{B}_d & 0 & V_{11}^T & S_1^T \\ * & -(S_1+S_1^T) & S_2\widetilde{A}+\hat{B}_F\widetilde{C}+\hat{P}_3 & \hat{A}_F+\hat{P}_2 & S_2\widetilde{B}_d & 0 & S_2 & S_1^T \\ * & * & -\mu\hat{P}_1 & -\mu\hat{P}_3^T & 0 & \widetilde{L}_F^T & 0 & 0 \\ * & * & * & -\mu\hat{P}_2 & 0 & -\hat{L}_F^T & 0 & 0 \\ * & * & * & * & -\gamma_d I & -\begin{bmatrix}0 & I\end{bmatrix}H_F^T & 0 & 0 \\ * & * & * & * & * & -\gamma_d I & 0 & 0 \\ * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_1 & -\frac{1}{\mu}\hat{P}_3^T \\ * & * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_2 \end{bmatrix} < 0$$

$$\begin{bmatrix} -(V_{11}+V_{11}^T) & -(S_2^T+S_1^T) & V_{11}^T\widetilde{A}+\hat{B}_F\widetilde{C}+\hat{P}_1 & \hat{A}_F+\hat{P}_3^T & V_{11}^T\widetilde{B}_f & 0 & V_{11}^T & S_1^T \\ * & -(S_1+S_1^T) & S_2\widetilde{A}+\hat{B}_F\widetilde{C}+\hat{P}_3 & \hat{A}_F+\hat{P}_2 & S_2\widetilde{B}_f & 0 & S_2 & S_1^T \\ * & * & -\mu\hat{P}_1 & -\mu\hat{P}_3^T & 0 & \widetilde{L}_F^T & 0 & 0 \\ * & * & * & -\mu\hat{P}_2 & 0 & -\hat{L}_F^T & 0 & 0 \\ * & * & * & * & -\gamma_f I & -\begin{bmatrix}0 & I\end{bmatrix}H_F^T & 0 & 0 \\ * & * & * & * & * & -\gamma_f I & 0 & 0 \\ * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_1 & -\frac{1}{\mu}\hat{P}_3^T \\ * & * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_2 \end{bmatrix} < 0$$

$$\begin{bmatrix} -(V_{11}+V_{11}^T) & -(S_2^T+S_1^T) & V_{11}^T\widetilde{A}+\hat{B}_F\widetilde{C}+\hat{P}_1 & \hat{A}_F+\hat{P}_3^T & V_{11}^T\widetilde{B}_u+\hat{B}_F\begin{bmatrix}0\\I\end{bmatrix} & 0 & V_{11}^T & S_1^T \\ * & -(S_1+S_1^T) & S_2\widetilde{A}+\hat{B}_F\widetilde{C}+\hat{P}_3 & \hat{A}_F+\hat{P}_2 & S_2\widetilde{B}_u+\hat{B}_F\begin{bmatrix}0\\I\end{bmatrix} & 0 & S_2 & S_1^T \\ * & * & -\mu\hat{P}_1 & -\mu\hat{P}_3^T & 0 & \widetilde{L}_F^T & 0 & 0 \\ * & * & * & -\mu\hat{P}_2 & 0 & -\hat{L}_F^T & 0 & 0 \\ * & * & * & * & -\gamma_u I & -\begin{bmatrix}0 & I\end{bmatrix}H_F^T & 0 & 0 \\ * & * & * & * & * & -\gamma_u I & 0 & 0 \\ * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_1 & -\frac{1}{\mu}\hat{P}_3^T \\ * & * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_2 \end{bmatrix} < 0 \tag{4.3.7}$$

where

$$S_1 = V_{21}^T V_{22}^{-T} V_{21} \tag{4.3.8}$$

$$S_2 = V_{21}^T V_{22}^{-T} V_{12}^T \tag{4.3.9}$$

$$\hat{B}_F = V_{21}^T B_F \tag{4.3.10}$$

$$\hat{P}_1 = P_1 \tag{4.3.11}$$

$$\hat{P}_2 = V_{21}^T V_{22}^{-T} P_2 V_{22}^{-1} V_{21} \tag{4.3.12}$$

$$\hat{P}_3^T = P_3^T V_{22}^{-1} V_{21} \tag{4.3.13}$$

$$\hat{A}_F = V_{21}^T A_F V_{22}^{-1} V_{21} \tag{4.3.14}$$

$$\hat{L}_F = L_F V_{22}^{-1} V_{21} \tag{4.3.15}$$

$$\hat{P} = \begin{bmatrix} \hat{P}_1 & \hat{P}_3^T \\ \hat{P}_3 & \hat{P}_2 \end{bmatrix} \tag{4.3.16}$$

To reduce the filter order, we'll need partitions of the V and P variables as we did in Section 3.3

$$V = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix} \tag{4.3.17}$$

$$P = \begin{bmatrix} P_1 & P_3^T \\ P_3 & P_2 \end{bmatrix} \tag{4.3.18}$$

Using the same method as in Section 3.3, we can solve for the reduced-order filter whose filter order is $k(k \le n_F)$. The partitioned submatrices in (4.3.17) will be dimensionalized as $V_{11}(n_F \times n_F), V_{12}(n_F \times k), V_{21}(k \times n_F)$ and $V_{22}(k \times k)$. We need to enforce some special structure on V_{21} as

$$V_{21} = \begin{bmatrix} \tilde{V}_{21} & 0_{k \times (n_F - k)} \end{bmatrix} \tag{4.3.19}$$

where $\tilde{V}_{21}$ is a $k \times k$ matrix. Finally, we get three LMIs for the reduced-order RFDF with order k as follows:

$$\begin{bmatrix} -(V_{11}+V_{11}^T) & -(S_2^T+\widetilde{S}_1^T) & V_{11}^T\widetilde{A}+\widetilde{B}_F\widetilde{C}+\hat{P}_1 & \widetilde{A}_F+\hat{P}_3^T & V_{11}^T\widetilde{B}_d & 0 & V_{11}^T & \widetilde{S}_1^T \\ * & -(S_1+S_1^T) & S_2\widetilde{A}+\hat{B}_F\widetilde{C}+\hat{P}_3 & \hat{A}_F+\hat{P}_2 & S_2\widetilde{B}_d & 0 & S_2 & S_1^T \\ * & * & -\mu\hat{P}_1 & -\mu\hat{P}_3^T & 0 & \widetilde{L}_F^T & 0 & 0 \\ * & * & * & -\mu\hat{P}_2 & 0 & -\hat{L}_F^T & 0 & 0 \\ * & * & * & * & -\gamma_d I & -\begin{bmatrix}0 & I\end{bmatrix}H_F^T & 0 & 0 \\ * & * & * & * & * & -\gamma_d I & 0 & 0 \\ * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_1 & -\frac{1}{\mu}\hat{P}_3^T \\ * & * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_2 \end{bmatrix} < 0$$

$$\begin{bmatrix} -(V_{11}+V_{11}^T) & -(S_2^T+\widetilde{S}_1^T) & V_{11}^T\widetilde{A}+\widetilde{B}_F\widetilde{C}+\hat{P}_1 & \widetilde{A}_F+\hat{P}_3^T & V_{11}^T\widetilde{B}_f & 0 & V_{11}^T & \widetilde{S}_1^T \\ * & -(S_1+S_1^T) & S_2\widetilde{A}+\hat{B}_F\widetilde{C}+\hat{P}_3 & \hat{A}_F+\hat{P}_2 & S_2\widetilde{B}_f & 0 & S_2 & S_1^T \\ * & * & -\mu\hat{P}_1 & -\mu\hat{P}_3^T & 0 & \widetilde{L}_F^T & 0 & 0 \\ * & * & * & -\mu\hat{P}_2 & 0 & -\hat{L}_F^T & 0 & 0 \\ * & * & * & * & -\gamma_f I & -\begin{bmatrix}0 & I\end{bmatrix}H_F^T & 0 & 0 \\ * & * & * & * & * & -\gamma_f I & 0 & 0 \\ * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_1 & -\frac{1}{\mu}\hat{P}_3^T \\ * & * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_2 \end{bmatrix} < 0$$

$$\begin{bmatrix} -(V_{11}+V_{11}^T) & -(S_2^T+\widetilde{S}_1^T) & V_{11}^T\widetilde{A}+\widetilde{B}_F\widetilde{C}+\hat{P}_1 & \widetilde{A}_F+\hat{P}_3^T & V_{11}^T\widetilde{B}_u+\widetilde{B}_F\begin{bmatrix}0\\I\end{bmatrix} & 0 & V_{11}^T & \widetilde{S}_1^T \\ * & -(S_1+S_1^T) & S_2\widetilde{A}+\hat{B}_F\widetilde{C}+\hat{P}_3 & \hat{A}_F+\hat{P}_2 & S_2\widetilde{B}_u+\hat{B}_F\begin{bmatrix}0\\I\end{bmatrix} & 0 & S_2 & S_1^T \\ * & * & -\mu\hat{P}_1 & -\mu\hat{P}_3^T & 0 & \widetilde{L}_F^T & 0 & 0 \\ * & * & * & -\mu\hat{P}_2 & 0 & -\hat{L}_F^T & 0 & 0 \\ * & * & * & * & -\gamma_u I & -\begin{bmatrix}0 & I\end{bmatrix}H_F^T & 0 & 0 \\ * & * & * & * & * & -\gamma_u I & 0 & 0 \\ * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_1 & -\frac{1}{\mu}\hat{P}_3^T \\ * & * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_2 \end{bmatrix} < 0 \tag{4.3.20}$$

where

$$S_1 = \widetilde{V}_{21}^T V_{22}^{-T}\widetilde{V}_{21} \tag{4.3.21}$$

$$\tilde{S}_1 = \begin{bmatrix} S_1 & 0_{k\times(n_F-k)} \end{bmatrix} \tag{4.3.22}$$

$$S_2 = \tilde{V}_{21}^T V_{22}^{-T} V_{12}^T \tag{4.3.23}$$

$$\hat{B}_F = \tilde{V}_{21}^T B_F \tag{4.3.24}$$

$$\tilde{B}_F = \begin{bmatrix} \hat{B}_F \\ 0_{(n_F-k)\times m} \end{bmatrix} \tag{4.3.25}$$

$$\hat{P}_1 = P_1 \tag{4.3.26}$$

$$\hat{P}_2 = \tilde{V}_{21}^T V_{22}^{-T} P_2 V_{22}^{-1} \tilde{V}_{21} \tag{4.3.27}$$

$$\hat{P}_3^T = P_3^T V_{22}^{-1} \tilde{V}_{21} \tag{4.3.28}$$

$$\hat{A}_F = \tilde{V}_{21}^T A_F V_{22}^{-1} \tilde{V}_{21} \tag{4.3.29}$$

$$\tilde{A}_F = \begin{bmatrix} \hat{A}_F \\ 0_{(n_F-k)\times k} \end{bmatrix} \tag{4.3.30}$$

$$\hat{L}_F = L_F V_{22}^{-1} \tilde{V}_{21} \tag{4.3.31}$$

$$\hat{P} = \begin{bmatrix} \hat{P}_1 & \hat{P}_3^T \\ \hat{P}_3 & \hat{P}_2 \end{bmatrix} \tag{4.3.32}$$

Remark 4.3.1.

The filter matrices (A_F, B_F, L_F) can be derived by means of the following procedure

(i) Compute V_{22} and $\tilde{V}_{21}$ by solving the factorization problem

$$S_1 = \tilde{V}_{21}^T V_{22}^{-T} \tilde{V}_{21} \tag{4.3.33}$$

(ii) Compute A_F, B_F and L_F from

$$A_F = \tilde{V}_{21`}^{-T} \hat{A}_F \tilde{V}_{21}^{-1} V_{22} \tag{4.3.34}$$

$$B_F = \tilde{V}_{21}^{-T} \hat{B}_F \tag{4.3.35}$$

$$L_F = \hat{L}_F \widetilde{V}_{21}^{-1} V_{22} \tag{4.3.36}$$

Now, we need to remember that we had a polytopic uncertain system. As explained in Section 3.3, the parameter dependent Lyapunov matrix $P(\alpha)$

$$P(\alpha) = \sum_{i=1}^{s} \alpha_i P_i(\alpha), \alpha \in \Gamma \tag{4.3.37}$$

is symmetric positive definite for all admissible values of α if and only if all the P_i's are positive definite. Therefore, we need to check (4.3.20) only at the vertices, $i = 1,2,\cdots,s$. This gives us the final form

$$\begin{bmatrix}
-(V_{11}+V_{11}^T) & -(S_2^T+\widetilde{S}_1^T) & V_{11}^T\widetilde{A}_i+\widetilde{B}_F\widetilde{C}_i+\hat{P}_{1,i} & \widetilde{A}_F+\hat{P}_{3,i}^T & V_{11}^T\widetilde{B}_{d,i} & 0 & V_{11}^T & \widetilde{S}_1^T \\
* & -(S_1+S_1^T) & S_2\widetilde{A}_i+\hat{B}_F\widetilde{C}_i+\hat{P}_{3,i} & \hat{A}_F+\hat{P}_{2,i} & S_2\widetilde{B}_{d,i} & 0 & S_2 & S_1^T \\
* & * & -\mu\hat{P}_{1,i} & -\mu\hat{P}_{3,i}^T & 0 & \widetilde{L}_F^T & 0 & 0 \\
* & * & * & -\mu\hat{P}_{2,i} & 0 & -\hat{L}_F^T & 0 & 0 \\
* & * & * & * & -\gamma_d I & -\begin{bmatrix}0 & I\end{bmatrix}H_F^T & 0 & 0 \\
* & * & * & * & * & -\gamma_d I & 0 & 0 \\
* & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_{1,i} & -\frac{1}{\mu}\hat{P}_{3,i}^T \\
* & * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_{2,i}
\end{bmatrix} < 0$$

$$\begin{bmatrix}
-(V_{11}+V_{11}^T) & -(S_2^T+\widetilde{S}_1^T) & V_{11}^T\widetilde{A}_i+\widetilde{B}_F\widetilde{C}_i+\hat{P}_{1,i} & \widetilde{A}_F+\hat{P}_{3,i}^T & V_{11}^T\widetilde{B}_{f,i} & 0 & V_{11}^T & \widetilde{S}_1^T \\
* & -(S_1+S_1^T) & S_2\widetilde{A}_i+\hat{B}_F\widetilde{C}_i+\hat{P}_{3,i} & \hat{A}_F+\hat{P}_{2,i} & S_2\widetilde{B}_{f,i} & 0 & S_2 & S_1^T \\
* & * & -\mu\hat{P}_{1,i} & -\mu\hat{P}_{3,i}^T & 0 & \widetilde{L}_F^T & 0 & 0 \\
* & * & * & -\mu\hat{P}_{2,i} & 0 & -\hat{L}_F^T & 0 & 0 \\
* & * & * & * & -\gamma_f I & -\begin{bmatrix}0 & I\end{bmatrix}H_F^T & 0 & 0 \\
* & * & * & * & * & -\gamma_f I & 0 & 0 \\
* & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_{1,i} & -\frac{1}{\mu}\hat{P}_{3,i}^T \\
* & * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_{2,i}
\end{bmatrix} < 0$$

$$
\begin{bmatrix}
-(V_{11}+V_{11}^T) & -(S_2^T+\widetilde{S}_1^T) & V_{11}^T\widetilde{A}_i+\widetilde{B}_F\widetilde{C}_i+\hat{P}_{1,i} & \widetilde{A}_F+\hat{P}_{3,i}^T & V_{11}^T\widetilde{B}_{u,i}+\widetilde{B}_F\begin{bmatrix}0\\ I\end{bmatrix} & 0 & V_{11}^T & \widetilde{S}_1^T \\
* & -(S_1+S_1^T) & S_2\widetilde{A}_i+\hat{B}_F\widetilde{C}_i+\hat{P}_{3,i} & \hat{A}_F+\hat{P}_{2,i} & S_2\widetilde{B}_{u,i}+\hat{B}_F\begin{bmatrix}0\\ I\end{bmatrix} & 0 & S_2 & S_1^T \\
* & * & -\mu\hat{P}_{1,i} & -\mu\hat{P}_{3,i}^T & 0 & \widetilde{L}_F^T & 0 & 0 \\
* & * & * & -\mu\hat{P}_{2,i} & 0 & -\hat{L}_F^T & 0 & 0 \\
* & * & * & * & -\gamma_u I & -\begin{bmatrix}0 & I\end{bmatrix}H_F^T & 0 & 0 \\
* & * & * & * & * & -\gamma_u I & 0 & 0 \\
* & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_{1,i} & -\frac{1}{\mu}\hat{P}_{3,i}^T \\
* & * & * & * & * & * & * & -\frac{1}{\mu}\hat{P}_{2,i}
\end{bmatrix} < 0
$$

$(i = 1,2,\cdots,s)$

(4.3.38)

Theorem 4.3.2

The reduced-order RFDF can be obtained by solving

$$
\min_{V_{11},S_1,S_2,\hat{A}_F,\hat{B}_F,\hat{L}_F,H_F,\gamma_d,\gamma_f,\gamma_u,\hat{P}_i} \left\{\alpha_d\gamma_d+\alpha_f\gamma_f+\alpha_u\gamma_u : (4.3.38)\right\} \tag{4.3.39}
$$

where $\hat{P}>0$ and α_d, α_f and α_u are given as positive scalars. ∎

In summary, the reduced order RFDF filter in the multi-objective H_∞ formulation for polytopic uncertain system can be solved if and only if (4.3.38) holds for all vertices, $i = 1,2,\cdots,s$. The minimum value can be found from (4.3.39). Also, the filter realization (A_F, B_F, L_F, H_F) defining the k_{th} order filter is obtained from (i) and (ii) in ***Remark 4.3.1*** and (4.3.39).

4.4 Examples

4.4.1 Example 1

In this section, a numerical example is given. Consider the uncertain LTI plant that is borrowed from [71], but is modified to include plant uncertainties. The plant is

$$\dot{x} = \begin{bmatrix} 0 & -1+0.3\alpha \\ 1 & -0.5 \end{bmatrix} x + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u + \begin{bmatrix} 0 \\ 1 \end{bmatrix} f + \begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix} d \qquad (4.4.1.1)$$
$$y = \begin{bmatrix} -100+10\beta & 100 \end{bmatrix} x + 0.1u + f + 0.05d$$

where the uncertainty parameters, α and β, have the uncertainty set given by

$$|\alpha| \le 1, |\beta| \le 1 \qquad (4.4.1.2)$$

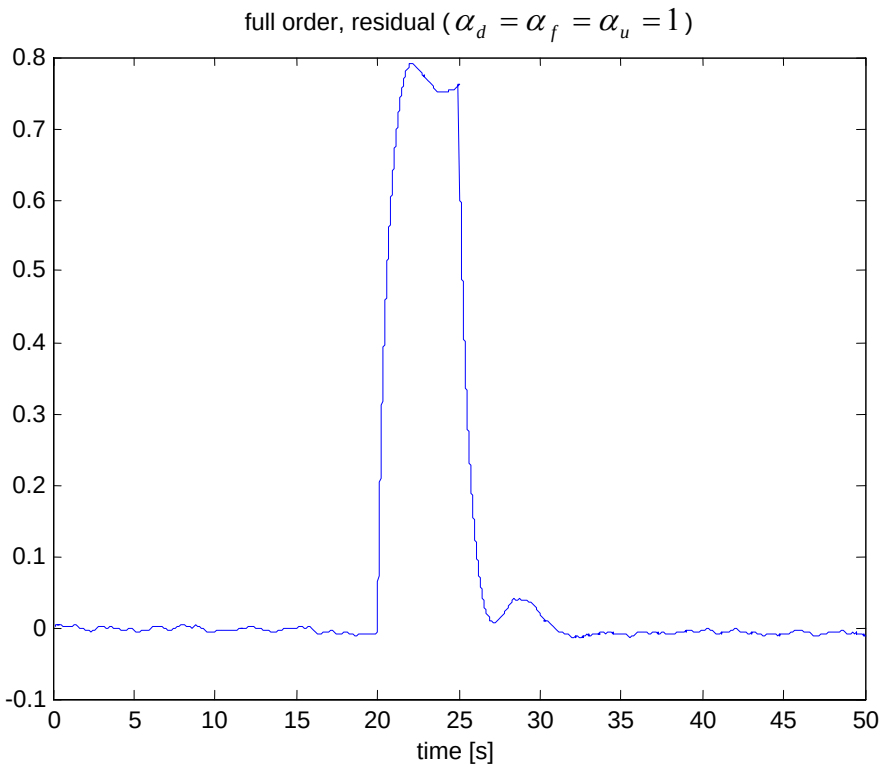
The selected reference model is given from (4.2.1.13). The fault signal f is simulated as a pulse of unit amplitude that occurs from 20 to 25 seconds and is zero elsewhere. The input u is taken as $1-e^{-0.01t}$. The unknown input d is assumed to be band-limited white noise with power 0.0005 and the upperbound of $\|d\|_{RMS,T}$ is given as 0.15. This value is used to calculate J_{th} of (4.2.2.4). With these information, we can get the RFDF data from (4.3.39).

We checked the detection time with weights set at ($\alpha_d = \alpha_f = \alpha_u = 1$) and ($\alpha_d = \alpha_f = 1, \alpha_u = 100$). The higher value for α_u means that we place more emphasis on filtering the control signal rather than tracking the fault signal. The detection time is defined as the time span where the J-residual (time-windowed residual RMS value) exceeds the threshold, J_{th}. From (4.3.3), the full order FDF is 4(=n_F). As the order k of the RFDF is reduced from 4 to 1, the effectiveness of the filter is compared by monitoring the detection time. The results are shown in Figures 7 through 11. Table 4 shows a comparison of the results. The adaptive threshold changes according to the control input u. In this example, we can see that the fault detection time does not change much as the order of the RFDF is reduced. From [72] [73], we also know that the time window size of the threshold calculation and the proper selection of threshold are also important factors in effective fault detection.

TABLE 4

COMPARISON OF THE DETECTION TIME OF RFDF WITH DIFFERENT ORDERS

RFDF Order (k)		Detection time
$k = n_F = 4$ (full)	$\alpha_d = \alpha_f = \alpha_u = 1$	20.705~34.96
	$\alpha_d = \alpha_f = 1, \alpha_u = 100$	22~34.7
$k = 3$, ($\alpha_d = \alpha_f = \alpha_u = 1$)		22.99~31.36
$k = 2$, ($\alpha_d = \alpha_f = \alpha_u = 1$)		23.115~31.45
$k = 1$, ($\alpha_d = \alpha_f = \alpha_u = 1$)		23.58~31.157



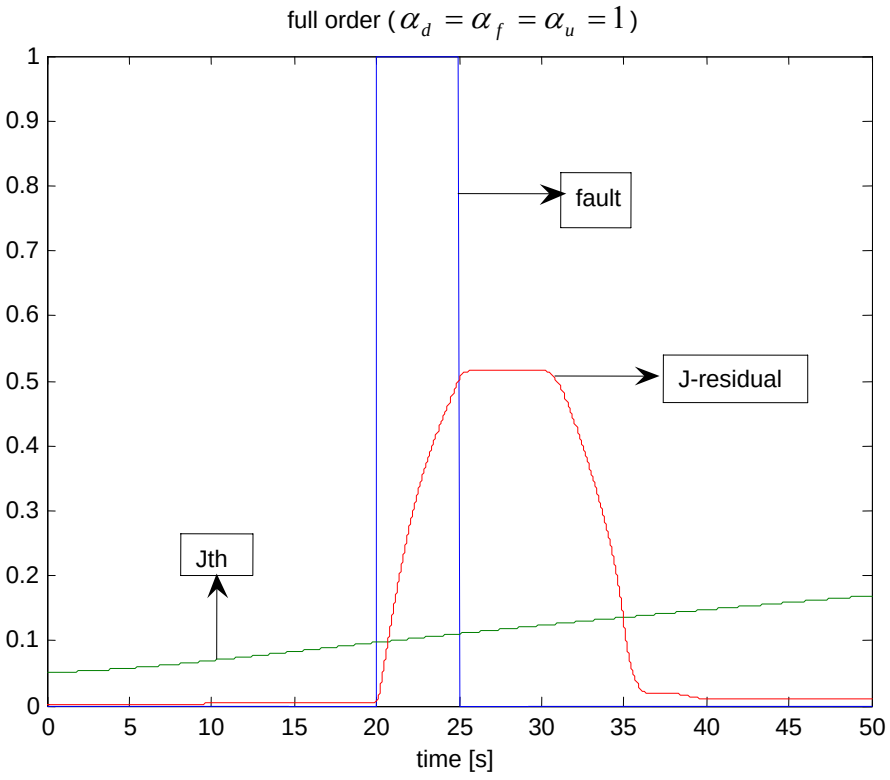
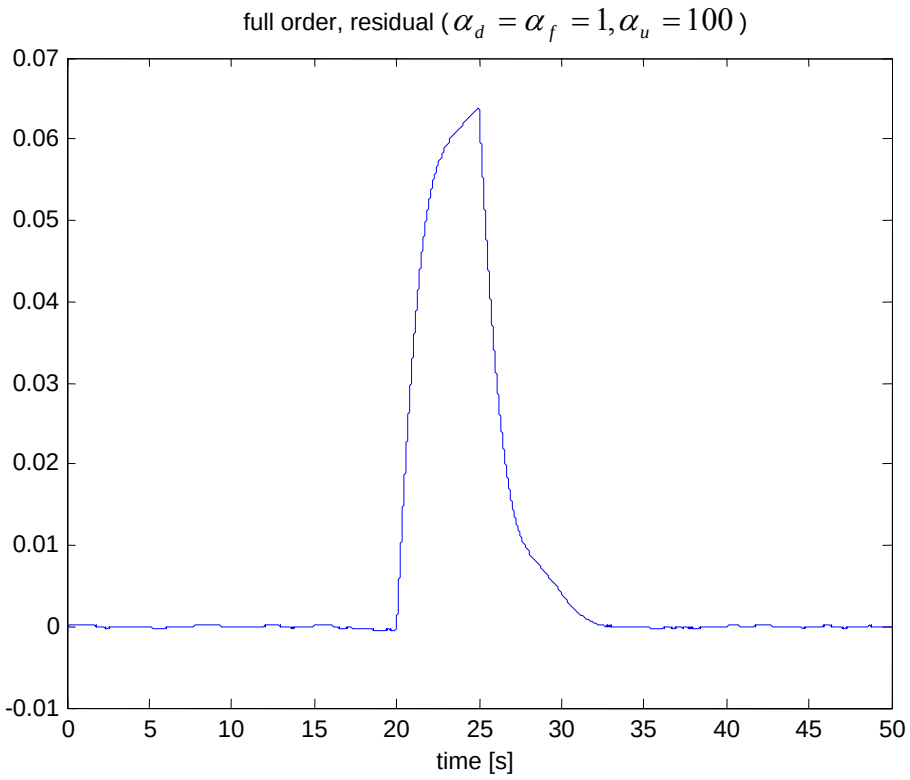


Figure 7. Full order response, ($\alpha_d = \alpha_f = \alpha_u = 1$)



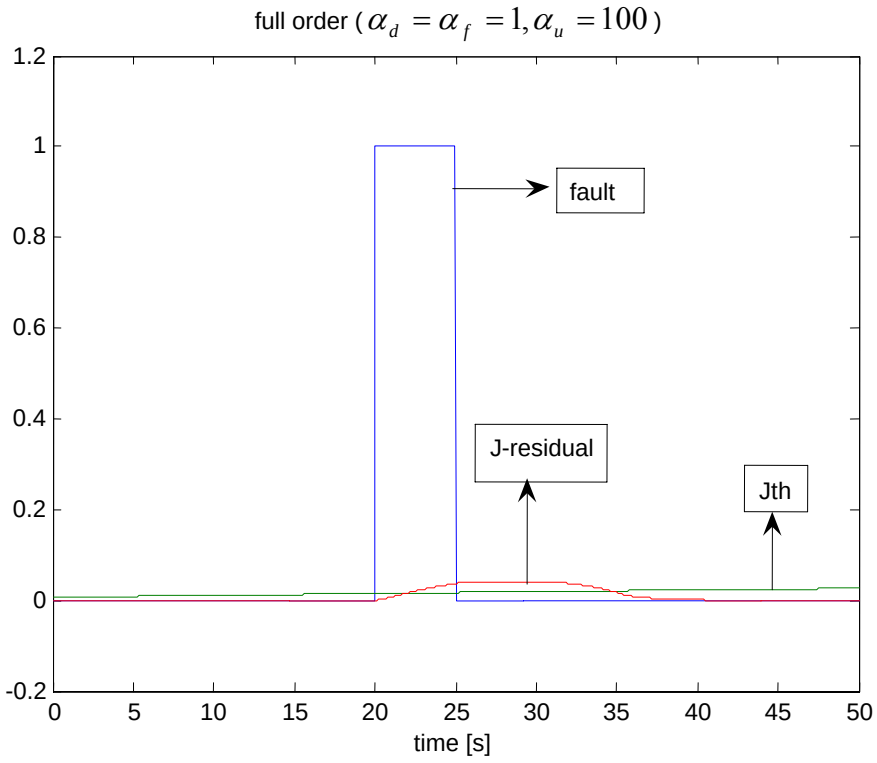


Figure 8. Full order response, ($\alpha_d = \alpha_f = 1, \alpha_u = 100$)

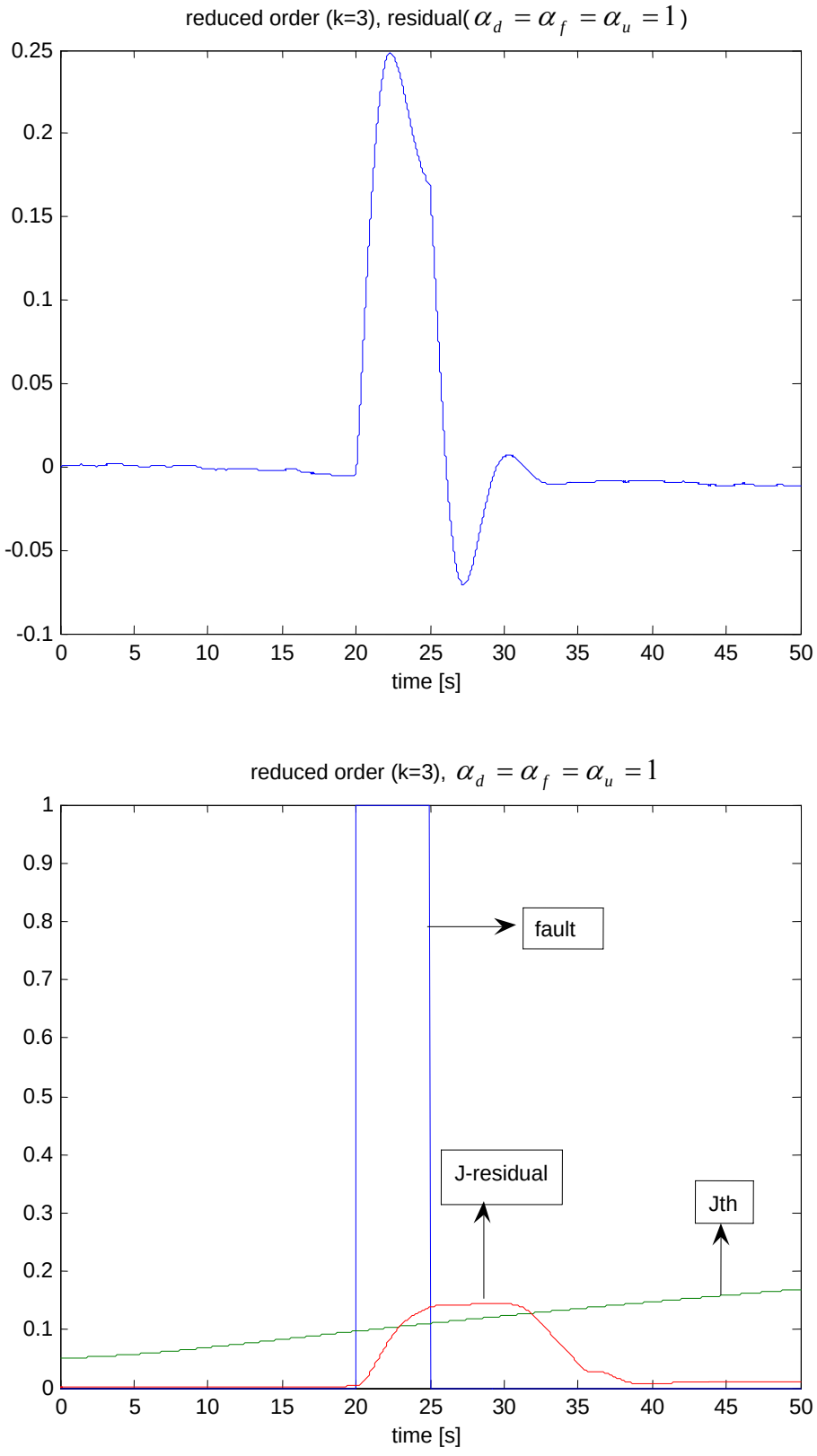


Figure 9. Reduced order (k=3) response, ($\alpha_d = \alpha_f = \alpha_u = 1$)

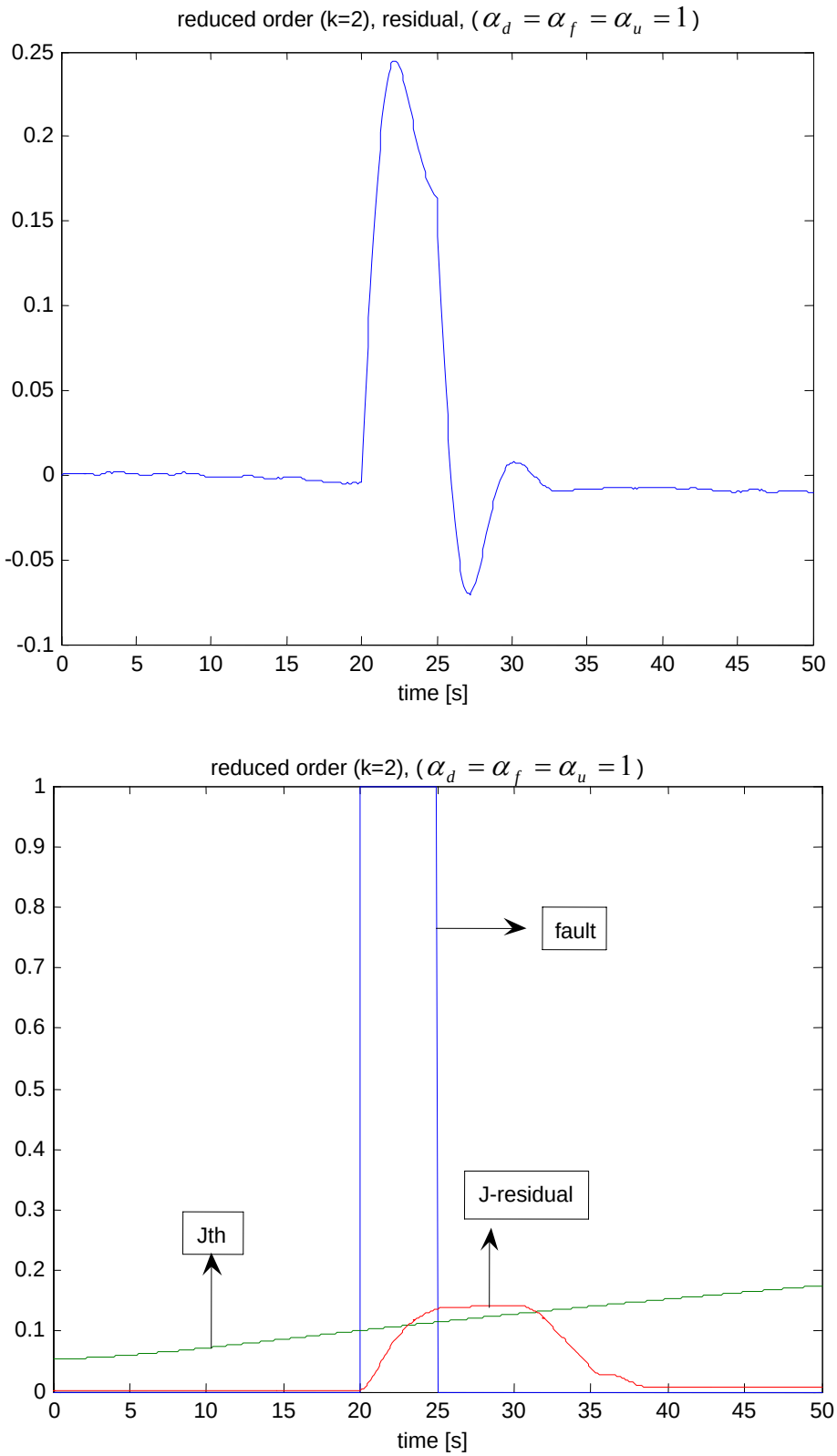


Figure 10. Reduced order (k=2) response, ($\alpha_d = \alpha_f = \alpha_u = 1$)

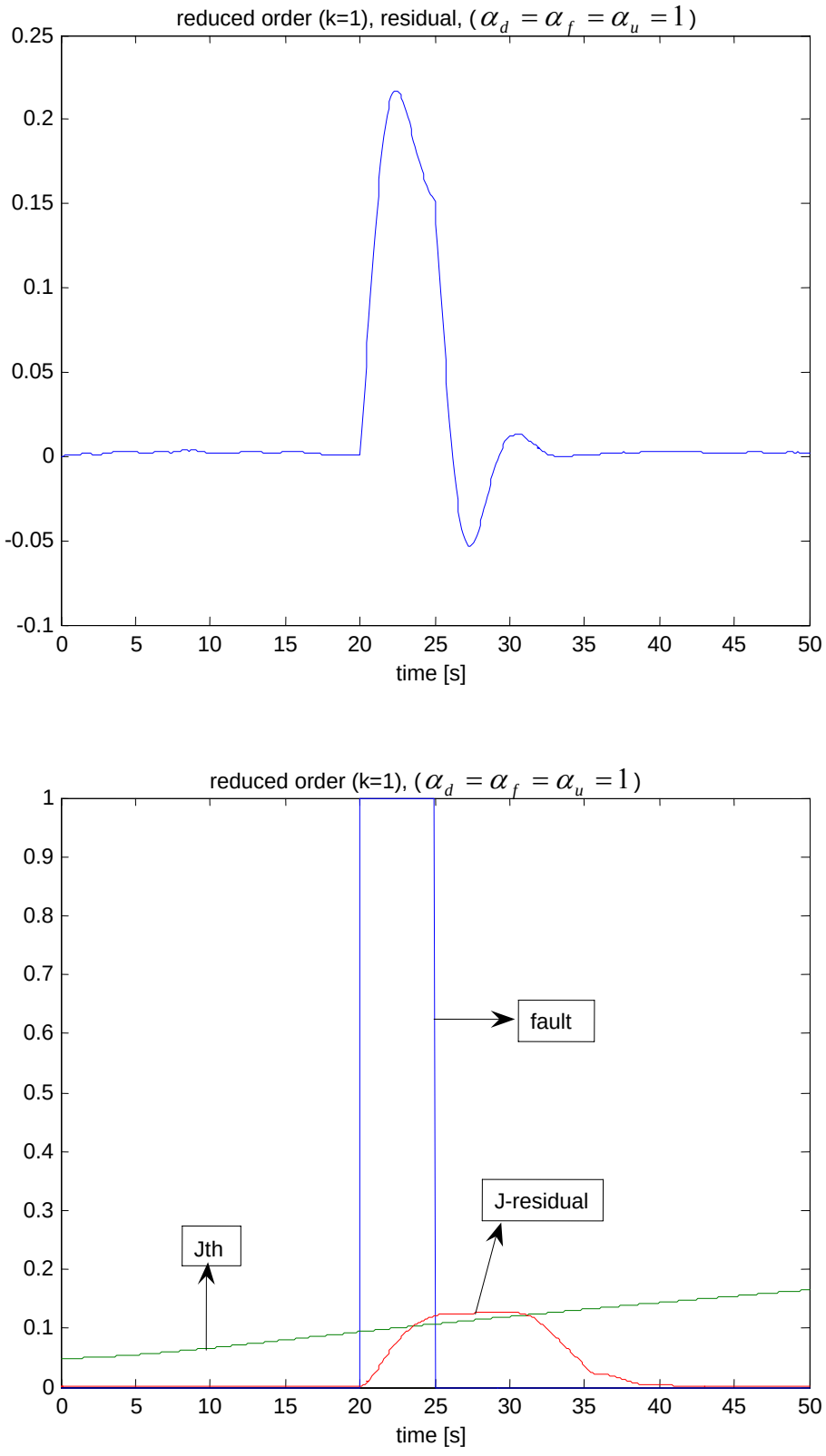


Figure 11. Reduced order (k=1) response, ($\alpha_d = \alpha_f = \alpha_u = 1$)

4.4.2 Example 2

Here, the order reduction technique of the fault detection filter is applied to the more complicated plant. The system is borrowed from [61]. The system is as below

$$\left\{\begin{array}{c} y_1 = \dfrac{k_3 s}{s^2+\theta_1 s+\theta_2}(u(s)+f(s))+\dfrac{k_1 k_3 s}{(sT_1+1)(s^2+\theta_1 s+\theta_2)}d_1(s) \\ y_2 = \dfrac{k_3}{s^2+\theta_1 s+\theta_2}(u(s)+f(s))+\dfrac{k_1 k_3}{(sT_1+1)(s^2+\theta_1 s+\theta_2)}d_1(s)+\dfrac{k_2(sT_2+1)}{sT_3+1}d_2(s) \end{array}\right\} \quad (4.4.2.1)$$

where $T_1 = 0.1s$, $T_2 = 10s$, $T_3 = 0.2s$, $k_1 = 0.3$, $k_2 = 0.2$, $k_3 = 10$ and the parameters (θ_1, θ_2) belong to the intervals

$$0.5 \le \theta_1 \le 1.2,\ 1 \le \theta_2 \le 1.5 \quad (4.4.2.2)$$

In this example, the fault signal is superimposed on the control signal $u(t)$. The control signal is given as $u(t) = 1 - e^{-0.001t}$ and the fault signal occurs as below

$$f(t) = \left\{\begin{array}{ll} 0, & t < 50s \\ 1, & 50s \le t \le 100s \\ 0, & t > 100s \end{array}\right\} \quad (4.4.2.3)$$

The signals $d_1(t)$ and $d_2(t)$ are assumed to be unitary variance white noises.

In this case, the reference model $W(s)$ is chosen to detect faults in the frequency range of between 0 and 1 rad/s with a -40dB/dec roll-off at higher frequencies. To match the fault dimension of the output of the plant, the reference model is selected as

$$W(s) = \begin{bmatrix} \dfrac{1}{s^2+2s+1} \\ \dfrac{1}{s^2+2s+1} \end{bmatrix} \quad (4.4.2.4)$$

In this example, a tracking filter is not inserted between the plant and the filter. Thus, the full order of the filter is 8 (plant order + reference model order). There are four vertices and for

each vertex, a different Lyapunov function P is used to calculate (4.2.1.14) so a less conservative result can be obtained. The designed filter data (A_F, B_F, L_F, H_F) has eight states, three inputs and two outputs in the full order case. To check the effectiveness of the order reduction, the filter order is gradually reduced from eight to one. During the simulations, the uncertain plant parameters have been kept constant at one of four vertices, $\theta_1 = 1.2$ and $\theta_2 = 1$. The worst disturbance value, which is used to calculate J_{th} in (4.2.2.4), was measured on-line to be 0.015.

The results are shown at the following figures. The fault can be detected until the filter order is reduced to the sixth order, but it can not be detected from the fifth order and below. The Matlab code for the sixth order filter is given in Appendix C and the Simulink diagram is given in Appendix D. In this example, the weighting constants (α_u, α_f, α_d) are fixed to one. This example was chosen to compare the dissertation results with those in [61]. Unfortunately, there were many problems with the results in [61] and no comparison was possible.

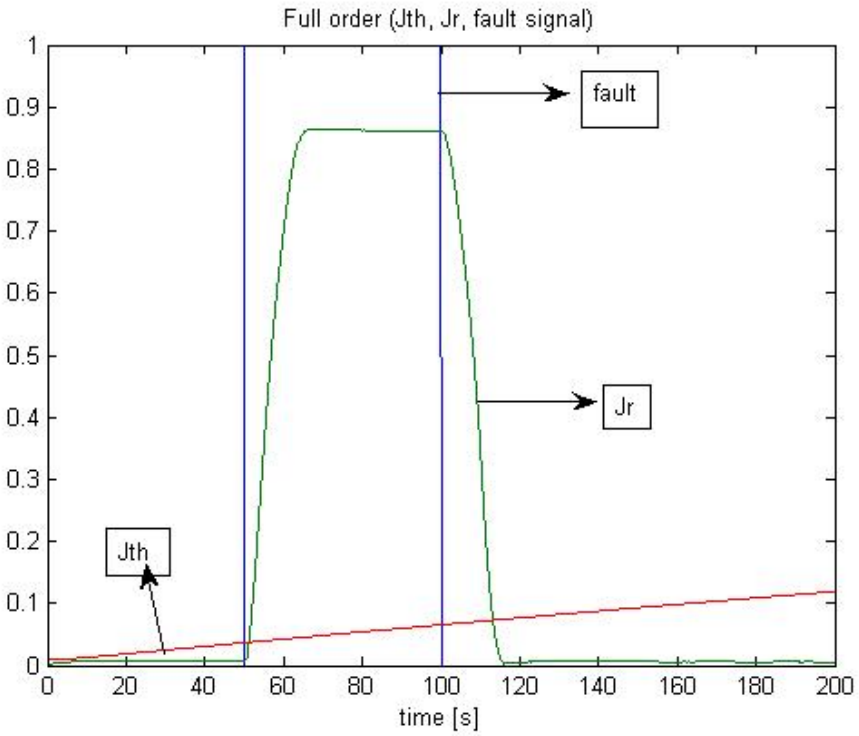


Figure 12. Full order response (k=8)

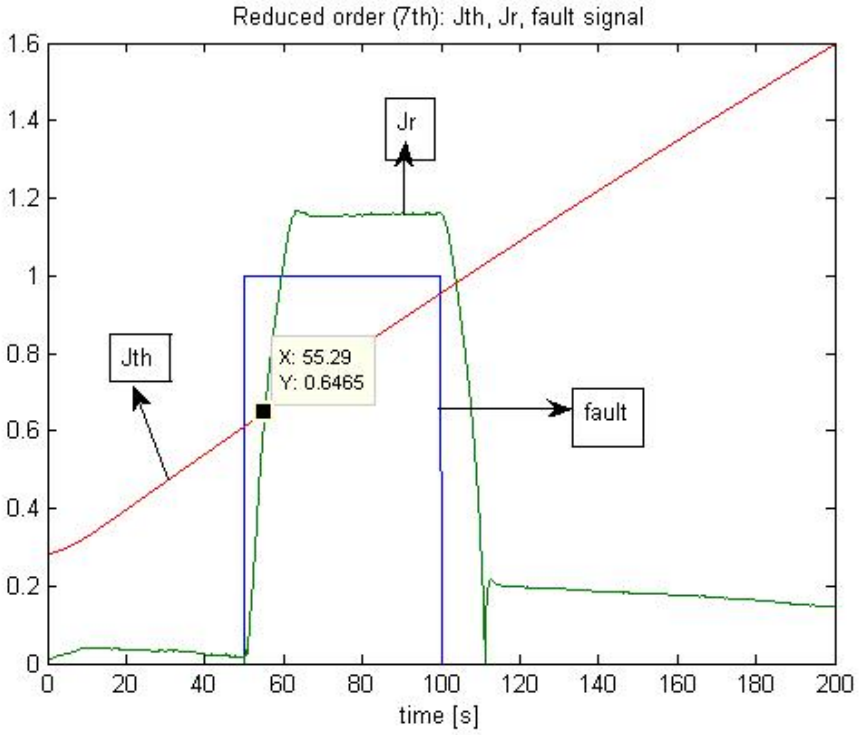


Figure 13. Reduced order response (k=7)

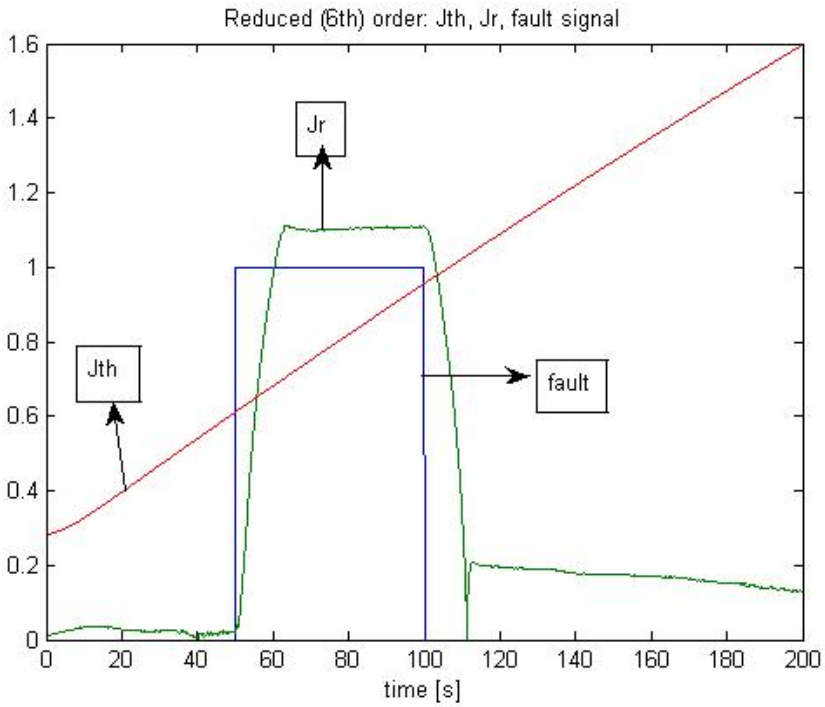


Figure 14. Reduced order response (k=6)

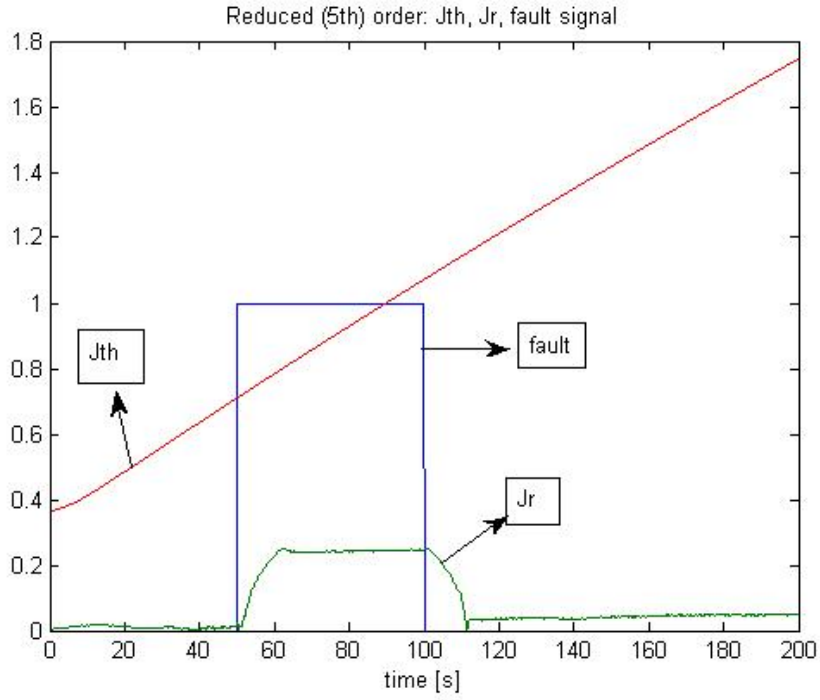


Figure 15. Reduced order response (k=5)

CHAPTER 5

SUMMARY AND FUTURE WORK

5.1 Summary

In this work, we developed in detail a practical approach for the H_∞ reduced filter synthesis problem in an LMI framework [74,75]. The LMI characteristics of filtering problem were described and formulated with the BRL. To overcome conservatism, parameter-dependent Lyapunov functions were used. A congruence transformation and a change of variable technique were used to linearize the problem. This formulation had formerly been applied to the H_2 problem. As is well-known, the H_∞ approach provides better robustness than the H_2 approach.

In the chapter on fault detection, the order reduction of an RFDF was considered and the results were illustrated with two examples [75,76]. The main approach was rooted in our reduced order filter design. First, the reference model was introduced and the problem was formulated in multi-objective form. For the first example considered, the detection time did not change much as the filter order was reduced. In other words, the order reduction does not degrade the detection performance. In the second example, this technique was applied to a MIMO (Multiple Input Multiple Output) plant with two disturbance inputs and two outputs. We found that our order reduction technique was also applicable to MIMO system. Thus, our reduced-order RFDF design shows promising results.

Key contributions were:

- Development of H_∞ filter design using LMIs for polytopic uncertain system: More robust than H_2 filter
- Development of the order reduction technique for full order H_∞ filter
- Less conservative results were found using a different Lyapunov matrix for each vertice
- Development of the order reduced multi-objective H_∞ fault detection filter design
- Application of the order reduced RFDF design technique to MIMO system

5.2 Future Work

The limit of LMI technique is that the simulation time increases exponentially as the order of plant increase. Thus, the randomized algorithm approach is now researched.

The relationship between the time window size and the false alarm rate is an important area of future research. The proper selection of the time window size severely affects the fault detectibility of this estimator-based fault detection filter. There are many kinds of threshold selection methods. Even though, zero false alarm and zero miss alarm are almost impossible in real world, more research should be done about the residual evaluation technique.

LIST OF REFERENCES

LIST OF REFERENCES

[1] B. D. O. Anderson, J. B. Moore, *Optimal filtering*, Englewood Cliffs, NJ, Prentice Hall, 1979.

[2] K. M. Nagpal, P. P. Khargoneckar, "Filtering and smoothing in an H_∞ setting," *IEEE Transactions on Automatic Control*, 36(2), 1991, pp. 152-166.

[3] U. Shaked, Y. Theodor, "H_∞ optimal estimation: a tutorial," *Proceedings of Conference on Decision and Control*, Tucson, USA, 1992, pp. 2278-2286.

[4] M. Fu, C. E. De Souza, L. Xie, "H_∞ estimation for uncertain systems," *International Journal of Robust and Nonlinear Control*, vol. 4, 1994, pp. 421-448.

[5] E. Friedman, U. Shaked, "A new H_∞ design for linear time delay systems," *IEEE Transactions on Signal Processing*, vol. 49, 2001, pp. 2839-2843.

[6] M. Fu, "Interpolation approach to H_∞ optimal estimation and its interconnection to loop transfer recovery," *System and Control Letters*, vol. 17, 1991, pp. 29-36.

[7] H. S. Kim, C. S. Sims, K. M. Nagpal, "Reduced order filtering in an H_∞ setting," *Proceedings of American Control Conference*, Chicago, USA, 1992, pp. 1876-1877.

[8] K. M. Nagpal, R. E. Helmick, C. S. Sims, "Reduced order estimation, I. Filtering," *International Journal of Control*, vol. 45, 1987, pp. 1867-1888.

[9] K. M. Grigoriadis, J. T. Watson, "Reduced order H_∞ and $L_2 - L_\infty$ filtering via linear matrix Inequalities," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 33, 1997, pp. 1326-1338.

[10] S. Xu, T. Chen, "Reduced order H_∞ filtering for stochastic systems," *IEEE Transactions on Signal Processing*, vol. 50, no. 12, 2002, pp. 2998-3007.

[11] M. Bettayeb, D. Kavranoglu, "Reduced order H_∞ filtering," *Proceedings of American Control Conference*, 1994, Baltimore, USA, pp. 1884-1888.

[12] R. T. O'Brien Jr., Kiriakos Kiriakidis, "Reduced Order H_∞ Filtering for Discrete Time Linear Time Varying Systems," *Proceedings of American Control Conference*, Boston, USA, 2004, pp. 4120-4125.

[13] S. Boyd, L. E. Ghaoui, E. Feron, V. Balakrishnan, *Linear matrix inequalities in system and control theory*, SIAM studies in applied mathematics, Philadelphia, 1994.

[14] R. E. Skelton, T. Iwasaki, K. Grigoriadis, *A unified algebraic approach to linear control design*, Taylor & Francis, 1998.

[15] H. D. Tuan, P. Apkarian, T. Q. Nguyen, "Robust and reduced order filtering: New Characterizations and methods," *Proceedings of American Control Conference*, Chicago, USA, 2000, pp. 1327-1331.

[16] Y. Nesterov, A. Nemirovski, "*Interior point polynomial methods in convex programming: theory and applications*," Philadelpia, PA, SIAM, 1994.

[17] P. Gahinet, A. Nemirovski, A. J. Laub, M. Chilali, "The LMI control toolbox, a tutorial," *Matlab v7.0*, Mathworks Inc.

[18] A. Ben-Tal, A. S. Nemirovski, "*Robust convex programming*," working paper, Faculty at Technion, Israel.

[19] D. P. Bertsekas, "*Nonlinear programming*," Athena Scientific, second edition, 1999.

[20] L. E. Gahoui, F. Oustry, H. Levert, "Robust solutions to uncertain semidefinite programs," *SIAM Journal of Control and Optimization*, vol. 9, No. 1, 1998, pp. 33-52.

[21] M. C. Oliveira, "Stability test for constrained linear systems," *Lecture notes in control and information sciences*, vol. 268, 2001, pp. 241-257.

[22] H. P. Horisberger, P. R. Belanger, "Regulators for linear time invariants plants with uncertain parameters," *IEEE Transactions on Automatic Control*, 1976, pp. 705-707.

[23] J. C. Geromel, P. L. D. Peres, J. Bernussou, "On a convex parameter space method for linear control design of uncertain systems," *SIAM Journal of Control and Optimization*, vol. 29, No. 2, 1991, pp. 381-402.

[24] T. Iwasaki, R. E. Skelton, "All controllers for the general H_∞ control problem: LMI existence conditions and state space formulas," *Automatica*, vol. 30, 1994, pp. 1307-1317.

[25] P. Gahinet, P. Apkarian, "A linear matrix inequality approach to H_∞ control," *International Journal of Robust and Nonlinear Control*, vol. 4,1994, pp. 421-448.

[26] G. S. Becker, "*Quadratic stability and performance of linear parameter dependent systems*," Ph.D dissertation, Dept. of Mech. Eng., University of California at Berkely, 1993.

[27] P. Apkarian, R. J. Adams, "Advanced gain scheduling techniques for uncertain systems," *IEEE Transactions on Control System Technology*, vol. 6, No. 1, 1998.

[28] E. Feron, P. Apkarian, P. Gahinet, "Analysis and synthesis of robust control systems via parameter dependent Lyapunov functions," *IEEE Transactions on Automatic Control*, vol. 42, No. 7, 1996, pp. 1041-1046.

[29] P. C. Parks, "A new proof of the Routh-Hurwitz stability criterion using second method of Lyapunov," *Proc. Cambridge Philos. Soc*., 58, 1962, pp. 669-672.

[30] V. M. Popov, "Absolute stability of nonlinear systems of automatic control," *Automatic Remote Control*, 22, 1962, pp. 857-875.

[31] W. M. Haddad, D. S. Bernstein, "Parameter dependent Lyapunov functions, constant real parameter uncertainty and Popov criterion in robust analysis and synthesis: Part 1 and 2," *Proceedings of Conference on Decision and Control*, Brighton, England, 1991, pp. 2262-2263, pp. 2274-2279.

[32] P. Gahinet, P. Apkarian, M.Chilali, "Parameter dependent Lyapunov functions for real parametric uncertainty," *IEEE Transactions on Automatic Control*, vol. 41, 1996, pp. 436-442.

[33] J. Yu, A. Sideris, "H_∞ control with parametric Lyapunov functions," *System and Control Letters*, vol. 30, 1997, pp. 57-69.

[34] H. D. Tuan, P. Apkarian, "Relaxations of parametrized LMIs with control applications," *Proceedings of Conference on Decision and Control*, Tampa FL, USA, 1998, pp. 1747-1752.

[35] P. Apkarain, H. D. Tuan, J. Bernussou, "Analysis, eigenstructure assignment and H_2 multi-channel synthesis with enhanced LMI characterizations," *Proceedings of Conference on Decision and Control*, Sydney, Australia, 2000, pp. 1489-1494.

[36] H. D. Tuan, P. Apkarian, T. Q. Nguyen, "Robust filtering for uncertain nonlinearly parametrized plants," *IEEE Transactions on Signal Processing*, vol. 58, No. 7, 2003.

[37] P. Apkarian, H. D. Tuan, J. Bernussou, "Continuous time analysis and H_2 multi channel synthesis with enhanced LMI characterizations," *Proceedings of Conference on Decision and Control*, Sydney, Australia, 2000, pp. 1489-1494.

[38] J. C. Geromel, "Optimal linear filtering under parameter uncertainty," *IEEE Transactions on Signal Processing*, vol. 47, 1999, pp. 168-175.

[39] C. E de Souza, A. Trofino, "An LMI approach to the design of robust H_2 filters," *Recent advances in linear matrix inequality methods in control*, Philadelpia, PA:SIAM, 1999.

[40] A. Kanchanaharuthai, P. Ngamsom, "Robust H_∞ load frequency control for interconnected power systems with D-stability constraints via LMI approach," *Proceedings of American Control Conference*, Portland, USA, 2005, pp. 4387-4392.

[41] R. N. Clark, "Instrument fault detection," *IEEE Transactions on Aerospace and electronic systems*, vol. 14, No. 3, 1978.

[42] M. A. Massoumnia, G. C. Verghese, A. S. Willsky, "Failure detection and identification," *IEEE Transactions on Automatic Control,* vol. 34, No. 3, 1989.

[43] J. Chen, H. Y. Zhang, "Robust detection of faulty actuators via unknown input observers," *Int. J. System Sci.,* vol. 22, No. 10, 1991, pp. 1829-1839.

[44] J. Chen, R. J. Patton, "Obserer-based fault detection and isolation: robustness and applications," *Control Engineering Practice,* vol. 5, No. 5, 1997, pp. 671-682.

[45] J. Chen, R. J. Patton, H. Y. Zhang, "Design of unknown input observers and robust fault detection filters," *Int. J. of Control,* vol. 63, No. 1, 1996, pp. 85-105.

[46] R. S. Mangoubi, B. D. Appleby, G. C. Verghese, W. E. Vandervelde, "A robust failure detection and isolation algorithm," *Proceedings of Conference on Decision and Control,* New Orleans, USA, 1995, pp. 2377-2382.

[47] P. M. Frank "Enhancement of robustness in observer-based fault detection," *Int. J. Contr.,* vol. 59, No. 4, 1994, pp. 955-981.

[48] S. X. Ding, T. Jeinsch, P. M. Frank, E. L. Ding, "A unified approach to the optimization of fault detection systems," *Int. J. Adaptive Control Signal Process,* vol. 14, 2000, pp. 725-745.

[49] Z. Qiu, J. Gertler, "Robust FDI systems and H_∞ optimization," *Proceedings of 32nd Conf. on Decision and Control*, San Antonio, USA, 1993, pp. 1710-1715.

[50] J. Liu, J. L. Wang, G. H. Yang, "Worst case fault detection observer design: An LMI approach," *Proceedings of Int. Conf. on Control and Automation,* Xiamen, China*, 2002*, pp. 1243-1247.

[51] J. Liu, J. L. Wang, G. H. Yang, "An LMI approach to worst case analysis for fault detection observers," *Proceedings of American Control Conference,* Denver, USA, 2003, pp. 2985-2990.

[52] H. Wang, J. Wang, J. Lam, "An optimization approach for worst case fault detection observer design," *Proceedings of American Control Conference*, Boston, USA, 2004, pp. 2474-2480.

[53] M. Zhong, S. X. Ding, J. Lam, H. Wang, "An LMI approach to design robust fault detection filter for uncertain LTI systems," *Automatica,* 39, 2003, pp. 543-550.

[54] R. J. Patton, J. Chen, "Robust fault detection and isolation (FDI) systems," *Advances in theory and applications, Control and Dynamic Systems,* Academic Press*, 1996*, pp. 171-224

[55] J. Chen, R. J. Patton, "*Robust model-based fault diagnosis for dynamic systems*," Kluwer Academic Publishers, 1999.

[56] J. Liu, J. L. Wang, G. H. Yang, "An LMI approach to worst case analysis for fault detection observers," *Proceedings of American Control Conference,* Denver, USA, 2003, pp. 2985-2990.

[57] S. X. Ding, P. M. Frank, "Threshold calculation using LMI-technique and its integration in the design of fault detection systems," *Proceedings of 42nd Conf. on Decision and Contol,* Maui, Hawaii, 2003, pp. 469-474.

[58] P. M. Frank, S. X. Ding, "Survey of robust residual generation and evaluation methods in observer-based fault detection systems," *J. Process Contr.,* vol. 7, No. 6, 1997, pp. 403-424.

[59] A. Casavola, D. Famularo, G. Franze, "Robust fault isolation of uncertain linear systems: An LMI approach," *Proceedings of IFAC fault detection supervision and safety of technical processes,* Washington D.C., USA, 2003

[60] S. X. Ding, P. M. Frank, E. L. Ding, T. Jeinsch, "Fault detection system design based on a new trade-off strategy," *Proceedings of 39th Conf. on Decision and Contol.,* Sydney, Australia, 2000, pp. 4144-4149.

[61] A. Casavola, D. Famularo, G. Franze, "A robust deconvolution scheme for fault detection and isolation of uncertain linear systems: an LMI approach," *Automatica,* 41, 2005, pp. 1463-1472.

[62] A. Casavola, D. Famularo, G. Franze, "Robust fault detection of uncertain linear systems via quasi-LMIs," *Proceedings of American Control Conference,* Portland USA, 2005, pp. 1654-1659.

[63] R. J. Patton, M. Hou, "On sensitivity of robust fault detection observers," *Proceedings of 14th IFAC world congress,* Beijing, China, 1999, pp. 67-72.

[64] M. L. Rank, H. Niemann, "Norm based design of fault detectors," *Int. J. Contr.,*vol. 72, No. 9, 1999, pp. 773-783.

[65] H. Niemann, J. Stoustrup, "Design of fault detectors using H_∞ optimization," *Proceedings of 39th IEEE Conf. on Decision and Control,* Sydney, Australia, 2000, pp. 4327-4328.

[66] E. Frisk, "*Residual generation for fault diagnosis*," Ph.D dissertation, Dept. of Electrical Engineering, Linkoping University, Sweden, 2001.

[67] K. Astrom, B. Wittenmark, "*Computer controlled systems-Theory and Design*," Prentice Hall, 1994.

[68] K. Astrom, B. Wittenmark, "*Adaptive Control*," Prentice Hall, 1989.

[69] K. Zhou, J. C. Doyle, K. Glover, "*Robust and Optimal Control*," Prentice Hall, 1996.

[70] S. Boyd, C. Barratt, "*Linear Controller Design, Limits of Performance*," Prentice Hall, 1991, pp. 98-99.

[71] E. G. Nobrega, M. O. Abdalla, K. M. Grigoriadis, "LMI-based filter design for fault detection and isolation," *Proceedings of 39th Conf. on Decision and Control,* Sydney, Australia, 2000, pp. 4329-4334.

[72] A. Emami-Naeini, M. M. Akhter, S. M. Rock, "Effect of model uncertainty on failure detection: the threshold selector," *IEEE Trans. Automatic Control,* vol. 33, No. 12, 1988, pp. 1106-1115.

[73] F. Rambeaux, F. Hamelin, D. Sauter, "Optimal thresholding for robust fault detection of uncertain systems," *Int. J. Robust Nonlinear Control,* 2000, vol. 10, pp. 1155-1173.

[74] Young-Man Kim, John M. Watkins, "A new approach for robust and reduced order H_∞ filtering," *Proceedings of American Control Conference,* Minneapolis, USA, 2006, Accepted.

[75] Young-Man Kim, John M. Watkins, "Robust and reduced order H_∞ filtering via LMI approach and its application for fault detection," *System and Control Letters*, In-Preparation.

[76] Young-Man Kim, John M. Watkins, "A new approach for robust and reduced order fault detection filter design," *Proceedings of Conference on Decision and Control*, 2006, San Diego, USA, Submitted.

APPENDICES

APPENDIX A

MATLAB CODE TO SYNTHESIZE THE FULL ORDER FILTER FOR TABLE 3

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                Synthesis Part
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A1=[0 -0.1;1 -0.5];
B1=[-2 0;1 0];
C1=[-90 100];
D1=[0 1];
L1=[1 0];

A2=[0 -0.1;1 -0.5];
B2=[-2 0;1 0];
C2=[-110 100];
D2=[0 1];
L2=[1 0];

A3=[0 -1.9;1 -0.5];
B3=[-2 0;1 0];
C3=[-90 100];
D3=[0 1];
L3=[1 0];

A4=[0 -1.9;1 -0.5];
B4=[-2 0;1 0];
C4=[-110 100];
D4=[0 1];
L4=[1 0];

mu=0.074;

setlmis([])
V11=lmivar(2,[2 2]);
S1=lmivar(1,[2 1]);
S2=lmivar(2,[2 2]);
Afhat=lmivar(2,[2 2]);
Bfhat=lmivar(2,[2 1]);
Lfhat=lmivar(2,[1 2]);
P11hat=lmivar(1,[2 1]);
P31hat=lmivar(1,[2 1]);
P21hat=lmivar(1,[2 1]);
P12hat=lmivar(1,[2 1]);
P32hat=lmivar(1,[2 1]);
P22hat=lmivar(1,[2 1]);
P13hat=lmivar(1,[2 1]);
P33hat=lmivar(1,[2 1]);
P23hat=lmivar(1,[2 1]);
```

```
P14hat=lmivar(1,[2 1]);
P34hat=lmivar(1,[2 1]);
P24hat=lmivar(1,[2 1]);
gamma=lmivar(1,[1 0]);

lmiterm([1 1 1 V11],-1,1,'s');
lmiterm([1 1 2 -S1],-1,1);
lmiterm([1 1 2 -S2],-1,1);
lmiterm([1 1 3 -V11],1,A1);
lmiterm([1 1 3 Bfhat],1,C1);
lmiterm([1 1 3 P11hat],1,1);
lmiterm([1 1 4 Afhat],1,1);
lmiterm([1 1 4 -P31hat],1,1);
lmiterm([1 1 5 -V11],1,B1);
lmiterm([1 1 5 Bfhat],1,D1);
lmiterm([1 1 7 -V11],1,1);
lmiterm([1 1 8 -S1],1,1);
lmiterm([1 2 2 S1],-1,1,'s');
lmiterm([1 2 3 S2],1,A1);
lmiterm([1 2 3 Bfhat],1,C1);
lmiterm([1 2 3 P31hat],1,1);
lmiterm([1 2 4 Afhat],1,1);
lmiterm([1 2 4 P21hat],1,1);
lmiterm([1 2 5 S2],1,B1);
lmiterm([1 2 5 Bfhat],1,D1);
lmiterm([1 2 7 S2],1,1);
lmiterm([1 2 8 -S1],1,1);
lmiterm([1 3 3 P11hat],-1,mu);
lmiterm([1 3 4 -P31hat],-mu,1);
lmiterm([1 3 6 0],L1');
lmiterm([1 4 4 P21hat],-1,mu);
lmiterm([1 4 6 -Lfhat],-1,1);
lmiterm([1 5 5 gamma],-1,1);
lmiterm([1 6 6 gamma],-1,1);
lmiterm([1 7 7 P11hat],inv(mu),-1);
lmiterm([1 7 8 -P31hat],inv(mu),-1);
lmiterm([1 8 8 P21hat],inv(mu),-1);

lmiterm([2 1 1 V11],-1,1,'s');
lmiterm([2 1 2 -S1],-1,1);
lmiterm([2 1 2 -S2],-1,1);
lmiterm([2 1 3 -V11],1,A2);
lmiterm([2 1 3 Bfhat],1,C2);
lmiterm([2 1 3 P12hat],1,1);
lmiterm([2 1 4 Afhat],1,1);
lmiterm([2 1 4 -P32hat],1,1);
lmiterm([2 1 5 -V11],1,B2);
lmiterm([2 1 5 Bfhat],1,D2);
lmiterm([2 1 7 -V11],1,1);
lmiterm([2 1 8 -S1],1,1);
lmiterm([2 2 2 S1],-1,1,'s');
lmiterm([2 2 3 S2],1,A2);
lmiterm([2 2 3 Bfhat],1,C2)
lmiterm([2 2 3 P32hat],1,1)
lmiterm([2 2 4 Afhat],1,1)
lmiterm([2 2 4 P22hat],1,1)
```

```
lmiterm([2 2 5 S2],1,B2)
lmiterm([2 2 5 Bfhat],1,D2);
lmiterm([2 2 7 S2],1,1);
lmiterm([2 2 8 -S1],1,1);
lmiterm([2 3 3 P12hat],-1,mu);
lmiterm([2 3 4 -P32hat],-mu,1);
lmiterm([2 3 6 0],L1');
lmiterm([2 4 4 P22hat],-1,mu);
lmiterm([2 4 6 -Lfhat],-1,1);
lmiterm([2 5 5 gamma],-1,1);
lmiterm([2 6 6 gamma],-1,1);
lmiterm([2 7 7 P12hat],inv(mu),-1);
lmiterm([2 7 8 -P32hat],inv(mu),-1);
lmiterm([2 8 8 P22hat],inv(mu),-1);

lmiterm([3 1 1 V11],-1,1,'s');
lmiterm([3 1 2 -S1],-1,1);
lmiterm([3 1 2 -S2],-1,1);
lmiterm([3 1 3 -V11],1,A3);
lmiterm([3 1 3 Bfhat],1,C3);
lmiterm([3 1 3 P13hat],1,1);
lmiterm([3 1 4 Afhat],1,1);
lmiterm([3 1 4 -P33hat],1,1);
lmiterm([3 1 5 -V11],1,B3);
lmiterm([3 1 5 Bfhat],1,D3);
lmiterm([3 1 7 -V11],1,1);
lmiterm([3 1 8 -S1],1,1);
lmiterm([3 2 2 S1],-1,1,'s');
lmiterm([3 2 3 S2],1,A3);
lmiterm([3 2 3 Bfhat],1,C3);
lmiterm([3 2 3 P33hat],1,1);
lmiterm([3 2 4 Afhat],1,1);
lmiterm([3 2 4 P23hat],1,1);
lmiterm([3 2 5 S2],1,B3);
lmiterm([3 2 5 Bfhat],1,D3);
lmiterm([3 2 7 S2],1,1);
lmiterm([3 2 8 -S1],1,1);
lmiterm([3 3 3 P13hat],-1,mu);
lmiterm([3 3 4 -P33hat],-mu,1);
lmiterm([3 3 6 0],L1');
lmiterm([3 4 4 P23hat],-1,mu);
lmiterm([3 4 6 -Lfhat],-1,1);
lmiterm([3 5 5 gamma],-1,1);
lmiterm([3 6 6 gamma],-1,1);
lmiterm([3 7 7 P13hat],inv(mu),-1);
lmiterm([3 7 8 -P33hat],inv(mu),-1);
lmiterm([3 8 8 P23hat],inv(mu),-1);

lmiterm([4 1 1 V11],-1,1,'s');
lmiterm([4 1 2 -S1],-1,1);
lmiterm([4 1 2 -S2],-1,1);
lmiterm([4 1 3 -V11],1,A4);
lmiterm([4 1 3 Bfhat],1,C4);
lmiterm([4 1 3 P14hat],1,1);
lmiterm([4 1 4 Afhat],1,1);
lmiterm([4 1 4 -P34hat],1,1);
```

```
lmiterm([4 1 5 -V11],1,B4);
lmiterm([4 1 5 Bfhat],1,D4);
lmiterm([4 1 7 -V11],1,1);
lmiterm([4 1 8 -S1],1,1);
lmiterm([4 2 2 S1],-1,1,'s');
lmiterm([4 2 3 S2],1,A4);
lmiterm([4 2 3 Bfhat],1,C4);
lmiterm([4 2 3 P34hat],1,1);
lmiterm([4 2 4 Afhat],1,1);
lmiterm([4 2 4 P24hat],1,1);
lmiterm([4 2 5 S2],1,B4);
lmiterm([4 2 5 Bfhat],1,D4);
lmiterm([4 2 7 S2],1,1);
lmiterm([4 2 8 -S1],1,1);
lmiterm([4 3 3 P14hat],-1,mu);
lmiterm([4 3 4 -P34hat],-mu,1);
lmiterm([4 3 6 0],L1');
lmiterm([4 4 4 P24hat],-1,mu);
lmiterm([4 4 6 -Lfhat],-1,1);
lmiterm([4 5 5 gamma],-1,1);
lmiterm([4 6 6 gamma],-1,1);
lmiterm([4 7 7 P14hat],inv(mu),-1);
lmiterm([4 7 8 -P34hat],inv(mu),-1);
lmiterm([4 8 8 P24hat],inv(mu),-1);

lmiterm([-5 1 1 P11hat],1,1);
lmiterm([-5 1 2 -P31hat],1,1);
lmiterm([-5 2 2 P21hat],1,1);
lmiterm([-6 1 1 P12hat],1,1);
lmiterm([-6 1 2 -P32hat],1,1);
lmiterm([-6 2 2 P22hat],1,1);
lmiterm([-7 1 1 P13hat],1,1);
lmiterm([-7 1 2 -P33hat],1,1);
lmiterm([-7 2 2 P23hat],1,1);
lmiterm([-8 1 1 P14hat],1,1);
lmiterm([-8 1 2 -P34hat],1,1);
lmiterm([-8 2 2 P24hat],1,1);

lmiterm([-9 1 1 gamma],1,1);

LMISYS=getlmis
[tmin,xfeas]=feasp(LMISYS);

nx=decnbr(LMISYS);
c=zeros(nx,1);
for i=1:nx
    [gammaj]=defcx(LMISYS,i,gamma);
    c(i)=gammaj
end

[copt,xopt]=mincx(LMISYS,c);

Afhatm=dec2mat(LMISYS,xopt,Afhat);
Bfhatm=dec2mat(LMISYS,xopt,Bfhat);
Lfhatm=dec2mat(LMISYS,xopt,Lfhat);
S1m=dec2mat(LMISYS,xopt,S1);
```

```
[U,T]=schur(S1m);
V21T=U
V21=U'
V22inv=T'
Af=(inv(V21))'*Afhatm*inv(V21)*inv(V22inv)
Bf=(inv(V21))'*Bfhatm
Lf=Lfhatm*inv(V21)*inv(V22inv)

P11hatm=dec2mat(LMISYS,xopt,P11hat);
P31hatm=dec2mat(LMISYS,xopt,P31hat);
P21hatm=dec2mat(LMISYS,xopt,P21hat);
P12hatm=dec2mat(LMISYS,xopt,P12hat);
P32hatm=dec2mat(LMISYS,xopt,P32hat);
P22hatm=dec2mat(LMISYS,xopt,P22hat);
P13hatm=dec2mat(LMISYS,xopt,P13hat);
P33hatm=dec2mat(LMISYS,xopt,P33hat);
P23hatm=dec2mat(LMISYS,xopt,P23hat);
P14hatm=dec2mat(LMISYS,xopt,P14hat);
P34hatm=dec2mat(LMISYS,xopt,P34hat);
P24hatm=dec2mat(LMISYS,xopt,P24hat);

P1hatm_BRL=[P11hatm,P31hatm';P31hatm,P21hatm]
eig_11=eig(P1hatm_BRL)

P2hatm_BRL=[P12hatm,P32hatm';P32hatm,P22hatm];
eig_22=eig(P2hatm_BRL)

P3hatm_BRL=[P13hatm,P33hatm';P33hatm,P23hatm];
eig_33=eig(P3hatm_BRL)

P4hatm_BRL=[P14hatm,P34hatm';P34hatm,P24hatm];
eig_44=eig(P4hatm_BRL)

zero=[0,0;0,0];

Acl1=[A1,zero;Bf*C1,Af]
Bcl1=[B1;Bf*D1]
Lcl1=[L1,-Lf];

eig_1=eig(Acl1)

Acl2=[A2,zero;Bf*C2,Af];
Bcl2=[B2;Bf*D2];
Lcl2=[L2,-Lf];

eig_2=eig(Acl2)

Acl3=[A3,zero;Bf*C3,Af];
Bcl3=[B3;Bf*D3];
Lcl3=[L3,-Lf];

eig_3=eig(Acl3)
```

```
Acl4=[A4,zero;Bf*C4,Af];
Bcl4=[B4;Bf*D4];
Lcl4=[L4,-Lf];

eig_4=eig(Acl4)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                Analysis Part
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

setlmis([]);

gamma=lmivar(1,[1 0]);
P1=lmivar(1,[4 1]);
P2=lmivar(1,[4 1]);
P3=lmivar(1,[4 1]);
P4=lmivar(1,[4 1]);

BRL1=newlmi;
lmiterm([BRL1 1 1 P1],1,Acl1,'s');   % XA+A'X
lmiterm([BRL1 1 2 P1],1,Bcl1);       % BX
lmiterm([BRL1 1 3 0],Lcl1');        % C'
lmiterm([BRL1 2 2 gamma],-1,1);     % -muI
lmiterm([BRL1 3 3 gamma],-1,1);     % -muI

BRL1_P1_POS=newlmi;
lmiterm([-BRL1_P1_POS 1 1 P1],1,1);

BRL2=newlmi;
lmiterm([BRL2 1 1 P2],1,Acl2,'s');   % XA+A'X
lmiterm([BRL2 1 2 P2],1,Bcl2);       % BX
lmiterm([BRL2 1 3 0],Lcl2');        % C'
lmiterm([BRL2 2 2 gamma],-1,1);     % -muI
lmiterm([BRL2 3 3 gamma],-1,1);     % -muI

BRL2_P2_POS=newlmi;
lmiterm([-BRL2_P2_POS 1 1 P2],1,1);

BRL3=newlmi;
lmiterm([BRL3 1 1 P3],1,Acl3,'s');   % XA+A'X
lmiterm([BRL3 1 2 P3],1,Bcl3);       % BX
lmiterm([BRL3 1 3 0],Lcl3');        % C'
lmiterm([BRL3 2 2 gamma],-1,1);     % -muI
lmiterm([BRL3 3 3 gamma],-1,1);     % -muI

BRL3_P3_POS=newlmi;
lmiterm([-BRL3_P3_POS 1 1 P3],1,1);

BRL4=newlmi;
lmiterm([BRL4 1 1 P4],1,Acl4,'s');   % XA+A'X
lmiterm([BRL4 1 2 P4],1,Bcl4);       % BX
lmiterm([BRL4 1 3 0],Lcl4');        % C'
lmiterm([BRL4 2 2 gamma],-1,1);     % -muI
```

```
lmiterm([BRL4 3 3 gamma],-1,1);     % -muI

BRL4_P4_POS=newlmi;
lmiterm([-BRL4_P4_POS 1 1 P4],1,1);

LMISYS_BRL=getlmis
% Defining vector "c"
n = decnbr(LMISYS_BRL);
c = zeros(n,1);
for i=1:n
[gammaj] = defcx(LMISYS_BRL, i,gamma);
c(i) = gammaj;
end

[copt, xopt] = mincx (LMISYS_BRL, c, [1e-5 0 0 0 0] )
GAMMA=copt
```

APPENDIX B

MATLAB CODE TO SYNTHESIZE THE REDUCED ORDER FILTER FOR TABLE 3

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                    Synthesis Part
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

A1=[0 -0.1;1 -0.5];
B1=[-2 0;1 0];
C1=[-90 100];
D1=[0 1];
L1=[1 0];

A2=[0 -0.1;1 -0.5];
B2=[-2 0;1 0];
C2=[-110 100];
D2=[0 1];
L2=[1 0];

A3=[0 -1.9;1 -0.5];
B3=[-2 0;1 0];
C3=[-90 100];
D3=[0 1];
L3=[1 0];

A4=[0 -1.9;1 -0.5];
B4=[-2 0;1 0];
C4=[-110 100];
D4=[0 1];
L4=[1 0];

mu=0.066;

setlmis([])
V11=lmivar(2,[2 2])
[S1,n,sS1]=lmivar(1,[1 0])
[S1t,n,sS1t]=lmivar(3,[sS1,zeros(1)])
S2=lmivar(2,[1 2])
[Afhat,n,sAfhat]=lmivar(1,[1 0])
[Afhatt,n,sAfhatt]=lmivar(3,[sAfhat;zeros(1)])
[Bfhat,n,sBfhat]=lmivar(1,[1 0])
[Bfhatt,n,sBfhatt]=lmivar(3,[sBfhat;zeros(1)])
Lfhat=lmivar(1,[1 0])
P11hat=lmivar(1,[2 1])
P31hat=lmivar(2,[1 2])
P21hat=lmivar(1,[1 1])
P12hat=lmivar(1,[2 1])
P32hat=lmivar(2,[1 2])
```

```
P22hat=lmivar(1,[1 1])
P13hat=lmivar(1,[2 1])
P33hat=lmivar(2,[1 2])
P23hat=lmivar(1,[1 1])
P14hat=lmivar(1,[2 1])
P34hat=lmivar(2,[1 2])
P24hat=lmivar(1,[1 1])

gamma=lmivar(1,[1 0])

lmiterm([1 1 1 V11],-1,1,'s')
lmiterm([1 1 2 -S1t],-1,1)
lmiterm([1 1 2 -S2],-1,1)
lmiterm([1 1 3 -V11],1,A1)
lmiterm([1 1 3 Bfhatt],1,C1)
lmiterm([1 1 3 P11hat],1,1)
lmiterm([1 1 4 Afhatt],1,1)
lmiterm([1 1 4 -P31hat],1,1)
lmiterm([1 1 5 -V11],1,B1)
lmiterm([1 1 5 Bfhatt],1,D1)
lmiterm([1 1 7 -V11],1,1)
lmiterm([1 1 8 -S1t],1,1)
lmiterm([1 2 2 S1],-1,1,'s')
lmiterm([1 2 3 S2],1,A1)
lmiterm([1 2 3 Bfhat],1,C1)
lmiterm([1 2 3 P31hat],1,1)
lmiterm([1 2 4 Afhat],1,1)
lmiterm([1 2 4 P21hat],1,1)
lmiterm([1 2 5 S2],1,B1)
lmiterm([1 2 5 Bfhat],1,D1)
lmiterm([1 2 7 S2],1,1)
lmiterm([1 2 8 -S1],1,1)
lmiterm([1 3 3 P11hat],-1,mu)
lmiterm([1 3 4 -P31hat],-mu,1)
lmiterm([1 3 6 0],L1')
lmiterm([1 4 4 P21hat],-1,mu)
lmiterm([1 4 6 -Lfhat],-1,1)
lmiterm([1 5 5 gamma],-1,1)
lmiterm([1 6 6 gamma],-1,1)
lmiterm([1 7 7 P11hat],inv(mu),-1)
lmiterm([1 7 8 -P31hat],inv(mu),-1)
lmiterm([1 8 8 P21hat],inv(mu),-1)

lmiterm([2 1 1 V11],-1,1,'s')
lmiterm([2 1 2 -S1t],-1,1)
lmiterm([2 1 2 -S2],-1,1)
lmiterm([2 1 3 -V11],1,A2)
lmiterm([2 1 3 Bfhatt],1,C2)
lmiterm([2 1 3 P12hat],1,1)
lmiterm([2 1 4 Afhatt],1,1)
lmiterm([2 1 4 -P32hat],1,1)
lmiterm([2 1 5 -V11],1,B2)
lmiterm([2 1 5 Bfhatt],1,D2)
lmiterm([2 1 7 -V11],1,1)
lmiterm([2 1 8 -S1t],1,1)
lmiterm([2 2 2 S1],-1,1,'s')
```

```
lmiterm([2 2 3 S2],1,A2)
lmiterm([2 2 3 Bfhat],1,C2)
lmiterm([2 2 3 P32hat],1,1)
lmiterm([2 2 4 Afhat],1,1)
lmiterm([2 2 4 P22hat],1,1)
lmiterm([2 2 5 S2],1,B2)
lmiterm([2 2 5 Bfhat],1,D2)
lmiterm([2 2 7 S2],1,1)
lmiterm([2 2 8 -S1],1,1)
lmiterm([2 3 3 P22hat],-1,mu)
lmiterm([2 3 4 -P32hat],-mu,1)
lmiterm([2 3 6 0],L1')
lmiterm([2 4 4 P22hat],-1,mu)
lmiterm([2 4 6 -Lfhat],-1,1)
lmiterm([2 5 5 gamma],-1,1)
lmiterm([2 6 6 gamma],-1,1)
lmiterm([2 7 7 P12hat],inv(mu),-1)
lmiterm([2 7 8 -P32hat],inv(mu),-1)
lmiterm([2 8 8 P22hat],inv(mu),-1)

lmiterm([3 1 1 V11],-1,1,'s')
lmiterm([3 1 2 -S1t],-1,1)
lmiterm([3 1 2 -S2],-1,1)
lmiterm([3 1 3 -V11],1,A3)
lmiterm([3 1 3 Bfhatt],1,C3)
lmiterm([3 1 3 P13hat],1,1)
lmiterm([3 1 4 Afhatt],1,1)
lmiterm([3 1 4 -P33hat],1,1)
lmiterm([3 1 5 -V11],1,B3)
lmiterm([3 1 5 Bfhatt],1,D3)
lmiterm([3 1 7 -V11],1,1)
lmiterm([3 1 8 -S1t],1,1)
lmiterm([3 2 2 S1],-1,1,'s')
lmiterm([3 2 3 S2],1,A3)
lmiterm([3 2 3 Bfhat],1,C3)
lmiterm([3 2 3 P33hat],1,1)
lmiterm([3 2 4 Afhat],1,1)
lmiterm([3 2 4 P23hat],1,1)
lmiterm([3 2 5 S2],1,B3)
lmiterm([3 2 5 Bfhat],1,D3)
lmiterm([3 2 7 S2],1,1)
lmiterm([3 2 8 -S1],1,1)
lmiterm([3 3 3 P13hat],-1,mu)
lmiterm([3 3 4 -P33hat],-mu,1)
lmiterm([3 3 6 0],L1')
lmiterm([3 4 4 P23hat],-1,mu)
lmiterm([3 4 6 -Lfhat],-1,1)
lmiterm([3 5 5 gamma],-1,1)
lmiterm([3 6 6 gamma],-1,1)
lmiterm([3 7 7 P13hat],inv(mu),-1)
lmiterm([3 7 8 -P33hat],inv(mu),-1)
lmiterm([3 8 8 P23hat],inv(mu),-1)

lmiterm([4 1 1 V11],-1,1,'s')
lmiterm([4 1 2 -S1t],-1,1)
lmiterm([4 1 2 -S2],-1,1)
```

```
lmiterm([4 1 3 -V11],1,A4)
lmiterm([4 1 3 Bfhatt],1,C4)
lmiterm([4 1 3 P14hat],1,1)
lmiterm([4 1 4 Afhatt],1,1)
lmiterm([4 1 4 -P34hat],1,1)
lmiterm([4 1 5 -V11],1,B4)
lmiterm([4 1 5 Bfhatt],1,D4)
lmiterm([4 1 7 -V11],1,1)
lmiterm([4 1 8 -S1t],1,1)
lmiterm([4 2 2 S1],-1,1,'s')
lmiterm([4 2 3 S2],1,A4)
lmiterm([4 2 3 Bfhat],1,C4)
lmiterm([4 2 3 P34hat],1,1)
lmiterm([4 2 4 Afhat],1,1)
lmiterm([4 2 4 P24hat],1,1)
lmiterm([4 2 5 S2],1,B4)
lmiterm([4 2 5 Bfhat],1,D4)
lmiterm([4 2 7 S2],1,1)
lmiterm([4 2 8 -S1],1,1)
lmiterm([4 3 3 P14hat],-1,mu)
lmiterm([4 3 4 -P34hat],-mu,1)
lmiterm([4 3 6 0],L1')
lmiterm([4 4 4 P24hat],-1,mu)
lmiterm([4 4 6 -Lfhat],-1,1)
lmiterm([4 5 5 gamma],-1,1)
lmiterm([4 6 6 gamma],-1,1)
lmiterm([4 7 7 P14hat],inv(mu),-1)
lmiterm([4 7 8 -P34hat],inv(mu),-1)
lmiterm([4 8 8 P24hat],inv(mu),-1)

lmiterm([-5 1 1 P11hat],1,1);
lmiterm([-5 1 2 -P31hat],1,1);
lmiterm([-5 2 2 P21hat],1,1);
lmiterm([-6 1 1 P12hat],1,1);
lmiterm([-6 1 2 -P32hat],1,1);
lmiterm([-6 2 2 P22hat],1,1);
lmiterm([-7 1 1 P13hat],1,1);
lmiterm([-7 1 2 -P33hat],1,1);
lmiterm([-7 2 2 P23hat],1,1);
lmiterm([-8 1 1 P14hat],1,1);
lmiterm([-8 1 2 -P34hat],1,1);
lmiterm([-8 2 2 P24hat],1,1);

lmiterm([-9 1 1 gamma],1,1);

LMISYS=getlmis
[tmin,xfeas]=feasp(LMISYS);

nx=decnbr(LMISYS);
c=zeros(nx,1);
for i=1:nx
    [gammaj]=defcx(LMISYS,i,gamma)
    c(i)=gammaj
end

[copt,xopt]=mincx(LMISYS,c);
```

```
Afhatm=dec2mat(LMISYS,xopt,Afhat);
Bfhatm=dec2mat(LMISYS,xopt,Bfhat);
Lfhatm=dec2mat(LMISYS,xopt,Lfhat);
S1m=dec2mat(LMISYS,xopt,S1);

[U,T]=schur(S1m);
V21tT=U
V21t=U'
V22inv=T'
Af=(inv(V21t))'*Afhatm*inv(V21t)*inv(V22inv)
Bf=(inv(V21t))'*Bfhatm
Lf=Lfhatm*inv(V21t)*inv(V22inv)

P11hatm=dec2mat(LMISYS,xopt,P11hat);
P31hatm=dec2mat(LMISYS,xopt,P31hat);
P21hatm=dec2mat(LMISYS,xopt,P21hat);
P12hatm=dec2mat(LMISYS,xopt,P12hat);
P32hatm=dec2mat(LMISYS,xopt,P32hat);
P22hatm=dec2mat(LMISYS,xopt,P22hat);
P13hatm=dec2mat(LMISYS,xopt,P13hat);
P33hatm=dec2mat(LMISYS,xopt,P33hat);
P23hatm=dec2mat(LMISYS,xopt,P23hat);
P14hatm=dec2mat(LMISYS,xopt,P14hat);
P34hatm=dec2mat(LMISYS,xopt,P34hat);
P24hatm=dec2mat(LMISYS,xopt,P24hat);

P1hatm_BRL=[P11hatm,P31hatm';P31hatm,P21hatm]
eig_11=eig(P1hatm_BRL)

P2hatm_BRL=[P12hatm,P32hatm';P32hatm,P22hatm];
eig_22=eig(P2hatm_BRL)

P3hatm_BRL=[P13hatm,P33hatm';P33hatm,P23hatm];
eig_33=eig(P3hatm_BRL)

P4hatm_BRL=[P14hatm,P34hatm';P34hatm,P24hatm];
eig_44=eig(P4hatm_BRL)

zero=[0;0];

Acl1=[A1,zero;Bf*C1,Af];
Bcl1=[B1;Bf*D1];
Lcl1=[L1,-Lf];

eig_1=eig(Acl1)

Acl2=[A2,zero;Bf*C2,Af];
Bcl2=[B2;Bf*D2];
Lcl2=[L2,-Lf];

eig_2=eig(Acl2)

Acl3=[A3,zero;Bf*C3,Af];
```

```
Bcl3=[B3;Bf*D3];
Lcl3=[L3,-Lf];

eig_3=eig(Acl3)

Acl4=[A4,zero;Bf*C4,Af];
Bcl4=[B4;Bf*D4];
Lcl4=[L4,-Lf];

eig_4=eig(Acl4)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                Analysis Part
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

setlmis([]);

gamma=lmivar(1,[1 0]);
P1=lmivar(1,[3 1]);
P2=lmivar(1,[3 1]);
P3=lmivar(1,[3 1]);
P4=lmivar(1,[3 1]);

BRL1=newlmi;
lmiterm([BRL1 1 1 P1],1,Acl1,'s');   % XA+A'X
lmiterm([BRL1 1 2 P1],1,Bcl1);       % BX
lmiterm([BRL1 1 3 0],Lcl1');        % C'
lmiterm([BRL1 2 2 gamma],-1,1);     % -muI
lmiterm([BRL1 3 3 gamma],-1,1);     % -muI

BRL1_P1_POS=newlmi;
lmiterm([-BRL1_P1_POS 1 1 P1],1,1);

BRL2=newlmi;
lmiterm([BRL2 1 1 P2],1,Acl2,'s');   % XA+A'X
lmiterm([BRL2 1 2 P2],1,Bcl2);       % BX
lmiterm([BRL2 1 3 0],Lcl2');        % C'
lmiterm([BRL2 2 2 gamma],-1,1);     % -muI
lmiterm([BRL2 3 3 gamma],-1,1);     % -muI

BRL2_P2_POS=newlmi;
lmiterm([-BRL2_P2_POS 1 1 P2],1,1);

BRL3=newlmi;
lmiterm([BRL3 1 1 P3],1,Acl3,'s');   % XA+A'X
lmiterm([BRL3 1 2 P3],1,Bcl3);       % BX
lmiterm([BRL3 1 3 0],Lcl3');        % C'
lmiterm([BRL3 2 2 gamma],-1,1);     % -muI
lmiterm([BRL3 3 3 gamma],-1,1);     % -muI

BRL3_P3_POS=newlmi;
lmiterm([-BRL3_P3_POS 1 1 P3],1,1);
```

```
BRL4=newlmi;
lmiterm([BRL4 1 1 P4],1,Acl4,'s');   % XA+A'X
lmiterm([BRL4 1 2 P4],1,Bcl4);       % BX
lmiterm([BRL4 1 3 0],Lcl4');        % C'
lmiterm([BRL4 2 2 gamma],-1,1);     % -muI
lmiterm([BRL4 3 3 gamma],-1,1);     % -muI

BRL4_P4_POS=newlmi;
lmiterm([-BRL4_P4_POS 1 1 P4],1,1);

LMISYS_BRL=getlmis
% Defining vector "c"
n = decnbr(LMISYS_BRL);
c = zeros(n,1);
for i=1:n
[gammaj] = defcx(LMISYS_BRL, i,gamma);
c(i) = gammaj;
end

[copt, xopt] = mincx (LMISYS_BRL, c, [1e-5 0 0 0 0] )
GAMMA=copt
```

APPENDIX C

MATLAB CODE TO DESIGN THE 6th ORDER RFDF FOR 4.4.2 EXAMPLE 2

```
A1=[-0.5, -2, 0, 0, 0, 0;
    0.5, 0, 0, 0, 0, 0;
    0, 0, -10.5, -1.5, -1.25, 0;
    0, 0, 4, 0, 0, 0;
    0, 0, 0, 2, 0, 0;
    0, 0, 0, 0, 0, -5];

A2=[-0.5, -3, 0, 0, 0, 0;
    0.5, 0, 0, 0, 0, 0;
    0, 0, -10.5, -1.625, -1.875, 0;
    0, 0, 4, 0, 0, 0;
    0, 0, 0, 2, 0, 0;
    0, 0, 0, 0, 0, -5];

A3=[-1.2, -2, 0, 0, 0, 0;
    0.5, 0, 0, 0, 0, 0;
    0, 0, -11.2, -3.25, -1.25, 0;
    0, 0, 4, 0, 0, 0;
    0, 0, 0, 2, 0, 0;
    0, 0, 0, 0, 0, -5];

A4=[-1.2, -3, 0, 0, 0, 0;
    0.5, 0, 0, 0, 0, 0;
    0, 0, -11.2, -3.375, -1.875, 0;
    0, 0, 4, 0, 0, 0;
    0, 0, 0, 2, 0, 0;
    0, 0, 0, 0, 0, -5];

Bd=[0, 0;
    0, 0;
    2, 0;
    0, 0;
    0, 0;
    0, 8];

Bf=[4;0;0;0;0;0];

Bu=[4;0;0;0;0;0];

B=[Bu, Bf, Bd];

C1=[2.5, 0.5, 0, 3.75, 0, 0;
    0, 0, 0, 0, 1.875, -6.125];

C2=[2.5, 0.5, 0, 3.75, 0, 0;
    0, 0, 0, 0, 1.875, -6.125];
```

```
C3=[2.5, 0.5, 0, 3.75, 0, 0;
    0, 0, 0, 0, 1.875, -6.125];

C4=[2.5, 0.5, 0, 3.75, 0, 0;
    0, 0, 0, 0, 1.875, -6.125];

Dd=[0, 0;
    0, 10];

Df=zeros(2,1);
Du=zeros(2,1);

D=[Du, Df, Dd];

W=tf({1;1}, {[1 2 1];[1 2 1]});
[Aw,Bw,Cw,Dw]=ssdata(W)

At1=[A1,zeros(6,2);zeros(2,6),Aw]
At2=[A2,zeros(6,2);zeros(2,6),Aw]
At3=[A3,zeros(6,2);zeros(2,6),Aw]
At4=[A4,zeros(6,2);zeros(2,6),Aw]

Bft=[Bf;Bw]
But=[Bu;zeros(2,1)]
Bdt=[Bd;zeros(2)]

C3t=[C1;zeros(1,6)]
C1t=[-C3t,zeros(3,2)]
C2t=[C1,Cw]

HFt_f=[Df;zeros(1)]
HFt_d=[Dd;zeros(1,2)]
HFt_u=[Du;1]
C4t=[C3t,zeros(3,2)]

mu=1.2
dd=1;       %gamma_d
ff=1;       %gamma_f
uu=1        %gamma_u

setlmis([])
V11=lmivar(2,[8 8]);
[S1,n,sS1]=lmivar(1,[6 1]);
[S1t,n,sS1t]=lmivar(3,[sS1,zeros(6,2)]);
S2=lmivar(2,[6 8]);
[AFhat,n,sAFhat]=lmivar(2,[6 6]);
[AFhatt,n,sAFhatt]=lmivar(3,[sAFhat;zeros(2,6)]);
[BFhat,n,sBFhat]=lmivar(2,[6 3]);
[BFhatt,n,sBFhatt]=lmivar(3,[sBFhat;zeros(2,3)]);
LFhat=lmivar(2,[2 6])
HF=lmivar(2,[2 3]);
P11hat=lmivar(1,[8 1]);
P21hat=lmivar(1,[6 1]);
```

```
P31hat=lmivar(2,[6 8]);
P12hat=lmivar(1,[8 1]);
P22hat=lmivar(1,[6 1]);
P32hat=lmivar(2,[6 8]);
P13hat=lmivar(1,[8 1]);
P23hat=lmivar(1,[6 1]);
P33hat=lmivar(2,[6 8]);
P14hat=lmivar(1,[8 1]);
P24hat=lmivar(1,[6 1]);
P34hat=lmivar(2,[6 8]);

gamma_d=lmivar(1,[1 0])
gamma_f=lmivar(1,[1 0])
gamma_u=lmivar(1,[1 0])

lmiterm([1 1 1 V11],-1,1,'s')
lmiterm([1 1 2 -S1t],-1,1)
lmiterm([1 1 2 -S2],-1,1)
lmiterm([1 1 3 -V11],1,At1)
lmiterm([1 1 3 BFhatt],1,C4t)
lmiterm([1 1 3 P11hat],1,1)
lmiterm([1 1 4 AFhatt],1,1)
lmiterm([1 1 4 -P31hat],1,1)
lmiterm([1 1 5 -V11],1,Bdt)
lmiterm([1 1 5 BFhatt],1,HFt_d)
lmiterm([1 1 7 -V11],1,1)
lmiterm([1 1 8 -S1t],1,1)
lmiterm([1 2 2 S1],-1,1,'s')
lmiterm([1 2 3 S2],1,At1)
lmiterm([1 2 3 BFhat],1,C4t)
lmiterm([1 2 3 P31hat],1,1)
lmiterm([1 2 4 AFhat],1,1)
lmiterm([1 2 4 P21hat],1,1)
lmiterm([1 2 5 S2],1,Bdt)
lmiterm([1 2 5 BFhat],1,HFt_d)
lmiterm([1 2 7 S2],1,1)
lmiterm([1 2 8 -S1],1,1)
lmiterm([1 3 3 P11hat],-mu,1)
lmiterm([1 3 4 -P31hat],-mu,1)
lmiterm([1 3 6 -HF],C1t',-1)
lmiterm([1 3 6 0],-C2t')
lmiterm([1 4 4 P21hat],-mu,1)
lmiterm([1 4 6 -LFhat],1,1)
lmiterm([1 5 5 gamma_d],-eye(2),1)
lmiterm([1 5 6 -HF],HFt_d',1)
lmiterm([1 5 6 0],-Dd)
lmiterm([1 6 6 gamma_d],-1,1)
lmiterm([1 7 7 P11hat],-inv(mu),1)
lmiterm([1 7 8 -P31hat],-inv(mu),1)
lmiterm([1 8 8 P21hat],-inv(mu),1)

lmiterm([2 1 1 V11],-1,1,'s')
lmiterm([2 1 2 -S1t],-1,1)
lmiterm([2 1 2 -S2],-1,1)
lmiterm([2 1 3 -V11],1,At2)
lmiterm([2 1 3 BFhatt],1,C4t)
```

```
lmiterm([2 1 3 P12hat],1,1)
lmiterm([2 1 4 AFhatt],1,1)
lmiterm([2 1 4 -P32hat],1,1)
lmiterm([2 1 5 -V11],1,Bdt)
lmiterm([2 1 5 BFhatt],1,HFt_d)
lmiterm([2 1 7 -V11],1,1)
lmiterm([2 1 8 -S1t],1,1)
lmiterm([2 2 2 S1],-1,1,'s')
lmiterm([2 2 3 S2],1,At2)
lmiterm([2 2 3 BFhat],1,C4t)
lmiterm([2 2 3 P32hat],1,1)
lmiterm([2 2 4 AFhat],1,1)
lmiterm([2 2 4 P22hat],1,1)
lmiterm([2 2 5 S2],1,Bdt)
lmiterm([2 2 5 BFhat],1,HFt_d)
lmiterm([2 2 7 S2],1,1)
lmiterm([2 2 8 -S1],1,1)
lmiterm([2 3 3 P12hat],-mu,1)
lmiterm([2 3 4 -P32hat],-mu,1)
lmiterm([2 3 6 -HF],C1t',-1)
lmiterm([2 3 6 0],-C2t')
lmiterm([2 4 4 P22hat],-mu,1)
lmiterm([2 4 6 -LFhat],1,1)
lmiterm([2 5 5 gamma_d],-eye(2),1)
lmiterm([2 5 6 -HF],HFt_d',1)
lmiterm([2 5 6 0],-Dd)
lmiterm([2 6 6 gamma_d],-1,1)
lmiterm([2 7 7 P12hat],-inv(mu),1)
lmiterm([2 7 8 -P32hat],-inv(mu),1)
lmiterm([2 8 8 P22hat],-inv(mu),1)

lmiterm([3 1 1 V11],-1,1,'s')
lmiterm([3 1 2 -S1t],-1,1)
lmiterm([3 1 2 -S2],-1,1)
lmiterm([3 1 3 -V11],1,At3)
lmiterm([3 1 3 BFhatt],1,C4t)
lmiterm([3 1 3 P13hat],1,1)
lmiterm([3 1 4 AFhatt],1,1)
lmiterm([3 1 4 -P33hat],1,1)
lmiterm([3 1 5 -V11],1,Bdt)
lmiterm([3 1 5 BFhatt],1,HFt_d)
lmiterm([3 1 7 -V11],1,1)
lmiterm([3 1 8 -S1t],1,1)
lmiterm([3 2 2 S1],-1,1,'s')
lmiterm([3 2 3 S2],1,At3)
lmiterm([3 2 3 BFhat],1,C4t)
lmiterm([3 2 3 P33hat],1,1)
lmiterm([3 2 4 AFhat],1,1)
lmiterm([3 2 4 P23hat],1,1)
lmiterm([3 2 5 S2],1,Bdt)
lmiterm([3 2 5 BFhat],1,HFt_d)
lmiterm([3 2 7 S2],1,1)
lmiterm([3 2 8 -S1],1,1)
lmiterm([3 3 3 P13hat],-mu,1)
lmiterm([3 3 4 -P33hat],-mu,1)
lmiterm([3 3 6 -HF],C1t',-1)
lmiterm([3 3 6 0],-C2t')
```

```
lmiterm([3 4 4 P23hat],-mu,1)
lmiterm([3 4 6 -LFhat],1,1)
lmiterm([3 5 5 gamma_d],-eye(2),1)
lmiterm([3 5 6 -HF],HFt_d',1)
lmiterm([3 5 6 0],-Dd)
lmiterm([3 6 6 gamma_d],-1,1)
lmiterm([3 7 7 P13hat],-inv(mu),1)
lmiterm([3 7 8 -P33hat],-inv(mu),1)
lmiterm([3 8 8 P23hat],-inv(mu),1)

lmiterm([4 1 1 V11],-1,1,'s')
lmiterm([4 1 2 -S1t],-1,1)
lmiterm([4 1 2 -S2],-1,1)
lmiterm([4 1 3 -V11],1,At4)
lmiterm([4 1 3 BFhatt],1,C4t)
lmiterm([4 1 3 P14hat],1,1)
lmiterm([4 1 4 AFhatt],1,1)
lmiterm([4 1 4 -P34hat],1,1)
lmiterm([4 1 5 -V11],1,Bdt)
lmiterm([4 1 5 BFhatt],1,HFt_d)
lmiterm([4 1 7 -V11],1,1)
lmiterm([4 1 8 -S1t],1,1)
lmiterm([4 2 2 S1],-1,1,'s')
lmiterm([4 2 3 S2],1,At4)
lmiterm([4 2 3 BFhat],1,C4t)
lmiterm([4 2 3 P34hat],1,1)
lmiterm([4 2 4 AFhat],1,1)
lmiterm([4 2 4 P24hat],1,1)
lmiterm([4 2 5 S2],1,Bdt)
lmiterm([4 2 5 BFhat],1,HFt_d)
lmiterm([4 2 7 S2],1,1)
lmiterm([4 2 8 -S1],1,1)
lmiterm([4 3 3 P14hat],-mu,1)
lmiterm([4 3 4 -P34hat],-mu,1)
lmiterm([4 3 6 -HF],C1t',-1)
lmiterm([4 3 6 0],-C2t')
lmiterm([4 4 4 P24hat],-mu,1)
lmiterm([4 4 6 -LFhat],1,1)
lmiterm([4 5 5 gamma_d],-eye(2),1)
lmiterm([4 5 6 -HF],HFt_d',1)
lmiterm([4 5 6 0],-Dd)
lmiterm([4 6 6 gamma_d],-1,1)
lmiterm([4 7 7 P14hat],-inv(mu),1)
lmiterm([4 7 8 -P34hat],-inv(mu),1)
lmiterm([4 8 8 P24hat],-inv(mu),1)

lmiterm([5 1 1 V11],-1,1,'s')
lmiterm([5 1 2 -S1t],-1,1)
lmiterm([5 1 2 -S2],-1,1)
lmiterm([5 1 3 -V11],1,At1)
lmiterm([5 1 3 BFhatt],1,C4t)
lmiterm([5 1 3 P11hat],1,1)
lmiterm([5 1 4 AFhatt],1,1)
lmiterm([5 1 4 -P31hat],1,1)
lmiterm([5 1 5 -V11],1,Bft)
lmiterm([5 1 5 BFhatt],1,HFt_f)
```

```
lmiterm([5 1 7 -V11],1,1)
lmiterm([5 1 8 -S1t],1,1)
lmiterm([5 2 2 S1],-1,1,'s')
lmiterm([5 2 3 S2],1,At1)
lmiterm([5 2 3 BFhat],1,C4t)
lmiterm([5 2 3 P31hat],1,1)
lmiterm([5 2 4 AFhat],1,1)
lmiterm([5 2 4 P21hat],1,1)
lmiterm([5 2 5 S2],1,Bft)
lmiterm([5 2 5 BFhat],1,HFt_f)
lmiterm([5 2 7 S2],1,1)
lmiterm([5 2 8 -S1],1,1)
lmiterm([5 3 3 P11hat],-mu,1)
lmiterm([5 3 4 -P31hat],-mu,1)
lmiterm([5 3 6 -HF],-C1t',1)
lmiterm([5 3 6 0],C2t')
lmiterm([5 4 4 P21hat],-mu,1)
lmiterm([5 4 6 -LFhat],-1,1)
lmiterm([5 5 5 gamma_f],-1,1)
lmiterm([5 5 6 -HF],HFt_f',-1)
lmiterm([5 5 6 0],Df)
lmiterm([5 6 6 gamma_f],-1,1)
lmiterm([5 7 7 P11hat],-inv(mu),1)
lmiterm([5 7 8 -P31hat],-inv(mu),1)
lmiterm([5 8 8 P21hat],-inv(mu),1)

lmiterm([6 1 1 V11],-1,1,'s')
lmiterm([6 1 2 -S1t],-1,1)
lmiterm([6 1 2 -S2],-1,1)
lmiterm([6 1 3 -V11],1,At2)
lmiterm([6 1 3 BFhatt],1,C4t)
lmiterm([6 1 3 P12hat],1,1)
lmiterm([6 1 4 AFhatt],1,1)
lmiterm([6 1 4 -P32hat],1,1)
lmiterm([6 1 5 -V11],1,Bft)
lmiterm([6 1 5 BFhatt],1,HFt_f)
lmiterm([6 1 7 -V11],1,1)
lmiterm([6 1 8 -S1t],1,1)
lmiterm([6 2 2 S1],-1,1,'s')
lmiterm([6 2 3 S2],1,At2)
lmiterm([6 2 3 BFhat],1,C4t)
lmiterm([6 2 3 P32hat],1,1)
lmiterm([6 2 4 AFhat],1,1)
lmiterm([6 2 4 P22hat],1,1)
lmiterm([6 2 5 S2],1,Bft)
lmiterm([6 2 5 BFhat],1,HFt_f)
lmiterm([6 2 7 S2],1,1)
lmiterm([6 2 8 -S1],1,1)
lmiterm([6 3 3 P12hat],-mu,1)
lmiterm([6 3 4 -P32hat],-mu,1)
lmiterm([6 3 6 -HF],-C1t',1)
lmiterm([6 3 6 0],C2t')
lmiterm([6 4 4 P22hat],-mu,1)
lmiterm([6 4 6 -LFhat],-1,1)
lmiterm([6 5 5 gamma_f],-1,1)
lmiterm([6 5 6 -HF],HFt_f',-1)
lmiterm([6 5 6 0],Df)
```

```
lmiterm([6 6 6 gamma_f],-1,1)
lmiterm([6 7 7 P12hat],-inv(mu),1)
lmiterm([6 7 8 -P32hat],-inv(mu),1)
lmiterm([6 8 8 P22hat],-inv(mu),1)

lmiterm([7 1 1 V11],-1,1,'s')
lmiterm([7 1 2 -S1t],-1,1)
lmiterm([7 1 2 -S2],-1,1)
lmiterm([7 1 3 -V11],1,At3)
lmiterm([7 1 3 BFhatt],1,C4t)
lmiterm([7 1 3 P13hat],1,1)
lmiterm([7 1 4 AFhatt],1,1)
lmiterm([7 1 4 -P33hat],1,1)
lmiterm([7 1 5 -V11],1,Bft)
lmiterm([7 1 5 BFhatt],1,HFt_f)
lmiterm([7 1 7 -V11],1,1)
lmiterm([7 1 8 -S1t],1,1)
lmiterm([7 2 2 S1],-1,1,'s')
lmiterm([7 2 3 S2],1,At3)
lmiterm([7 2 3 BFhat],1,C4t)
lmiterm([7 2 3 P33hat],1,1)
lmiterm([7 2 4 AFhat],1,1)
lmiterm([7 2 4 P23hat],1,1)
lmiterm([7 2 5 S2],1,Bft)
lmiterm([7 2 5 BFhat],1,HFt_f)
lmiterm([7 2 7 S2],1,1)
lmiterm([7 2 8 -S1],1,1)
lmiterm([7 3 3 P13hat],-mu,1)
lmiterm([7 3 4 -P33hat],-mu,1)
lmiterm([7 3 6 -HF],-C1t',1)
lmiterm([7 3 6 0],C2t')
lmiterm([7 4 4 P23hat],-mu,1)
lmiterm([7 4 6 -LFhat],-1,1)
lmiterm([7 5 5 gamma_f],-1,1)
lmiterm([7 5 6 -HF],HFt_f',-1)
lmiterm([7 5 6 0],Df)
lmiterm([7 6 6 gamma_f],-1,1)
lmiterm([7 7 7 P13hat],-inv(mu),1)
lmiterm([7 7 8 -P33hat],-inv(mu),1)
lmiterm([7 8 8 P23hat],-inv(mu),1)

lmiterm([8 1 1 V11],-1,1,'s')
lmiterm([8 1 2 -S1t],-1,1)
lmiterm([8 1 2 -S2],-1,1)
lmiterm([8 1 3 -V11],1,At4)
lmiterm([8 1 3 BFhatt],1,C4t)
lmiterm([8 1 3 P14hat],1,1)
lmiterm([8 1 4 AFhatt],1,1)
lmiterm([8 1 4 -P34hat],1,1)
lmiterm([8 1 5 -V11],1,Bft)
lmiterm([8 1 5 BFhatt],1,HFt_f)
lmiterm([8 1 7 -V11],1,1)
lmiterm([8 1 8 -S1t],1,1)
lmiterm([8 2 2 S1],-1,1,'s')
lmiterm([8 2 3 S2],1,At4)
lmiterm([8 2 3 BFhat],1,C4t)
```

```
lmiterm([8 2 3 P34hat],1,1)
lmiterm([8 2 4 AFhat],1,1)
lmiterm([8 2 4 P24hat],1,1)
lmiterm([8 2 5 S2],1,Bft)
lmiterm([8 2 5 BFhat],1,HFt_f)
lmiterm([8 2 7 S2],1,1)
lmiterm([8 2 8 -S1],1,1)
lmiterm([8 3 3 P14hat],-mu,1)
lmiterm([8 3 4 -P34hat],-mu,1)
lmiterm([8 3 6 -HF],-C1t',1)
lmiterm([8 3 6 0],C2t')
lmiterm([8 4 4 P24hat],-mu,1)
lmiterm([8 4 6 -LFhat],-1,1)
lmiterm([8 5 5 gamma_f],-1,1)
lmiterm([8 5 6 -HF],HFt_f',-1)
lmiterm([8 5 6 0],Df)
lmiterm([8 6 6 gamma_f],-1,1)
lmiterm([8 7 7 P14hat],-inv(mu),1)
lmiterm([8 7 8 -P34hat],-inv(mu),1)
lmiterm([8 8 8 P24hat],-inv(mu),1)

lmiterm([9 1 1 V11],-1,1,'s')
lmiterm([9 1 2 -S1t],-1,1)
lmiterm([9 1 2 -S2],-1,1)
lmiterm([9 1 3 -V11],1,At1)
lmiterm([9 1 3 BFhatt],1,C4t)
lmiterm([9 1 3 P11hat],1,1)
lmiterm([9 1 4 AFhatt],1,1)
lmiterm([9 1 4 -P31hat],1,1)
lmiterm([9 1 5 -V11],1,But)
lmiterm([9 1 5 BFhatt],1,HFt_u)
lmiterm([9 1 7 -V11],1,1)
lmiterm([9 1 8 -S1t],1,1)
lmiterm([9 2 2 S1],-1,1,'s')
lmiterm([9 2 3 S2],1,At1)
lmiterm([9 2 3 BFhat],1,C4t)
lmiterm([9 2 3 P31hat],1,1)
lmiterm([9 2 4 AFhat],1,1)
lmiterm([9 2 4 P21hat],1,1)
lmiterm([9 2 5 S2],1,But)
lmiterm([9 2 5 BFhat],1,HFt_u)
lmiterm([9 2 7 S2],1,1)
lmiterm([9 2 8 -S1],1,1)
lmiterm([9 3 3 P11hat],-mu,1)
lmiterm([9 3 4 -P31hat],-mu,1)
lmiterm([9 3 6 -HF],C1t',-1)
lmiterm([9 3 6 0],-C2t')
lmiterm([9 4 4 P21hat],-mu,1)
lmiterm([9 4 6 -LFhat],1,1)
lmiterm([9 5 5 gamma_u],-1,1)
lmiterm([9 5 6 -HF],HFt_u',1)
lmiterm([9 5 6 0],-Du)
lmiterm([9 6 6 gamma_u],-1,1)
lmiterm([9 7 7 P11hat],-inv(mu),1)
lmiterm([9 7 8 -P31hat],-inv(mu),1)
lmiterm([9 8 8 P21hat],-inv(mu),1)
```

```
lmiterm([10 1 1 V11],-1,1,'s')
lmiterm([10 1 2 -S1t],-1,1)
lmiterm([10 1 2 -S2],-1,1)
lmiterm([10 1 3 -V11],1,At2)
lmiterm([10 1 3 BFhatt],1,C4t)
lmiterm([10 1 3 P12hat],1,1)
lmiterm([10 1 4 AFhatt],1,1)
lmiterm([10 1 4 -P32hat],1,1)
lmiterm([10 1 5 -V11],1,But)
lmiterm([10 1 5 BFhatt],1,HFt_u)
lmiterm([10 1 7 -V11],1,1)
lmiterm([10 1 8 -S1t],1,1)
lmiterm([10 2 2 S1],-1,1,'s')
lmiterm([10 2 3 S2],1,At2)
lmiterm([10 2 3 BFhat],1,C4t)
lmiterm([10 2 3 P32hat],1,1)
lmiterm([10 2 4 AFhat],1,1)
lmiterm([10 2 4 P22hat],1,1)
lmiterm([10 2 5 S2],1,But)
lmiterm([10 2 5 BFhat],1,HFt_u)
lmiterm([10 2 7 S2],1,1)
lmiterm([10 2 8 -S1],1,1)
lmiterm([10 3 3 P12hat],-mu,1)
lmiterm([10 3 4 -P32hat],-mu,1)
lmiterm([10 3 6 -HF],C1t',-1)
lmiterm([10 3 6 0],-C2t')
lmiterm([10 4 4 P22hat],-mu,1)
lmiterm([10 4 6 -LFhat],1,1)
lmiterm([10 5 5 gamma_u],-1,1)
lmiterm([10 5 6 -HF],HFt_u',1)
lmiterm([10 5 6 0],-Du)
lmiterm([10 6 6 gamma_u],-1,1)
lmiterm([10 7 7 P12hat],-inv(mu),1)
lmiterm([10 7 8 -P32hat],-inv(mu),1)
lmiterm([10 8 8 P22hat],-inv(mu),1)

lmiterm([11 1 1 V11],-1,1,'s')
lmiterm([11 1 2 -S1t],-1,1)
lmiterm([11 1 2 -S2],-1,1)
lmiterm([11 1 3 -V11],1,At3)
lmiterm([11 1 3 BFhatt],1,C4t)
lmiterm([11 1 3 P13hat],1,1)
lmiterm([11 1 4 AFhatt],1,1)
lmiterm([11 1 4 -P33hat],1,1)
lmiterm([11 1 5 -V11],1,But)
lmiterm([11 1 5 BFhatt],1,HFt_u)
lmiterm([11 1 7 -V11],1,1)
lmiterm([11 1 8 -S1t],1,1)
lmiterm([11 2 2 S1],-1,1,'s')
lmiterm([11 2 3 S2],1,At3)
lmiterm([11 2 3 BFhat],1,C4t)
lmiterm([11 2 3 P33hat],1,1)
lmiterm([11 2 4 AFhat],1,1)
lmiterm([11 2 4 P23hat],1,1)
lmiterm([11 2 5 S2],1,But)
```

```
lmiterm([11 2 5 BFhat],1,HFt_u)
lmiterm([11 2 7 S2],1,1)
lmiterm([11 2 8 -S1],1,1)
lmiterm([11 3 3 P13hat],-mu,1)
lmiterm([11 3 4 -P33hat],-mu,1)
lmiterm([11 3 6 -HF],C1t',-1)
lmiterm([11 3 6 0],-C2t')
lmiterm([11 4 4 P23hat],-mu,1)
lmiterm([11 4 6 -LFhat],1,1)
lmiterm([11 5 5 gamma_u],-1,1)
lmiterm([11 5 6 -HF],HFt_u',1)
lmiterm([11 5 6 0],-Du)
lmiterm([11 6 6 gamma_u],-1,1)
lmiterm([11 7 7 P13hat],-inv(mu),1)
lmiterm([11 7 8 -P33hat],-inv(mu),1)
lmiterm([11 8 8 P23hat],-inv(mu),1)

lmiterm([12 1 1 V11],-1,1,'s')
lmiterm([12 1 2 -S1t],-1,1)
lmiterm([12 1 2 -S2],-1,1)
lmiterm([12 1 3 -V11],1,At4)
lmiterm([12 1 3 BFhatt],1,C4t)
lmiterm([12 1 3 P14hat],1,1)
lmiterm([12 1 4 AFhatt],1,1)
lmiterm([12 1 4 -P34hat],1,1)
lmiterm([12 1 5 -V11],1,But)
lmiterm([12 1 5 BFhatt],1,HFt_u)
lmiterm([12 1 7 -V11],1,1)
lmiterm([12 1 8 -S1t],1,1)
lmiterm([12 2 2 S1],-1,1,'s')
lmiterm([12 2 3 S2],1,At4)
lmiterm([12 2 3 BFhat],1,C4t)
lmiterm([12 2 3 P34hat],1,1)
lmiterm([12 2 4 AFhat],1,1)
lmiterm([12 2 4 P24hat],1,1)
lmiterm([12 2 5 S2],1,But)
lmiterm([12 2 5 BFhat],1,HFt_u)
lmiterm([12 2 7 S2],1,1)
lmiterm([12 2 8 -S1],1,1)
lmiterm([12 3 3 P14hat],-mu,1)
lmiterm([12 3 4 -P34hat],-mu,1)
lmiterm([12 3 6 -HF],C1t',-1)
lmiterm([12 3 6 0],-C2t')
lmiterm([12 4 4 P24hat],-mu,1)
lmiterm([12 4 6 -LFhat],1,1)
lmiterm([12 5 5 gamma_u],-1,1)
lmiterm([12 5 6 -HF],HFt_u',1)
lmiterm([12 5 6 0],-Du)
lmiterm([12 6 6 gamma_u],-1,1)
lmiterm([12 7 7 P14hat],-inv(mu),1)
lmiterm([12 7 8 -P34hat],-inv(mu),1)
lmiterm([12 8 8 P24hat],-inv(mu),1)

lmiterm([-13 1 1 P11hat],1,1)
lmiterm([-13 1 2 -P31hat],1,1)
lmiterm([-13 2 2 P21hat],1,1)
```

```
lmiterm([-14 1 1 P12hat],1,1)
lmiterm([-14 1 2 -P32hat],1,1)
lmiterm([-14 2 2 P22hat],1,1)

lmiterm([-15 1 1 P13hat],1,1)
lmiterm([-15 1 2 -P33hat],1,1)
lmiterm([-15 2 2 P23hat],1,1)

lmiterm([-16 1 1 P14hat],1,1)
lmiterm([-16 1 2 -P34hat],1,1)
lmiterm([-16 2 2 P24hat],1,1)

LMISYS=getlmis

[tmin,xfeas]=feasp(LMISYS)
tmin

nx=decnbr(LMISYS);
c=zeros(nx,1);
for i=1:nx
    [gamma_di,gamma_fi,gamma_ui]=defcx(LMISYS,i,gamma_d,gamma_f,gamma_u);
    c(i)=dd*gamma_di+ff*gamma_fi+uu*gamma_ui
end

[copt,xopt]=mincx(LMISYS,c);

evlmi=evallmi(LMISYS,xopt)
[lhs1,rhs1]=showlmi(evlmi,1)
[lhs2,rhs2]=showlmi(evlmi,2)
[lhs3,rhs3]=showlmi(evlmi,3)
[lhs4,rhs4]=showlmi(evlmi,4)

eig_lhs1=eig(lhs1)
eig_lhs2=eig(lhs2)
eig_lhs3=eig(lhs3)
eig_rhs4=eig(rhs4)

AFhatm=dec2mat(LMISYS,xopt,AFhat);
BFhatm=dec2mat(LMISYS,xopt,BFhat);
LFhatm=dec2mat(LMISYS,xopt,LFhat);
S1m=dec2mat(LMISYS,xopt,S1);
HFm=dec2mat(LMISYS,xopt,HF)
gamma_dm=dec2mat(LMISYS,xopt,gamma_d)
gamma_fm=dec2mat(LMISYS,xopt,gamma_f)
gamma_um=dec2mat(LMISYS,xopt,gamma_u)

[U,T]=schur(S1m);
V21T=U
V21=U'
V22inv=T'
AF=(inv(V21))'*AFhatm*inv(V21)*inv(V22inv)
BF=(inv(V21))'*BFhatm
LF=LFhatm*inv(V21)*inv(V22inv)
HFm
```

```
gamma_dm
gamma_um
gamma_fm
copt
```

APPENDIX D

SIMULATION DIAGRAM FOR 4.4.2 EXAMPLE 2