KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT TOEGEPASTE WETENSCHAPPEN
DEPARTEMENT ELECTROTECHNIEK
AFDELING ESAT
Kardinaal Mercierlaan 94 — 3001 Leuven (Heverlee)

ROBUSTNESS ANALYSIS AND CONTROLLER DESIGN FOR SYSTEMS WITH STRUCTURED UNCERTAINTIES

Jury :
Chairman : Prof. W. Dutré, vice-dean
Prof. B. De Moor, promotor
Prof. A. Barbé
Prof. J. De Schutter
Prof. J. Peperstraete
Dr. M. Steinbuch (Philips Research, NL)
Prof. A. Tits (Univ. Maryland, USA)(UCL)
Prof. J. Vandewalle

Dissertation submitted to obtain the degree of doctor in the applied sciences

by

YI Cheng

UDC : 681.51

May 1995

Acknowledgements

I wish to express my sincere gratitude and appreciation to my promotor Prof. B. De Moor for his constant supervision, stimulation and encouragement during the course of the research reported in this dissertation. His profound knowledge and broad research interests contributed a great deal to this work. His enthusiasm and creativity keep us all going in SISTA. Without him, it would have been impossible to finish the dissertation.

I would particularly like to thank Prof. J. Vandewalle and Prof. J. Peperstraete for providing me with the opportunity to study in ESAT, K.U. Leuven.

Many thanks go to the members of the reading committee, Prof. B. De Moor, Prof. J. Vandewalle, Dr. M. Steinbuch and Prof. A. Barbé for their critical review and constructive comments which have helped me to improve the quality of the dissertation. Thanks also go to the other members in the jury, Prof. A. L. Tits, Prof. J. De Schutter and Prof. J. Perperstraete.

I would like to thank my colleagues in ESAT-SISTA, both past and present, for the cooperation and assistance they provided in various way. In particular, I would like to thank Jeroen Dehaene for our cooporation work which has been partially reflected in the dissertation, Paul Vanvuchelen and Willem Minten for offering me the original data of the CVT, Johan David and Johan Suykens for numerous discussions, and SISTA secretary Ingrid Tokka for her daily help. I am also grateful to Ir. Ronnie Buckinx from V.C.S.T. Variable Transmission N.V. for helping me in all the experiments.

I am greatly indebted to the ESAT Laboratory and Katholieke Universiteit Leuven for their generous financial support during this research work.

Finally I would like to thank my wife, Yiqun, for her patience, understanding and love that have been of vital importance to the success of this work.

YI Cheng
May, 1995

Abstract

This dissertation treats the issues of robustness analysis, uncertainty modeling, controller design for systems with structured uncertainties.

For the **robustness analysis**, the focus is on numerical solution of the structured singular value or μ, which is a basic problem in robust control. The μ problem, cast as an optimization problem, is tackled by solving the related *optimality conditions* in an iterative way and by applying *general optimization algorithms* using function evaluation and gradient information. Several new algorithms are presented.

For the **structured uncertainty modeling**, a multidimensional realization algorithm is presented to solve parametric uncertainty modeling problems. Structured uncertainty normalization problems are also solved based on the sector transformation and loop transformation techniques.

For the **controller synthesis**, a D-K iteration procedure with mixed frequency and constant scalings is proposed. This procedure can be used to synthesize robust control for systems with both linear time invariant and nonlinear time varying uncertainties.

One **application** of the methods developed in this dissertation to an industrial hydraulic servo system of a continuously variable transmission is also presented.

Contents

Acknowledgements i

Abstract iii

Notation ix

Abbreviation xiii

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Overview of robust control . . . 7
  - 1.2.1 Input/output stability . . . 8
  - 1.2.2 Classes of uncertainties . . . 12
  - 1.2.3 Plant-uncertainty models . . . 14
  - 1.2.4 Performance specifications . . . 18
  - 1.2.5 Robustness analysis . . . 20
  - 1.2.6 Robust control synthesis . . . 24
- 1.3 Survey and contributions . . . 27
  - 1.3.1 Chapter by chapter survey . . . 27
  - 1.3.2 Main contributions . . . 31

2 Algorithms for calculation of lower bounds of μ 33
- 2.1 Preliminaries . . . 34
  - 2.1.1 Definition . . . 34
  - 2.1.2 Lower and upper bounds . . . 35
  - 2.1.3 Non convex optimization . . . 37
- 2.2 Continuous time power iteration . . . 39

  - 2.2.1 Optimality condition . . . 39
  - 2.2.2 Continuous time power iteration algorithm . . . 41
  - 2.2.3 Numerical examples . . . 44
- 2.3 Geometric optimization algorithms . . . 46
  - 2.3.1 Geometric properties of structured unitary matrices 46
  - 2.3.2 Gradient of spectral radius . . . 48
  - 2.3.3 The steepest ascent algorithm . . . 50
  - 2.3.4 The conjugate gradient algorithm . . . 53
  - 2.3.5 Numerical experiments . . . 55
- 2.4 Conclusions . . . 57

3 Real parametric μ calculation 59
- 3.1 Preliminaries . . . 60
  - 3.1.1 Definitions . . . 61
  - 3.1.2 Reformulation of the parameter margin problem . 62
- 3.2 Lower and upper bounds . . . 64
  - 3.2.1 Relations of μ_∞ and μ_2 bounds . . . 64
  - 3.2.2 Upper bound: LMI optimization . . . 65
  - 3.2.3 Lower bound: STLS . . . 65
- 3.3 Algorithms for lower bounds . . . 68
  - 3.3.1 Power method for μ_2 . . . 68
  - 3.3.2 Continuous time algorithm for μ_2 . . . 69
  - 3.3.3 Algorithms for μ_∞ . . . 70
- 3.4 Numerical examples . . . 73
- 3.5 Conclusions . . . 80

4 Parametric uncertainty modeling 83
- 4.1 System modeling and uncertainty modeling . . . 83
- 4.2 Multidimensional realization method . . . 85
  - 4.2.1 Definition . . . 86
  - 4.2.2 State space parametric uncertainty modeling . . . 87
  - 4.2.3 Formulation of the multiparameter margin problem 88
  - 4.2.4 Preliminary conditions . . . 89
- 4.3 Algorithm . . . 90
  - 4.3.1 Realization: affine combination . . . 92
  - 4.3.2 Realization: rational function . . . 93
  - 4.3.3 Data structure and program code . . . 100

4.4 Size reduction 102
4.4.1 Reducing size of each block one by one 103
4.4.2 Reducing the size for $\det(I + M\Delta(p)) = 0$ 105
4.5 Examples 107
4.6 Conclusions 110

5 Uncertainty normalization with LFTs 113
5.1 Preliminaries 114
5.1.1 Generalized sectors 114
5.1.2 Sector transformation 115
5.1.3 Loop shift transformation 115
5.1.4 Multiplier transformation 116
5.2 Normalization of general sectors 117
5.3 Normalization of some special sectors 122
5.3.1 General conic sector 122
5.3.2 Passive sector 124
5.4 Normalization of structured uncertainties 125
5.5 Examples 126
5.6 Conclusions 129

6 D-K iteration with mixed D-scalings 131
6.1 D-scaling for structured uncertainty 132
6.1.1 LTI and NLTV uncertainty structures 132
6.1.2 Frequency dependent D-scaling for LTI uncertainties 134
6.1.3 Constant D-scaling for NLTV uncertainties 135
6.1.4 Mixed D-scaling for LTI and NLTV uncertainties . 136
6.2 Calculation of D-scalings 137
6.2.1 Frequency dependent D-scalings 137
6.2.2 Constant D-scalings 139
6.2.3 Mixed D-scalings 140
6.3 D-K iteration 142
6.3.1 Control synthesis and D-K iteration 142
6.3.2 D-K iteration with mixed D-scalings 143
6.4 A practical example 145
6.5 Conclusions 148

7 Application to an hydraulic servo system 151
7.1 Process and uncertainties 152
7.2 Uncertainty modeling 156
7.2.1 System modeling 156
7.2.2 Practical procedure of uncertainty modeling 157
7.2.3 Modeling the uncertainties of the hydraulic circuit of CVT 158
7.2.4 Performance specification and open loop interconnection 168
7.3 Control design and robustness analysis 173
7.3.1 Classical PID auto-tuning 174
7.3.2 H_∞ control 178
7.3.3 μ control synthesis: standard D-K iteration 181
7.3.4 μ control synthesis: D-K iteration with mixed D-scaling 185
7.3.5 PID control design with parameter space search . 188
7.4 Time and frequency responses 190
7.4.1 Time responses 191
7.4.2 Frequency responses 192
7.5 Conclusions 193

8 Conclusions and open questions 195

A Convex and LMI optimization 199

B Super Block interconnection for the CVT hydraulic servo system 203

Bibliography 205

Notation

Notation	Meaning	Page
$\mathbb{R}$	field of real number	
$\mathbb{R}_+$	nonnegative real numbers	
$\mathbb{R}^{p\times q}$	field of real matrices with p rows and q columns.	
$\mathbb{Z}_+$	positive integers	
$\mathbb{C}$	field of complex number.	
$\mathbb{C}^{p\times q}$	field of complex matrices with p rows and q columns.	
$\mathbf{L}_p^n$ or $\mathbf{L}_p$	L_p-space: the set of all measurable functions $f(t):\mathbb{R}\to\mathbb{R}^n$, whose p-th power of vector l_p-norm are absolutely integrable over $[0,\infty)$.	8
$\mathbf{L}_{pe}^n$ or $\mathbf{L}_{pe}^n$	extended $\mathbf{L}_p^n$	9
$\mathcal{B}(\mathbf{L}_p^n)$	set of L_p-stable operators on $\mathbf{L}_p^n$	12
$\mathcal{S}^n$	set of LTI operators with stable proper rational transfer function matrices	12
$\mathcal{S}_n^p$	closure of $\mathcal{S}^n$ respect to L_p norm	12
$\mathcal{B}_N(\mathbf{L}_p^n)$	set of all causal nonlinear time varying operators in $\mathcal{B}(\mathbf{L}_p^n)$	12
$\mathbf{\Delta}$	uncertainty structure which is a set of block diagonal operators or matrices	13, 35, 132
$\mathbf{B\Delta}$	unit ball in $\mathbf{\Delta}$	14
$\mathbf{D}$	set of scaling matrices	22, 36, 133

$\mathbf{Q}$	the unitary matrix set in $\mathbf{\Delta}$	36
M^T	transpose of matrix M	
M^H	complex conjugate transpose of matrix M	
$\mathbf{diag}(a,b,\cdots,z)$	a block diagonal matrix, $\begin{bmatrix} a & & & 0 \\ & b & & \\ & & \ddots & \\ 0 & & & z \end{bmatrix}$	
$\mathrm{Tr}(M)$	trace of matrix M	
$\|\cdot\|_p$	signal L_p-norm ($\cdot$ is an signal vector), or vector l_p-norm ($\cdot$ is a parameter vector: $x\in\mathbb{C}^n$, $\|x\|_p=(\sum_{i=1}^n \|x_i\|^p)^{1/p}$, if p is missing, $\|\cdot\|$ is the Euclidean or l_2 norm)	9
P_T	standard truncation operator	9
$\|\cdot\|_{p-i}$ or $\gamma_p(\cdot)$	induced p-norm or L_p gain of a L_p-stable function	10
$\mathcal{T}(\cdot,\cdot)$	Linear fractional transformation	15
$\left[\begin{array}{c\|c} A & B \\ \hline C & D \end{array}\right]$	$=D+C(sI-A)^{-1}B$, a state-space realization of a linear plant	
$\left[\begin{array}{c\|cc} A & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{array}\right]=\begin{bmatrix} G_{11}(s) & G_{12}(s) \\ G_{21}(s) & G_{22}(s) \end{bmatrix}$	A state space realization of a linear plant with: $G_{ij}(s)=D_{ij}+C_i(sI-A)^{-1}B_j$	
$M>0$	M is a positive definite matrix	
$M\geq 0$	M is positive semi-definite matrix	
$\bar{\sigma}(M)$	largest singular value of a matrix M	
$\rho(M)$	spectral radius of a matrix M	
$\mu(M)$	structured singular value of a matrix M	21, 35
$\|G(s)\|_2$	H_2 norm of a stable transfer function $G(s)$	19
$\|G(s)\|_\infty$	H_∞ norm of a stable transfer function $G(s)$	19
$\|G(s)\|_D$	scaled H_∞-norm	173

$pm_k(A(p))$	parameter stability margin with k-norm	62
$\mu_k(M)$	real parametric μ of matrix M with k-norm	61
I_n	$n \times n$ identity matrix. Sometimes, the subscript will be omitted, when the size can be determined from context	
j	imaginary unit $\sqrt{-1}$.	
$G(s)$	transfer function for an LTI system	
$G(j\omega)$	frequency response for an LTI system	
M_ω	frequency dependent matrix	

Abbreviation

CGA	Conjugate Gradient Algorithm
CPMU2	Continuous time Power iteration algorithm for μ_2
CPMU∞	Continuous time Power iteration algorithm for μ_∞
CPIA	Continuous time Power Iteration Algorithm
CVT	Continuously Variable Transmission
LFT	Linear Fractional Transformation
LMI	Linear Matrix Inequality
LTI	Linear Time Invariant system
NLTV	NonLinear Time Varying system
PIA	Power Iteration Algorithm
PID	Proportional, Integral and Derivative
PMU2	Power iteration algorithm for μ_2
PMU∞	Power Iteration algorithm for μ_∞
SAA	Steepest Ascent Algorithm
STLS	Structured Total Least Squares

Chapter 1

Introduction

A fundamental problem in control system analysis and design is that of analyzing and designing control systems given a plant model that contains uncertainty. This is called robust control problem.

This dissertation focuses on robustness analysis and controller design for structured uncertainty systems.

In this chapter, we summarize the main results of the dissertation. In section 1.1, we first give a short motivation on dealing with the problem of robustness analysis and controller design for structured uncertainty systems. Then, in section 1.2 a brief overview is given on some topics that have appeared in the literature on robust control. Section 1.3 we give a chapter by chapter survey and discuss the main contributions of this dissertation.

1.1 Motivation

For model based control, a plant is represented by a mathematical model (nominal plant) which is built based on physical laws or experimental data via identification techniques. For mathematical reasons, such models are simple, and in many cases, they are linear and time invariant (LTI), we refer the reader to [53] for a comprehensive understanding of the methodology of modeling.

In practice, however, a real plant is always complex, and a mathematical model may never be an exact description of the plant. Thus what

can not be explained by the model is represented (or explained) by uncertainty. Thus robustness of control systems is always a major concern for control engineers because of the uncertainties inherently present in every model. It is necessary not only to design a controller for the nominal plant model, but for a whole class of models in its "neighborhood". Thus robustness can be described as follows:

> Given a set of models **M** and a property **P**. A control system is said to be robust to a given set **M** with a property **P** if property **P** is preserved for all the models in the given set **M**.

To define the problem more precisely, however, a number of problem elements must be carefully developed. In particular, classes of uncertainty models, plant-uncertainty models, and performance measures, we will discuss these elements in the next section.

Here we give a very simple example[1] to show our motivations. The system we will consider is a servomechanism illustrated in Fig. 1.1, where the aim is to control the speed of the inertia j_2 by applying a controlled torque to the inertia j_1: The control system is fed with a reference signal $\dot{\theta}_{ref}$ and a speed measurement $\dot{\theta}_2$. The drive motor is modeled by a simple gain m. Δ_1 represents the uncertainty of the actuator. The damping coefficients associated with j_1 and j_2 are d_1 and d_2 respectively. The load variations are modeled as a feedback uncertainty Δ_2. Numerical values for the problem data are in Table 1.1 in mks units. From the torque balance equations on j_1 and j_2, we can have a plant model P described by the following differential equations:

$$\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \dot{\theta}_e \end{bmatrix} = \begin{bmatrix} -d_1/j_1 & 0 & -k/j_1 \\ 0 & -d_2/j_2 & k/j_2 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \theta_e \end{bmatrix} + \begin{bmatrix} m/j_1 \\ 0 \\ 0 \end{bmatrix} e$$

$$\dot{\theta}_2 = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \theta_e \end{bmatrix},$$

[1] This example is taken from [40] with some modifications.

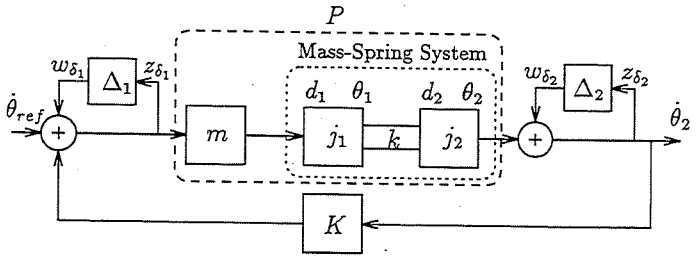


Figure 1.1 Mass-spring system, feedback controller

j_1	1.0
j_2	2.0
d_1	0.01
d_2	0.02
k	30.0
m	40
$\|\Delta_1\|_\infty$	≤ 0.5
$\|\Delta_2\|_\infty$	≤ 0.5

Table 1.1 Problem Data

where $\theta_e = \theta_1 - \theta_2$. Suppose now that a controller K is given as

$$K = \frac{a_5 s^5 + a_4 s^4 + a_3 s^3 + a_2 s^2 + a_1 s^1 + a_0}{s^6 + b_5 s^5 + b_4 s^4 + b_3 s^3 + b_2 s^2 + b_1 s^1 + b_0}.$$

where

$$\begin{aligned} a_5 &= -2.8692e07, & b_5 &= 2.5905e04, \\ a_4 &= -3.3895e09, & b_4 &= 1.0262e07, \\ a_3 &= -3.4287e12, & b_3 &= 7.8251e09, \\ a_2 &= -5.4429e13, & b_2 &= 1.4690e12, \\ a_1 &= 8.2650e13, & b_1 &= 4.8618e14. \\ a_0 &= -1.3082e15, & b_0 &= 1.7789e16. \end{aligned}$$

Our problem is as follows: Is the closed system stable for any stable

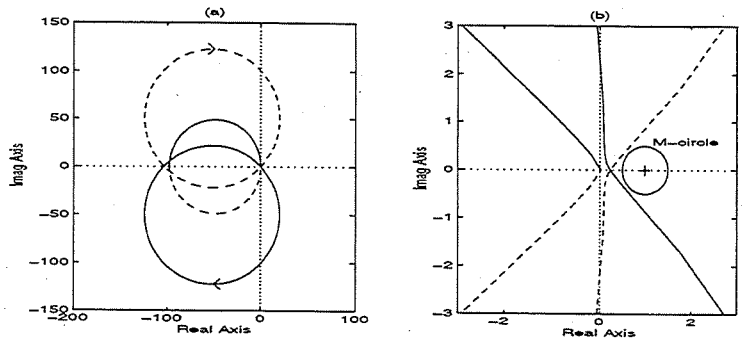


Figure 1.2 The Nyquist plot of the open loop system PK (a) and its enlarged part close to the M-circle (b)

linear time invariant Δ_1 and Δ_2 satisfying $\|\Delta_1\|_\infty < 0.5$ and $\|\Delta_2\|_\infty < 0.5$?

In the classical control, we can solve the problem in the frequency domain if there is only *one* SISO uncertainty, *e.g.*, Δ_2. The so-called M-circle (or D-circle depending on the position of the uncertainty) on the Nyquist plot of the open loop system (PK) is used to check the robust stability. Fig. 1.2 shows the Nyquist plot and the M-circle for the uncertainty Δ_2. Since the Nyquist plot does not enter the M-circle, the closed loop system is stable. However, in the case of two uncertainties as shown in Fig. 1.1, no conclusion can be reached by analyzing the two uncertainty systems with each including one uncertainty respectively. The singular value test offers a way to analyze robust stability when there is only *one* MIMO uncertainty. One, of course, could consider our two SISO uncertainties as a "big" MIMO uncertainty:

$$\Delta = \begin{bmatrix} \Delta_1 & 0 \\ 0 & \Delta_2 \end{bmatrix}. \tag{1.1}$$

Then, from the singular value test based on the small gain theorem or the multivariable Nyquist test, the closed loop system is stable if

$$\bar{\sigma}(G_{zw}(j\omega)) < 1/0.5 = 2 \quad \forall \omega \in \mathbb{R},$$

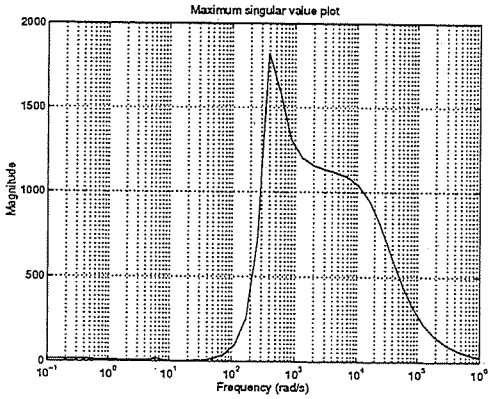


Figure 1.3 The maximum singular value plot of the closed loop system G_{zw}

where $G_{zw}(j\omega)$ is the frequency response of the system (not including Δ_1 and Δ_2) from w_{δ_1} and w_{δ_2} to z_{δ_1} and z_{δ_2}. Fig. 1.3 shows the singular value plot. The maximum singular value is about 1927 at frequency 426.6 rad/s. Thus the robust stability of the closed loop system is not guaranteed based on the singular value test. However the singular value test is a sufficient condition (for the robust stability) test. It is *not* a necessary condition test. Quite often it is conservative. The key point here is that both the M-circle test and the singular value test can not exploit the *structure* of the uncertainty.

The challenge to robustness analysis and controller design for systems with *structured* uncertainties is the main motivation for this research work, as claimed by the title of this dissertation.

To solve the problem of this example with the methods in this dissertation, we can first separate out and then normalize all the uncertainties by the use of the uncertainty modeling methods introduced in Chapter 4 and Chapter 5, and then the closed system is rearranged as a standard

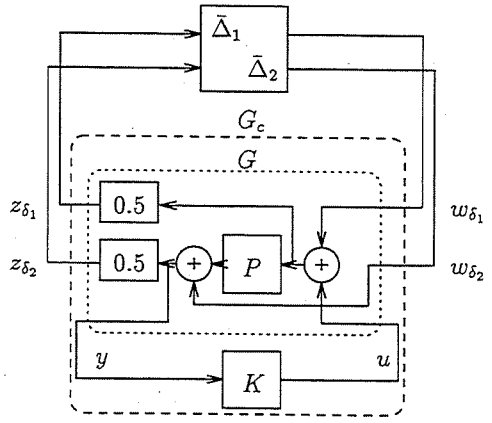


Figure 1.4 The standard plant-uncertainty interconnection.

plant-uncertainty interconnection[2] (see next section) shown in Fig. 1.4, where $\|\bar{\Delta}_1\|_\infty \leq 1$ and $\|\bar{\Delta}_2\|_\infty \leq 1$, G and G_c are called nominal open loop system and nominal closed loop system. Then based on the small μ theorem, the closed loop system is robustly stable if and only if

$$\mu(G_c(j\omega)) < 1 \quad \forall \omega \in \mathbb{R}.$$

where $\mu(G_c(j\omega))$ is the *structured singular value* of $G_c(j\omega)$ at the frequency ω corresponding to the uncertainty structure of (1.1). Chapter 2 presents some algorithms for the complex μ. The μ plot for our example here is shown in Fig. 1.5. The peak μ on the plot is about 0.85. Thus we conclude that the closed loop uncertain system is robustly stable! We have actually designed the controller K based on the μ synthesis method, which will be the topic of Chapter 6.

This example only shows the structure of the uncertainties based on the position of the uncertainties. Other structures of uncertainties can be exploited such as parametric uncertainties and nonlinear uncertainties,

[2] For such a simple uncertainty system, the uncertainty modeling is trivial.

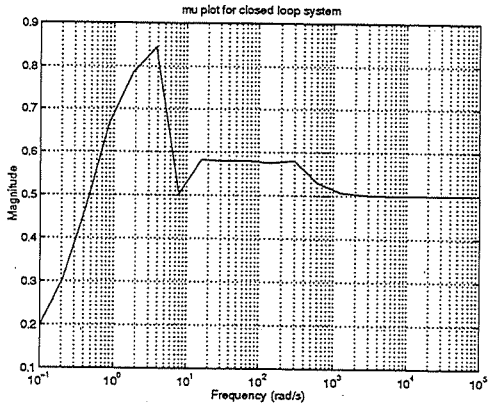


Figure 1.5 μ plot for G_c.

which will be discussed in Chapter 3 and Chapter 6. Chapter 7 gives an industrial application of our methods developed in this dissertation, which will further illustrate our motivations for this research work.

1.2 Overview of robust control

In this section we give a brief overview on some topics that have appeared in the literature on robust control. However this overview is not intented to be complete due to the large number of researchers and the huge amount of articles and books. Instead, we hope by this overview the readers will be able to situate better the contributions of the dissertation. This overview is focused on the following issues: input/output stability, classes of uncertainties, plant-uncertainty model, performance measures, robustness analysis and robust control design.

1.2.1 Input/output stability

Problems of stability are crucial in control system analysis and design. One of the fundamental differences between modern control (before 1980) and robust control (after 1980) is the way of stability analysis, the former uses the Lyapunov stability theory and the latter uses input/output stability theory mainly. Lyapunov stability theory is based on the systems under study being governed by a set of differential equations which describe the time evolution of the system state variables. In contrast, the systems encountered in input/output stability are assumed to be described by an input-output mapping that assigns, to each input, a corresponding output. Though an input/output representation and a state space representation are two different ways of looking at the same system, they each give a different kind of insight into how the system works. Especially the robust stability problem fits into the input/output stability framework naturally.

Input/output theory was pioneered by Sandberg and Zames in the 1960's [73, 74, 75, 101, 102]. In [23], Desoer and Vidyasagar gave a comprehensive discussion on the subject. Some relations between input/output stability and Lyapunov stability are discussed in [84] and the references therein.

$\mathbf{L}_p$-Spaces

Definition 1 *For each real $p \in [1, \infty)$, the set $\mathbf{L}_p^n[0,\infty) = \mathbf{L}_p^n$ consists of all measurable n-vector functions $f(\cdot) : \mathbb{R}_+ \to \mathbb{R}^n$ such that*

$$\int_0^\infty |f(t)|_p^p \, dt < \infty. \tag{1.2}$$

The set $\mathbf{L}_\infty^n[0,\infty) = \mathbf{L}_\infty^n$ consists of all measurable functions $f(\cdot)$: $\mathbb{R}_+ \to \mathbb{R}^n$ that are essentially bounded on $[0,\infty)$, where $|\cdot|_p$ is the vector l_p-norm.

Thus, for $p \in [1,\infty)$, $\mathbf{L}_p^n$ denotes the set of all measurable functions whose p-th powers of the vector l_p-norm are absolutely integrable over $[0,\infty)$, while $\mathbf{L}_\infty^n$ denotes the set of essentially bounded measurable functions.

Definition 2 *For $p \in [1,\infty)$, the p-norm of a function $f \in \mathbf{L}_p^n$, $\|f\|_p : \mathbf{L}_p^n \to \mathbb{R}_+$ is defined by*

$$\|f\|_p \triangleq \left[\int_0^\infty |f(t)|_p^p \, dt\right]^{1/p}. \tag{1.3}$$

The ∞-norm, $\|f\|_\infty : \mathbf{L}_\infty^n \to \mathbb{R}_+$ is defined by

$$\|f\|_\infty \triangleq \sup_{t\in[0,\infty)} |f(t)|_\infty. \tag{1.4}$$

This definition introduces the norm $\|\cdot\|_p$ for $p \in [1,\infty]$, which maps the set $\mathbf{L}_p^n$ into the half-line $\mathbb{R}_+$. It is also called signal p-norm.

Definition 3 *The set $\mathbf{L}_{pe}^n$ consists of all measurable functions: $f : \mathbb{R}_+ \to \mathbb{R}^n$ with the property that $P_T f \in \mathbf{L}_p^n$ for all finite T, and is called the extension of $\mathbf{L}_p^n$ or the extended $\mathbf{L}_p$-space, where P_T is the standard truncation operator:*

$$P_T f(t) = \begin{cases} f(t), & 0 \le t < T \\ 0, & T < t \end{cases}. \tag{1.5}$$

We shall sometimes dismiss the superscript n of $\mathbf{L}_p^n$ in case n is not relevant.

Input/output Stability

Though there are several definitions for input/output stability, and they are indeed different [84], here we only use the simplest one:

Definition 4 *Suppose a map $G : \mathbf{L}_{pe} \to \mathbf{L}_{pe}$. Then G is said to be L_p-stable if there exists a finite constant γ such that*

$$x \in \mathbf{L}_p \Rightarrow \|Gx\|_p \le \gamma\|x\|_p.$$

In this definition, attention is focused on the influence of inputs upon outputs, there is no "state" involved at all. While in Lyapunov stability theory, the stability is defined for unforced ordinary differential equations; there is no any input involved and the system evolves due to a nonzero initial state. However, these two stability definitions are actually closely related, if the system G is stabilizable and detectable.

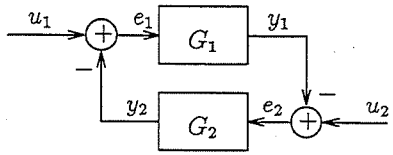


Figure 1.6 A feedback system.

Definition 5 *Suppose G is L_p-stable, then the L_p gain of G is defined as:*

$$\gamma_p(G) \triangleq \inf\{\gamma : \|Gx\|_p \le \gamma\|x\|_p, x \in \mathbf{L}_p\}. \tag{1.6}$$

This L_p gain is also called the induced operator norm over $\mathbf{L}_p$, or simply, induced p-norm, denoted by $\|\cdot\|_{p-i}$. Note that when G is linear and L_2-stable, then $\gamma_2(G) = \|G\|_{2-i} = \|G\|_\infty$.

So far we have been treating open loop input/output stability, however, for robust control, it is more relevant to consider input/output stability for feedback systems, of the form shown in Fig. 1.6. In this system, it is assumed that $G_1 : \mathbf{L}_{pe}^{n_1} \to \mathbf{L}_{pe}^{n_2}$ and $G_2 : \mathbf{L}_{pe}^{n_2} \to \mathbf{L}_{pe}^{n_1}$ with $p \in [0,\infty]$, so u_1, e_1, $y_2 \in \mathbf{L}_{pe}^{n_1}$ and u_2, e_2, $y_1 \in \mathbf{L}_{pe}^{n_2}$. Let $n = n_1 + n_2$, define maps $G, F \in \mathbf{L}_{pe}^n \to \mathbf{L}_{pe}^n$ by

$$G = \begin{bmatrix} G_1 & 0 \\ 0 & G_2 \end{bmatrix}, \quad F = \begin{bmatrix} 0 & I_{n_1} \\ -I_{n_2} & 0 \end{bmatrix},$$

and u, e, $y \in \mathbf{L}_{pe}^n$ by:

$$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \quad e = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}.$$

Then the overall system can described by the following two equations:

$$e = (I + FG)^{-1}u, \quad \text{and} \quad y = G(I + FG)^{-1}u,$$

where $(I + FG)^{-1}$ is the inverse of the map $(I + FG)$. These are just two open loop systems, $(I + FG)^{-1}$ and $G(I + FG)^{-1}$. It is proved that

$(I+FG)^{-1}$ is L_p-stable if and only if $G(I+FG)^{-1}$ is L_p-stable [84]. Thus, the input/output stability definition for the feedback system in Fig. 1.6 is given as follows:

Definition 6 *Suppose $G_1 : \mathbf{L}_{pe}^{n_1} \to \mathbf{L}_{pe}^{n_2}$ and $G_2 : \mathbf{L}_{pe}^{n_2} \to \mathbf{L}_{pe}^{n_1}$ with $p \in [0, \infty]$, and the map $(I + FG)^{-1}$ is well-posed. Then the feedback system in Fig. 1.6 is L_p-stable if $(I + FG)^{-1}$ is L_p-stable.*

Small gain theorem

A main result of input/output theory is the so-called small gain theorem, which gives a sufficient condition for the feedback system Fig. 1.6 to be L_p-stable. But first we give the definition of causal systems: a mapping G: $\mathbf{L}_p \to \mathbf{L}_p$ is said to be causal if

$$P_T(G(f)) = P_T(G(P_T(f)),\ \forall T \geq 0,\ \forall f \in \mathbf{L}_p.$$

The following is the small gain theorem.

Theorem 1 *Consider the feedback system in Fig. 1.6, and suppose $p \in [1, \infty]$ is specified. Suppose in addition that both G_1 and G_2 are causal and L_p-stable. Under these conditions, the system of Fig. 1.6 is L_p-stable if*

$$\gamma_p(G_1)\gamma_p(G_2) < 1. \tag{1.7}$$

(1.7) is often called small gain condition.

This small gain theorem plays a fundamental rule in robust control theory. Suppose G_1 is the uncertainty and G_2 is the nominal plant, then the feedback system of Fig. 1.6 represents the corresponding real plant. Suppose both G_1 and G_2 are norm bounded, the robustness analysis problem is to find the condition under which the real plant is stable. So the small gain theorem could be used to solve the problem. Note that for using the small gain theorem, we do not need to know the structures of G_1 and G_2: they can be linear or nonlinear, time invariant or time varying, lumped parametric or distributed systems. This therefore makes the small gain theorem rather versatile for feedback stability analysis. However, on the other hand, the small gain theorem only gives a sufficient condition for stability when G_1 is structured, and in this case it often

gives conservative results, and such conservativeness could be arbitrary large. Thus in robust control, it is one of the main subjects to reduce the conservativeness of the small gain condition, which is also the main concern of this dissertation.

1.2.2 Classes of uncertainties

For robust control, uncertainty is not defined as something which is completely unknown, instead it is defined as a set of unknown operators with some known structures (or properties). Obviously, the more we know on the uncertainty structure, the more accurate results we can get in analyzing and designing the control systems with the uncertainty. In this subsection, a precise description of the classes of uncertainties commonly used will be presented. We first describe three basic classes of uncertainty building blocks and then define a general structured uncertainty set from these building blocks. We denote by $\mathcal{B}(\mathbf{L}_p^n)$ the space of all bounded operators from $\mathbf{L}_p^n$ to $\mathbf{L}_p^n$, or L_p-stable operators. (For the sake of simplicity, only 'square' operators are considered as uncertainty building blocks). The sources of this subsection are mainly from [18, 5].

Linear time-invariant uncertainties

The first subset of $\mathcal{B}(\mathbf{L}_p^n)$, which will provide a rich class of uncertainties is the subset of linear time invariant (LTI) operators. Let $\mathcal{S}^n$ be the set of stable proper rational transfer function matrices. Any element in $\mathcal{S}^n$ defines a bounded operator on $\mathbf{L}_p^n$, and thus $\mathcal{S}^n \subset \mathcal{B}(\mathbf{L}_p^n)$. Take the closure for the set $\mathcal{S}^n$ with respect to the norm in $\mathcal{B}(\mathbf{L}_p^n)$ and denote this closure by $\mathcal{S}_p^n$. Therefore $\mathcal{S}_p^n$ is the smallest complete algebra of systems that contains the finite-dimensional, stable, time-invariant systems. Special cases that are frequently used in the literature are $\mathcal{S}_2^n \subset \mathrm{H}_\infty$ and $\mathcal{S}_1^n = \mathcal{S}_\infty^n$. The uncertainties in this class represent some unknown but norm bounded LTI dynamics.

Nonlinear and time varying uncertainties

The class of nonlinear and time varying uncertainties (NLTV) is a larger subset of uncertainties than $\mathcal{S}_p^n$, which is contained in $\mathcal{B}(\mathbf{L}_p^n)$. Denote by $\mathcal{B}_N(\mathbf{L}_p^n)$ the elements in $\mathcal{B}(\mathbf{L}_p^n)$ that are causal. $\mathcal{B}_N(\mathbf{L}_p^n)$ actually forms a

non-commutative, left-distributive algebra by defining induced p-norm in normal way. The lack of non-commutation and right-distribution in this algebra presents an essential technical difficulty in robustness analysis. ($\mathcal{S}_p^n$ is commutative in the scalar case when $n = 1$, and is left-right-distributive always.)

Real parametric uncertainties

The class of real parametric uncertainties is defined as a set of real numbers in $\mathbb{R}$. This class of uncertainties represents localized uncertainties, that can occur at any point in a feedback loop. Such uncertainties are not dynamic, and are in some sectors in $\mathbb{R}$, thus there is no difference due to induced p-norm.

Structured uncertainties

If there is only one uncertainty in a plant, which belongs to any one of the classes of uncertainty building blocks defined above (though it can be MIMO), then the uncertainty is called unstructured. However, uncertainties can occur at different locations simultaneously in a plant. If that is the case, we can take them together as one block, but with a certain specific structure which is block diagonal. The diagonal structure of the uncertainties is essential for incorporating information about the location of the system uncertainty. The uncertainty block with such a structure is called a structured uncertainty. A general structured uncertainty set is defined as follows:

$$\mathbf{\Delta} \triangleq \{\mathbf{diag}(\Delta_r, \Delta_l, \Delta_n); \Delta_r \in \mathbf{\Delta}_r, \Delta_l \in \mathbf{\Delta}_l, \Delta_n \in \mathbf{\Delta}_n\}, \tag{1.8}$$

where

$$\mathbf{\Delta}_r \triangleq \{\mathbf{diag}(\delta_1, \cdots, \delta_{S^r}); \delta_i \in \mathbb{R}, 1 \le i \le S^r\},$$

$$\mathbf{\Delta}_l \triangleq \{\mathbf{diag}(\Delta_1, \cdots, \Delta_{F^l}); \Delta_i \in \mathcal{S}_p^{m_i^l}, 1 \le i \le F^l\},$$

$$\mathbf{\Delta}_n \triangleq \{\mathbf{diag}(\Delta_1, \cdots, \Delta_{F^n}); \Delta_i \in \mathcal{B}_n(\mathbf{L}_p^{m_i^n}), 1 \le i \le F^n\},$$

where $\mathbf{\Delta}_r$, $\mathbf{\Delta}_l$ and $\mathbf{\Delta}_n$ are classes of real parametric, LTI and NLTV uncertainty respectively, S^r, F^l and F^n are integers denoting the block

numbers in corresponding classes, m_i^l and m_i^n are the dimensions of corresponding uncertainty blocks. The underlying structure of this general uncertainty set depends on the integer S^r and two integer vectors defined as:

$$m^l = [m_1^l, \cdots, m_{F^l}^l],$$

$$m^n = [m_1^n, \cdots, m_{F^n}^n],$$

with each element being the dimension of the corresponding uncertainty block. An even more general structured uncertainty set can be constructed if one considers repeated scalar blocks with a form as:

$$\delta I_n,$$

where $\delta \in \mathbb{R}$ or $\mathcal{S}_p^1$ or $\mathcal{B}_N(\mathbf{L}_p^1)$.

Normalized structured uncertainty

For simplifying the analysis, all blocks in a structured uncertainty Δ in (1.8) and thus Δ itself are supposed to be normalized in a closed unit ball:

$$\mathbf{B\Delta} \triangleq \{\Delta \mid \Delta \in \mathbf{\Delta}, \ \|\Delta\|_p \le 1\}. \tag{1.9}$$

When $p = 2$, this condition is also called non-expansive condition. Normalization of structured uncertainty to be in a unit ball without introducing any conservativeness is a main subject for uncertainty modeling, and we will present our results on this subject in Chapter 5.

1.2.3 Plant-uncertainty models

In the earlier times of robust control, only one uncertainty block, namely an unstructured uncertainty, was considered. Main types of the plant uncertainty models are additive and multiplicative plant-uncertainty models. The uncertainties in these types are connected to plant models in additive or multiplicative way respectively. Another type of uncertainties are coprime factor uncertainties with two additive uncertainties on two coprimefactors [17]. There was no uniform way to describe plant-uncertainty models until LFT (linear fractional transformation) was used

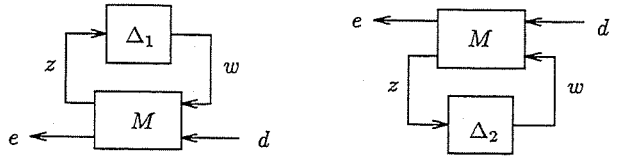


Figure 1.7 Left: an upper LFT, Right: a lower LFT.

together with the structured uncertainty models defined above. In this subsection, we'll first give the definition for LFT, and then give an LFT-type plant-uncertainty model.

LFTs

The notation of linear fractional transformations (LFTs) is the generalization of the same notation in the theory of functions of one complex variable to the matrix case [16].

Given two matrices:

$$M = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \in \mathbb{C}^{(p_1+p_2)\times(q_1+q_2)} \quad \text{and} \quad \Delta \in \mathbb{C}^{q_1 \times p_1}.$$

Note the compatibility of the two matrices in dimensions. The upper LFT with coefficient matrix M and parameter matrix Δ is defined as:

$$\mathcal{T}(\Delta, M) \triangleq M_{22} + M_{21}\Delta(I - M_{11}\Delta)^{-1}M_{12}. \tag{1.10}$$

If $\Delta \in \mathbb{C}^{q_2 \times p_2}$, then the lower LFT is defined as:

$$\mathcal{T}(M, \Delta) \triangleq M_{11} + M_{12}\Delta(I - M_{22}\Delta)^{-1}M_{21}. \tag{1.11}$$

Clearly the existence of inverse $(I - M_{11}\Delta)^{-1}$ and $(I - M_{22}\Delta)^{-1}$ is necessary for the LFTs to be well defined. The LFT formulae arise naturally when describing feedback systems as shown in Fig. 1.7. Note that for both definitions (1.10) and (1.11), there is always a (dimensionally) smaller matrix (here Δ), and all inputs/outputs into and from the smaller matrix are connected in the interconnecting transformation. Hence, we do not

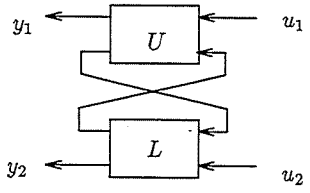


Figure 1.8 Star product.

need to specify the dimensions and locations of the interconnecting channels, since the interconnecting dimensions are equal to the dimension of the smaller matrix, and the locations of the channels are indicated by the position of the small matrix and the big matrix(here M) in $\mathcal{T}(\cdot,\cdot)$. Actually, these two LFTs are special cases of a more general linear feedback loop, called "Star product" of Redheffer [65], defined as follows. Given two matrices U and L as:

$$U = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix} \in \mathbb{C}^{(r_1+r)\times(s_1+s)},$$

$$L = \begin{bmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{bmatrix} \in \mathbb{C}^{(s+r_2)\times(r+q_2)}.$$

Note the compatibility of U and L in dimensions. The star product is defined as:

$$\mathcal{T}(U, L, s, r) \triangleq \begin{bmatrix} \mathcal{T}(U, L_{11}) & U_{12}(I - L_{11}U_{22})^{-1}L_{12} \\ L_{21}(I - U_{22}L_{11})^{-1}U_{21} & \mathcal{T}(U_{22}, L) \end{bmatrix}. \tag{1.12}$$

The star product can be described by the linear feedback loop as shown in Fig. 1.8. Any LFT is just a special case of a star product where $r_1 = s_1 = 0$ (Upper LFT) or $r_2 = s_2 = 0$ (Lower LFT). Compared to these LFTs, the notation for star product is $\mathcal{T}(\cdot,\cdot,\cdot,\cdot)$ with two additional arguments (s and r in (1.12)) to specify the dimensions of the partitions.

An operator LFT is defined exactly as matrix LFT in (1.10) and (1.11) but with all matrices replaced by operators, see [40] for more details on operator LFTs.

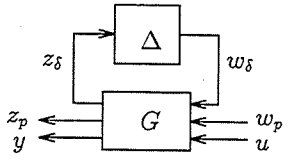


Figure 1.9 Plant-uncertainty interconnection.

Plant-uncertainty models

In robust control, a general plant-uncertainty model is described by an LFT as

$$\mathcal{T}(\Delta, G)$$

and Fig. 1.9 shows the interconnection, where z_δ and w_δ are the input and output of the uncertainty; z_p and w_p are the regulated output and exogenous (command or disturbance) input of the plant and y and u are the measurement and control input respectively. The plant model[3] is represented by an LTI system G and

$$G = \left[\begin{array}{c|ccc} A & B_{w_\delta} & B_{w_p} & B_u \\ \hline C_{z_\delta} & D_{z_\delta w_\delta} & D_{z_\delta w_p} & D_{z_\delta u} \\ C_{z_p} & D_{z_p w_\delta} & D_{z_p w_p} & D_{z_p u} \\ C_y & D_{y w_\delta} & D_{y w_p} & D_{y u} \end{array}\right] = \left[\begin{array}{ccc} G_{z_\delta w_\delta} & G_{z_\delta w_p} & G_{z_\delta u} \\ G_{z_p w_\delta} & G_{z_p w_p} & G_{z_p u} \\ G_{y w_\delta} & G_{y w_p} & G_{y u} \end{array}\right].$$

Uncertainty Δ is defined as above. In this plant-uncertainty setup,

$$\mathcal{T}(0, G) = \left[\begin{array}{cc} G_{z_p w_p} & G_{z_p u} \\ G_{y w_p} & G_{y u} \end{array}\right]$$

may be thought of as the nominal plant model. $G_{z_\delta w_\delta}$, $G_{z_\delta w_p}$, $G_{z_\delta u}$, $G_{z_p w_\delta}$ and $G_{y w_\delta}$ together with the formula $\mathcal{T}(\cdot,\cdot)$ reflect prior knowledge

[3] We also include all LFTs for uncertainty normalization, see Chapter 5, and weighting functions for performance specifications in this plant model.

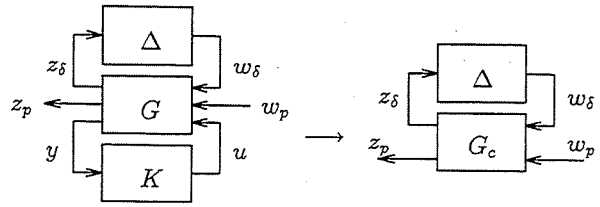


Figure 1.10 Closed loop plant-uncertainty model.

on how the uncertainty affects the nominal plant. This LFT-type of plant-uncertainty model is natural for robust control problem and it actually defines the model set M:

$$\mathrm{M} \triangleq \{\mathcal{T}(\Delta, G) \mid \Delta \in \mathrm{B}\Delta\}.$$

Separating out the uncertainty Δ from the plant model G is called uncertainty modeling, which will be discussed in Chapter 4 of this dissertation.

The plant-uncertainty model in Fig. 1.9 is called an open loop system since there is no controller feedback connected. Robustness analysis is made usually for a closed loop system shown in Fig. 1.10, where K is a controller. This closed loop system is also described by an LFT:

$$\mathcal{T}(\Delta, G_c),$$

where $G_c = \mathcal{T}(G, K)$ with partitioning:

$$G_c = \left[\begin{array}{cc} (G_c)_{z_\delta w_\delta} & (G_c)_{z_\delta w_p} \\ (G_c)_{z_p w_\delta} & (G_c)_{z_p w_p} \end{array}\right].$$

1.2.4 Performance specifications

Performance specifications describe how the closed loop system should perform. A comprehensive discussion on performance specifications can be found in [11] by Boyd and Barratt. Here we give a brief description on the performance specifications with emphasis on structures of the performance specifications.

Performance specifications are defined on the the input/output pair $(w_p,\ z_p)$ of a closed loop system. When w_p is a command input, the specifications are input/output specifications (*e.g.* step response, tracking error specifications), or when w_p is a disturbance input, the specifications are regulation specifications (*e.g.* classical frequency domain regulation). However from the mathematical point of view, most of these specifications are of the same format: a certain norm of the operator or transfer function from w_p to z_p of the closed loop system must be small. This format can be specified in a uniform way as:

$$\|\mathcal{T}(\Delta, G_c)\| < 1. \tag{1.13}$$

The norm $\|\cdot\|$ used here can be any induced p-norm or H_2-norm. Actually, only induced 1-, 2- and ∞-norms and the H_2-norm have been explored so far in the literature. Induced 1- and ∞-norms are equal for a LTI system [5], and often the induced ∞-norm is called the L_1 norm. The induced ∞-norm measures the system's amplification of L_∞ signal. The induced 2-norm measures the system's gain of signal energy of L_2 signals and for an LTI system coincides with H_∞-norm of the transfer function of that system [36], which is defined as

$$\|G\|_\infty \triangleq \sup_{\alpha>0}\sup_{\omega} \bar{\sigma}(G(\alpha+j\omega)). \tag{1.14}$$

Sometimes, the H_∞-norm is also abused for NLTV systems as its induced 2-norm. The H_2-norm is defined on the transfer function of a stable LTI system [36]. :

$$\|G\|_2 \triangleq \left(\sup_{\alpha>0}\frac{1}{2\pi}\int_{-\infty}^{\infty} G^H(\alpha+j\omega)G(\alpha+j\omega)d\omega\right)^{1/2}. \tag{1.15}$$

This norm is not an induced norm as defined above. However it can also be interpreted as an induced norm in a stochastic context: the H_2-norm measures the RMS (root mean square) response of its output when it is driven by a white noise excitation.

The performance specification in (1.13) is "unstructured" since we specify a norm on all inputs to all outputs. Now partition w_p and z_p as:

$$w_p = \begin{bmatrix} w_{p1} \\ w_{p1} \end{bmatrix},\ z_p = \begin{bmatrix} z_{p1} \\ z_{p2} \end{bmatrix}.$$

The performance specifications can be set to each of the input/output pairs: (w_{p1}, z_{p1}) and (w_{p2}, z_{p2}). That is:

$$\|(\mathcal{T}(\Delta, G_c))_{z_{p1}w_{p1}}\|_a < 1,\ \|(\mathcal{T}(\Delta, G_c))_{z_{p2}w_{p2}}\|_b < 1, \tag{1.16}$$

where the norms $\|\cdot\|_a$ and $\|\cdot\|_b$ can be the same or different. This kind of performance specifications are thus "structured" and reflect the requirements of specific inputs to specific outputs. A typical example of such structured performance specification is the mixed H_2/H_∞ [17] specification.

In this dissertation we are only interested in induced 2-norm performance specifications.

1.2.5 Robustness analysis

Robustness analysis is to verify whether or not a closed loop system in Fig. 1.10 with a given controller that stabilizes the nominal plant satisfies the stability and performance specifications for all uncertainties $\Delta \in \mathbf{B}\mathbf{\Delta}$. Robustness analysis includes: robust stability analysis, in which case the property $\mathbf{P}$ is the stability and robust performance analysis, in which case $\mathbf{P}$ is performance. Robustness analysis is also called μ-analysis when $\mathbf{\Delta}$ is structured and the norm used is the induced 2-norm or ∞-norm.

Robust stability

As mentioned earlier, robust stability naturally fits into in the framework of input/output stability of feedback systems. The following definition of robust stability is just an alternative version of input/output stability of feedback systems by definition 6 (for all $\Delta \in \mathbf{\Delta}$) and is adopted from [18, 5]:

Definition 7 *The system in Fig. 1.11 is robustly* L_p*-stable for a set of uncertainties* $\mathbf{\Delta}$*, if*

1. *for all* $\Delta \in \mathbf{\Delta}$, $(I + G_c\Delta)$ *has an inverse in* $\mathcal{S}_p^n$ *for LTI cases and is in* $\mathcal{B}_N(\mathbf{L}_p^n)$ *for NLTV cases.*
2. $\|(I + G\Delta)\|_{p-i} < \infty$.

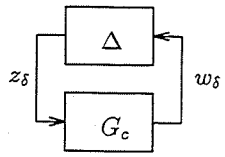


Figure 1.11 Robust stability configuration.

The equivalence to input/output stability for feedback systems can be found in [23].

For LTI and parametric uncertainties, $\mathbf{\Delta}_{rl} \triangleq \mathbf{diag}(\mathbf{\Delta}_r, \mathbf{\Delta}_l)$, the techniques based on the structured singular value (μ-analysis) play an important role in the study of robust stability [26, 62]. The structured singular value on a matrix $G(j\omega)$ with respect to a class of time invariant uncertainties $\mathbf{\Delta}$ is defined as follows:

$$\mu(G(j\omega))^{-1} \triangleq \min\{\bar{\sigma}(\Delta(j\omega)) : \det(I+G\Delta)(j\omega) = 0, \Delta \in \mathbf{\Delta}_{rl}\}. \quad (1.17)$$

We shall give formal definitions of μ and algorithms in Chapter 2 and 3. Then the following theorem gives a necessary and sufficient condition for the robust L_2-stability of systems with LTI and parametric uncertainties [18]:

Theorem 2 *Consider the interconnection in Fig. 1.11, let $M \in \mathcal{S}_n^2$ with $n = S^r + \sum_{i=1}^{F^l} m_i^l$. The following statements are equivalent:*

1. *The system is robustly L_2-stable for the uncertainty set $\mathbf{B\Delta}_{rl}$.*
2. *$\sup_{\omega\in\mathbb{R}} \mu(G_c(j\omega)) < 1$, where μ is computed with respect to the uncertainty structure $\mathbf{\Delta}_{rl}$.*

This theorem is the so-called small μ theorem. In the case that $\mathbf{B\Delta}_{rl}$ has only one block, the theorem is just the small gain theorem for LTI systems since in this case $\mu(G_c(j\omega)) = \bar{\sigma}(G_c(j\omega))$. The plot of $\mu(G_(j\omega))$ as a function of ω is called μ plot. When there is only one block, then the μ plot is called singular value plot. μ plots are very useful for robustness analysis.

For NLTV uncertainties, the following theorem gives a necessary and sufficient condition [78]:

Theorem 3 *Consider the interconnection in Fig. 1.11, let $G_c \in \mathcal{S}_2^n$ with $n = \sum_i^{F^n} m_i^n$. The following statements are equivalent:*

1. *The system is robustly L_2-stable for the uncertainty set $\mathbf{B\Delta}_n$.*
2. *$\inf_{D\in\mathbf{D}} \|DG_cD^{-1}\|_\infty < 1$ where*

$$\mathbf{D} = \mathbf{diag}\{d_1 I_{m_1^n}, \cdots, d_{F^n} I_{m_{F^n}^n}, d_i \in \mathbb{R}, d_i \neq 0\}. \quad (1.18)$$

This theorem is also called the scaled small gain theorem with constant scalings. Obviously when $D = I$, then this is just the small gain theorem. The scaled small gain theorem can also be used for LTI uncertainties if the scalings $d_i \in \mathcal{S}_2^1$ and the block number is less than 3 [26].

Robust performance

Here we only consider an unstructured performance specification and the case in which performance specification has a norm consistent to the uncertainty norm.

Definition 8 *The system in Fig. 1.10 achieves L_p robust performance for a set uncertainties $\mathbf{B\Delta}$ if*

1. *The system is robustly L_p-stable for the set of uncertainties $\mathbf{B\Delta}$.*
2. *$\sup_{\Delta\in\mathbf{B\Delta}} \|\mathcal{T}(\Delta, G_c)\|_{p-i} < 1$.*

Robust performance is, in general, a much harder problem to analyze than robust stability. However in certain cases, the equivalence between robust stability and performance can be obtained by introducing an artificial uncertainty block into the real uncertainty structure. Suppose that the number of inputs w_p and outputs z_p is the same, namely, n_p. The following two theorems give conditions for the equivalence between robust stability and robust performance for two special cases with uncertainty structures of $\mathbf{\Delta}_{rl}$ and $\mathbf{\Delta}_n$, and they are based on Theorem 2 and Theorem 3 [18].

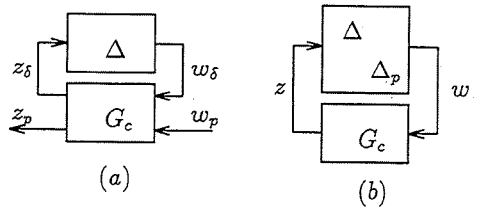


Figure 1.12 Convert a robust performance problem (a) to a robust stability problem (b) by augmenting (a) with an artificial uncertainty block Δ_P. Here $z = [z_\delta^T \;\; z_p^T]^T$ and $w = [w_\delta^T \;\; w_p^T]^T$.

Theorem 4 *Consider the interconnection in Fig. 1.10, let $G \in \mathcal{S}_2^n$ where $n = S^r + \sum_i^{F^l} m_i^l + n_p$, The following statements are equivalent:*

1. *The system achieves L_2-robust performance with respect to the uncertainty structure $\mathbf{B}\boldsymbol{\Delta}_{rl}$.*
2. *The system is robustly L_2-stable with respect to the uncertainty structure $\mathbf{diag}(\mathbf{B}\boldsymbol{\Delta}_{rl}, \mathbf{B}\boldsymbol{\Delta}_P)$, where*

$$\mathbf{B}\boldsymbol{\Delta}_P = \{\Delta|, \Delta \in \mathcal{S}_2^{n_P}, \|\Delta\|_\infty < 1\}.$$

This theorem gives a necessary and sufficient condition for equivalence of L_2 robust performance with parametric and LTI uncertainties and a L_2 robust stability condition for a modified uncertainty structure with an additional LTI uncertainty block, see Fig. 1.12.

Theorem 5 *Consider the interconnection in Fig. 1.10, and let $G \in \mathcal{S}_2^n$ with $n = \sum_{i=1}^{F^n} m_i^n + n_p$. The following statements are equivalent:*

1. *The system achieves L_2 robust performance with respect to the uncertainty structure $\mathbf{B}\boldsymbol{\Delta}$.*
2. *The system is L_2 robustly stable with respect to the uncertainty structure $\mathbf{diag}(\mathbf{B}\boldsymbol{\Delta}_n, \mathbf{B}\boldsymbol{\Delta}_P)$, where:*

$$\mathbf{B}\boldsymbol{\Delta}_P = \{\Delta|, \Delta \in \mathcal{B}_N(\mathbf{L}_2^{n_P}), \|\Delta\|_{2-i} < 1\}.$$

Table 1.2 Classification of linear controller designs.

		Performance	
		Unstructured	Structured
Controller	unstructured	LQG(H_2)/LTR,H_∞ L_1, Pole placement	μ-synthesis H_2/H_∞ synthesis
	structured	PID, Lead-lag Fixed order controller	PID, Lead-lag Fixed order controller

This theorem give a necessary and sufficient condition for the equivalence of L_2 robust performance with NLTV uncertainties and the robust L_2-stability condition for a modified uncertainty structure with an additional NLTV uncertainty block.

1.2.6 Robust control synthesis

The problem of robust control system synthesis is to design a controller K such that the closed loop system in Fig. 1.10 satisfies both robust stability and robust performance specifications. Note that robust stability and robust performance requirements can be eventually represented by structured performance requirements on G_c with input $z = \begin{bmatrix} z_\delta \\ z_p \end{bmatrix}$ and output $w = \begin{bmatrix} w_\delta \\ w_p \end{bmatrix}$, which can be seen from Theorem 4 and Theorem 5 for two special cases. So for control synthesis, we only consider G and the structured performance requirements on G and distinguishing z_δ/w_δ and z_p/w_p is not necessary. Like the structured performance, the controller K can be also structured, but with a different meaning now on structure. The controller K is unstructured if K is allowed to be any LTI systems, otherwise, K is structured as would be the case if for instance we require it to be a PID. We classify the robust control synthesis based on the structures of both performance and controller. Table 1.2 summarizes the classification and enumerates some typical existing controller synthesis methods for each class.

Unstructured performance and unstructured controller

Theoretically, controller designs with unstructured performance and unstructured controller are easiest among all the controller design categories in Table 1.2. Actually it is the only category which contains some analytical solutions like H_∞ and H_2 controllers.

H_∞ control plays a key role in robust control design. The goal of optimal H_∞ control design is to minimize the H_∞-norm of a closed loop system:

$$\min_{K,\ \text{stabilizing}} \|\mathcal{T}(G,K)\|_\infty.$$

Obviously K must be a stabilizing controller.

The robustness of the H_∞ control is based on the small gain theorem. Recall that $\|\mathcal{T}(G,K)\|_\infty < 1$ is the exact robust stability condition from Theorem 3 when there is only one uncertainty block. H_∞ control may also be used for closed-loop performance specification with the H_∞-norm, which is used in tracking or disturbance attenuation problems. The research work on H_∞ control was pioneered by Zames [103] and it was eventually solved by Doyle, Glover, Khargonekar, and Francis [28] (known as the DGKF paper). It turns out that (sub) optimal H_∞ control can be solved by solving just two Riccati equations, and the controller has a separation structure as in LQG or H_2 control theory and has the same order as the plant G. We refers to [28] for more details.

The LQG (Linear Quadratic Gaussian) regulator or H_2 controller minimize the H_2 norm of the closed loop system:

$$\min_{K,\ \text{stabilizing}} \|\mathcal{T}(G,K)\|_2.$$

The solution of the LQG problem is a combination of the solutions of Kalman filtering and full-state feedback based on the so-called separation principle. Robust property (in terms of classical gain and phase margins) can be very poor [24]. The LQR (linear quadratic regulator) with full-state feedback has however good robust property (good gain and phase margin at the input). The LQG/LTR, LQG loop transfer recovery design method is developed to recover the robust property of LQR as much as possible [25].

For L_1 control synthesis, we refer to a recent survey paper [19].

Structured performance and unstructured controller

Control synthesis in this category is already complicated, since the problem in general is not convex. Basically we can divide this category into two parts: structured performance with uniform norms and structured performance with mixed norms.

For structured performance with uniform norms, the uncertainty $\Delta \in \mathbf{B}\mathbf{\Delta}$ and performance specifications might be structured but with induced p-norm specifications: both $\|\cdot\|_a$ and $\|\cdot\|_b$ in (1.16) must be induced p-norms. Control synthesis with such structured performance with uniform norms is called μ-synthesis which can be cast into the following optimization problem:

$$\min_{K,D\in\mathbf{D}} \|D\mathcal{T}(G,K)D^{-1}\|_{p-i}, \tag{1.19}$$

where set $\mathbf{D}$ has the structure shown in (1.18), but now $d_i, d_i^{-1} \in \mathbb{R}$ or $\mathcal{S}_p^1$. This synthesis is used in the cases of $p = 2,\ \infty$: scaled H_∞ and scaled L_1 control. In the case of $p = 2$, $d_i \in \mathbb{R}$, the robustness of such scaled H_∞ control is based on Theorem 3 for NLTV structured uncertainties and in the case of $d_i \in \mathcal{S}_2^1$, it is based on Theorem 2, with μ been replaced with its upper bound, for LTI structured uncertainties. Unlike μ-analysis, the theoretical basis for μ-synthesis is much weaker. D-K iteration algorithm is a heuristic algorithm proposed by Doyle [27] to tackle the optimization problem approximately. The D-K iteration involves a sequence of minimizations: first over the controller K holding D fixed, which is an H_∞ control problem, second over scaling D holding K fixed, which is a convex optimization problem. However, the D-K iteration is not guaranteed to converge to the minimum. This D-K iteration method is also used for scaled L_1 control [19]. We will discuss the D-K iteration further in Chapter 6.

The H_2/H_∞ synthesis problem is a typical mixed norm structured performance synthesis problem, where an H_2 norm requirement is imposed corresponding to performance specifications, while an H_∞ norm requirement ensures robustness. Again the problem is not convex and in general there is no analytic solution. We refers to [17] and the references therein for details on the problem setup and some possible solutions.

Unstructured performance and structured controller

Most of the controllers in classical control (1930-1960), like PID and Lead-Lag compensation, are in this category. The controller structure is fixed independently of the controlled plant. The structure is often very simple. The control synthesis merely consists of tuning the parameters in the controller while holding the structure fixed. There are many heuristic and trial-and-error methods to tune such controllers and they have often been proved effective in practice. Theoretically, however, it is misleading to think that finding an optimal controller with simple but fixed structure is easy. Actually, it has already been proved that lower order H_∞ synthesis can be cast as LMI minimum rank problem [32], which is shown to be NP hard [17]. To find an optimal structured controller, we have to do the so called "brute force" search in the controller parameter space or branch and bound search [6] if some upper bound and lower bound of a box in parameter space can be calculated.

Structured performance and structured controller

Obviously this is the most difficult category of control synthesis since both performance and controller are structured. There is not much theoretical research work on this subject so far. Actually "brute force" search or branch and bound search (if possible), combining μ-analysis methods, are the only ways to tackle this optimal control synthesis problem. If we only try to find a controller that satisfies the performance specifications instead of an optimal controller, there can be some heuristic methods. We shall discuss a PID control design for a system with structured uncertainties in Chapter 7.

1.3 Survey and contributions

In this section we give a chapter by chapter survey and a summary of main contributions of this dissertation.

1.3.1 Chapter by chapter survey

The dissertation consists of four parts. Fig. 1.13 shows the relations between chapters. Chapter 2 and Chapter 3 of the first part study the

robustness analysis, the focus is on μ calculation. Chapter 4 and Chapter 5 of the second part study uncertainty modeling problems. Chapter 6 studies robust controller design problems, in which the focus will be on the D-K iteration with mixed D-scaling. Chapter 7 presents the application of theories and algorithms developed in this dissertation to an hydraulic servo system of a continuously variable transmission. The concluding remarks and potential further research work are given in the last chapter.

Robustness analysis

In Chapter 2, we focus on the computation of lower bounds on the complex μ. The structured singular value μ plays a fundamental role in robustness analysis for control systems with structured uncertainties. It is, however, difficult to calculate. Thus only lower and upper bounds are calculated. The upper bound problem is a convex optimization problem and can be solved easily. The lower bound problem is not convex. The existing power iteration algorithm sometimes suffers from limit cycling. We first present a continuous time power iteration algorithm based on the existing power iteration algorithm. The convergence property is improved numerically but not guaranteed theoretically. Then we describe two gradient based optimization algorithms: steepest ascent and conjugate gradient algorithms. These algorithms have guaranteed convergence properties.

In Chapter 3, the real parametric μ is studied. It is shown that the parameter stability margin problem can be cast as the real parametric μ problem. The study focuses on the calculation of a lower bound of μ_2 or the inverse of l_2 parameter margin, which turns out to be a structured total least squares problem. A new algorithm is proposed to solve the lower bound problem. The algorithm is an iterative process based on a variant of the power iteration. In most of the numerical examples, the algorithm shows the good convergence.

Uncertainty modeling

In Chapter 4, it is shown that the parametric uncertainty modeling problem can be cast as the realization of a multidimensional system, which is defined in the framework of Linear Fractional Transformations (LFTs).

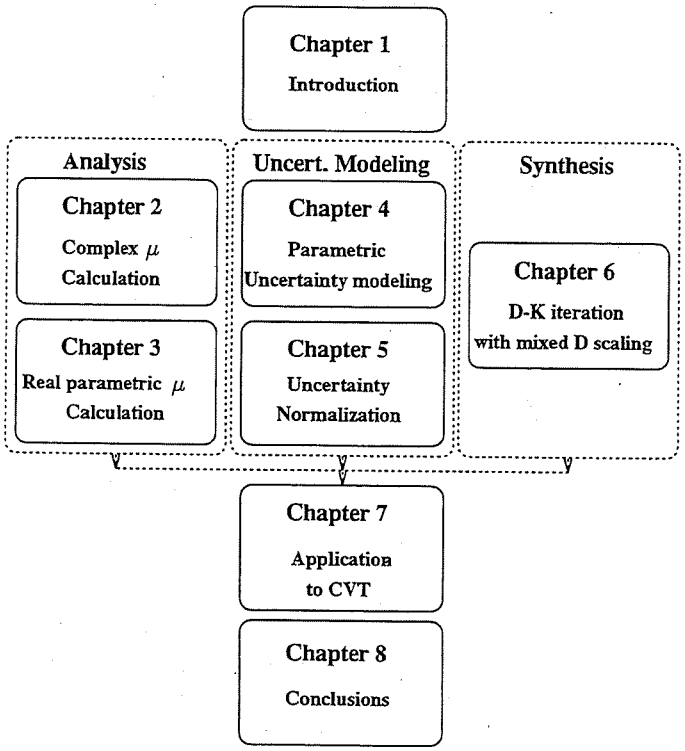


Figure 1.13 A schematic overview of the dissertation.

An algorithm to systematically solve the multidimensional realization problem is given. The trick we used is to realize the affine part of a rational matrix at first, then the polynomial part in a second step. In addition two size reduction procedures are introduced. The first one is based on a minimal realization method for one dimensional systems and can be used for size reduction of general multidimensional realization, while the second one, which is used in each of the two realization steps, is aimed to reduce the size in the problem of $\det(I + M\Delta) = 0$.

In Chapter 5, we introduce a method for normalizing structured uncertainties in general sectors to a unit ball (non-expansive). The significance of the results is that we can use a systematic way to normalize structured uncertainties, which in practice are often not non-expansive or even not norm bounded but belong to some general sectors, and that it is easy to code a computer program to implement the method based on all the formulae in the chapter. The method used here is based on sector transformations and loop transformations.

Controller synthesis

In Chapter 6, different D-scaling methods are discussed for robustness analysis for different uncertainty structures. A D-K iteration procedure with frequency and constant mixed D-scaling is introduced. This procedure can be used to synthesize robust control for systems with both linear and nonlinear uncertainties. An industrially motivated example shows that the result with this D-K iteration is not so conservative as with the D-K iteration with constant D-scaling.

Application

In Chapter 7, it is shown how the design and analysis methods developed in previous chapters can be applied to the control system design of a hydraulic servo system of a CVT. First, we use uncertainty modeling techniques developed in Chapter 4 and 5 to separate out uncertainties with nominal plant and to normalize all uncertainty blocks to be non-expansive. Secondly several methods are used for design and analysis of robust control systems. The results show that the classic gain and phase margin test and PID control design can not guarantee robust performance, while H_∞ control design fails to get a useful controller. Both

standard D-K iteration and D-K iteration with mixed D scaling successfully give controllers which ensure both stability and performance robustness, and in doing so, an order reduction procedure is also introduced. By locally searching in the PID parameter space, a PID controller was found such that closed-loop robustness is guaranteed. Actually it is the best controller for our practical application because of its simple structure. Finally both time responses and frequency responses are calculated. The time responses are simulated on the original nonlinear hydraulic closed loop system, while frequency responses are calculated on the nominal closed loop system where all uncertainties are omitted. Both time responses and frequency responses confirm the results from robustness analysis.

Chapter 8 includes some concluding remarks and suggestions for further research.

1.3.2 Main contributions

Here we give a summary on the main contributions of this dissertation. Some of the results have already been published in journals and proceedings of conferences.

- Algorithms for lower bounds on complex μ: First, a continuous time power iteration algorithm is developed for calculating a lower bound of the structured singular value μ based on an optimality condition of μ [93]. Then two gradient based optimization algorithms (steepest ascent and conjugate gradient) are developed, convergence of which can be guaranteed [96].

- Algorithms for lower bounds of real parametric μ: An algorithm is developed for calculating a lower bound of μ_2, the inverse of l_2 parameter margin, based on the STLS (structured total least squares) method.

- Parametric uncertainty modeling: LFTs are used for representing uncertainty systems. A multidimensional realization method is developed for parametric uncertainty modeling [92, 95].

- Uncertainty normalization: It is shown that structured uncertainties can be normalized systematically by LFTs based on sector and

loop transformation techniques, such that the normalized structured uncertainties have all non-expansive uncertainty blocks, which is necessary for robustness analysis and controller design [98].

- D-K iteration with mixed D-scaling: An algorithm is developed for D-K iteration with constant and frequency-dependent scaling D, which is used in robust control synthesis when the uncertainty contains both linear and nonlinear blocks [97].

- An industrial application: It is shown how the whole theory we have developed is applied for robust analysis and control system design for a hydraulic servo system in a continuous variable transmission [94, 91].

Chapter 2

Algorithms for calculation of lower bounds of μ

As we can see from the previous chapter, the structured singular value μ plays a fundamental role in the robustness analysis for control systems with structured and bounded (respect to the H_∞ norm) uncertainties. Much of the subsequent research on μ has focused on computational schemes, which actually determine the usefulness of μ for practical purposes. Up to now, the most general description of μ is the so-called mixed μ description, which includes repeated real scalar blocks, repeated complex scalar blocks and full blocks [34]. However, exact calculation of the mixed μ was proved to be NP hard [14]. The fact that the mixed μ problem is NP hard means that given any algorithm to compute μ, there will be specific numerical examples for which the algorithm cannot find the answer in polynomial time. Moreover the mixed μ problem is not necessarily continuous in the problem data. The complex μ, which is less general than the mixed μ, includes repeated complex scalar and complex full blocks. It is continuous in the problem data (actually it is a C^1 function of the matrix almost everywhere, see section 2.3.2), and in some cases (depending on the block structures) the complex μ problems are convex [62]. But in general, the complex μ problems are non-convex. So all computationally attractive approaches to solve μ problems are basically inexact: they calculate upper and lower bounds. In [26], Doyle introduced an upper bound for the complex μ, which was then generalized for the mixed μ in [34]. This upper bound problem is convex, thus it

can be solved efficiently as mentioned in Appendix A. It is however not a tight bound in general. On the other hand, Doyle also introduced a lower bound in [26], which was generalized for the mixed μ in [99]. Though this lower bound is actually tight if the global maximum can be found, unfortunately, the lower bound problem is not a convex one. Packard and Doyle [61] derived a so-called power iteration algorithm to calculate a lower bound for the complex μ, which was generalized for the mixed μ in [99]. However it sometimes suffers from limit cycling.

In this chapter, we focus on the computation of lower bounds of complex μ. After giving some preliminaries on μ and optimization theory in section 2.1, we present a continuous time version of the power iteration algorithm in section 2.2. In section 2.3, geometric optimization algorithms: steepest ascent and conjugate gradient, are given for the calculation of μ. Finally, some concluding remarks are made in section 2.4.

2.1 Preliminaries

In this section, some preliminary results on μ and its lower and upper bounds are given without proofs. We refer the readers to a paper [62] by Packard and Doyle for all details. We also give a brief survey on optimization theory in the end of this section.

2.1.1 Definition

The structured singular value, or μ, is defined as a matrix function $\mu : \mathbb{C}^{n\times n} \to \mathbb{R}_+$. In the definition of μ, there is an underlying block structure $\mathbf{\Delta}$ described by a set of diagonal matrices[1]. There are three things to define such a structure: the type of each block, their dimensions and the total number of the blocks. For complex μ, two types of blocks are considered: repeated scalar blocks and full blocks and correspondingly, two nonnegative integers s and f used to denote the number of the scalar repeated and full blocks respectively. For the dimensions of the blocks, two positive integer vectors are defined: $r = [\, r_1 \cdots r_s \,]$ and $m = [\, m_1 \cdots m_f \,]$. The ith repeated scalar block is of dimension $r_k \times r_k$

[1]Note that we here abuse the notation of $\mathbf{\Delta}$. In chapter 1, we have used $\mathbf{\Delta}$ as a structured perturbation functional set, while here it is a structured matrix set. Hereafter, we assume that the meaning of $\mathbf{\Delta}$ follows from its context.

and the lth full block is of dimension $m_l \times m_l$. Now the definition of the block structure is given as:

$$\begin{aligned}\mathbf{\Delta} \triangleq \{&\mathbf{diag}\left[\delta_1 I_{r_1}, \cdots, \delta_s I_{r_s}, \Delta_1, \cdots, \Delta_f\right] : \\ &\delta_k \in \mathbb{C}, \Delta_l \in \mathbb{C}^{m_l \times m_l}, 1 \le k \le s, 1 \le l \le f\}.\end{aligned} \tag{2.1}$$

Obviously $\sum_{k=1}^{s} r_i + \sum_{l=1}^{f} m_j = n$ has to be satisfied. Note that the block structure can be represented alternatively by integer vectors r and m. Now the definition of μ is given as [62]:

Definition 9 *For $M \in \mathbb{C}^{n\times n}$, $\mu(M)$ is defined*

$$\mu(M) \triangleq \frac{1}{\min\{\bar{\sigma}(\Delta) : \Delta \in \mathbf{\Delta}, \det(I + M\Delta) = 0\}}, \tag{2.2}$$

unless no $\Delta \in \mathbf{\Delta}$ makes $I + M\Delta$ singular, then $\mu(M) = 0$.

Obviously, $\mu(M)$ depends on the matrix M as well as the block structure $\mathbf{\Delta}$.

2.1.2 Lower and upper bounds

The structured singular value μ can be related to familiar linear algebra quantities: the spectral radius ρ and the maximum singular value $\bar{\sigma}$, depending on the block structure $\mathbf{\Delta}$, as follows:

- if $\mathbf{\Delta} = \delta I : \delta \in \mathbb{C}$ $(s = 1, f = 0, r_1 = n)$ then $\mu(M) = \rho(M)$, the spectral radius of M.
- if $\mathbf{\Delta} = \mathbb{C}^{n\times n}$ $(s = 0, f = 1, m_1 = n)$, then $\mu(M) = \bar{\sigma}(M)$.
- For general $\mathbf{\Delta}$ as in (2.1),

$$\rho(M) \le \mu(M) \le \bar{\sigma}(M). \tag{2.3}$$

These results can be proved directly from the definition of μ. From (2.3), obviously, we can take $\rho(M)$ and $\bar{\sigma}(M)$ as lower and upper bounds of $\mu(M)$ respectively. However, these bounds themselves may give little information on the value of μ, because the gap between ρ and $\bar{\sigma}$ can be arbitrarily large. To refine these bounds, the matrix M is transformed

such that the transformations do not affect μ, but do affect ρ and $\bar{\sigma}$. To do these, define two matrix sets:

$$\mathbf{Q} \triangleq \left\{Q \in \mathbf{\Delta} : Q^H Q = I_n\right\}. \tag{2.4}$$

$$\begin{aligned}\mathbf{D} \triangleq \Big\{&\mathbf{diag}\left[D_1, \cdots, D_s, d_1 I_{m_1}, \cdots, d_f I_{m_f}\right] : \\ &D_k \in \mathbb{C}^{r_k \times r_k}, D_k = D_k^H > 0, d_l > 0\Big\}.\end{aligned} \tag{2.5}$$

$\mathbf{Q}$ is the unitary matrix set in $\mathbf{\Delta}$ so $Q\Delta \in \Delta$ for all $Q \in \mathbf{Q}$, $\Delta \in \mathbf{\Delta}$. $\mathbf{D}$ is a matrix set with such a structure that D commutes with Δ, that is $\Delta D = D\Delta$, for all $D \in \mathbf{D}$, $\Delta \in \mathbf{\Delta}$. Then for all $Q \in \mathbf{Q}$ and $D \in \mathbf{D}$,

$$\mu(MQ) = \mu(MQ) = \mu(M) = \mu(DMD^{-1}). \tag{2.6}$$

Therefore the bounds in (2.3) can be tightened via (2.6) to

$$\max_{Q\in\mathbf{Q}} \rho(QM) \le \mu(M) \le \inf_{D\in\mathbf{D}} \bar{\sigma}(DMD^{-1}). \tag{2.7}$$

The upper bound problem of the right side of (2.7) is actually an LMI optimization problem, since

$$\inf_{D\in\mathbf{D}} \bar{\sigma}(DMD^{-1}) = \inf_{D\in\mathbf{D}} \beta, \quad \text{subject to } \bar{\sigma}(D^{1/2}MD^{-1/2}) \le \beta,$$

and

$$\bar{\sigma}(D^{1/2}MD^{-1/2}) \le \beta \Leftrightarrow M^H DM - \beta^2 D < 0.$$

where the last inequality is an LMI. Therefore the upper bound problem can be efficiently solved by LMI optimization algorithms discussed in Appendix A. However the upper bound is not tight (the right inequality of (2.7) can not be an equality) in general. Table 2.1 summarizes the tightness of the upper bounds. Thus, it is necessary to compute the lower bound of the left side of (2.7). Though the lower bound is actually a tight bound (the left side inequality of (2.7) is also an equality) if one can find a global maximum [26], it is not a convex optimization problem. The function $\rho(QM)$ has local maxima which are not global. One can only get a lower bound from the left side of (2.7) in general.

Table 2.1 Guaranteed Equality between μ and the upper bounds.

	f=0	f=1	f=2	f=3	f=4
s=0		Yes	Yes	Yes	No
s=1	Yes	Yes	No	No	No
s=2	No	No	No	No	No

2.1.3 Non convex optimization

A general setup for optimization problem is expressed as

$$\min_{x \in \mathbf{S}} f(x), \tag{2.8}$$

where $\mathbf{S}$ is a set, $f(x) : \mathbf{S} \to \mathbb{R}$ is an objective function. To solve a non-convex optimization problem (see Appendix A for convex optimization), one could either concentrate on the necessary and/or sufficient *optimality conditions* for the given problem and if the *optimality conditions* cannot be solved analytically, develop an iterative scheme to solve them; or one could directly apply *general optimization algorithms* using function evaluations and (possibly) gradient information.

Optimality conditions are a set of equations derived from the given problem. Any solution to the set of equations is a local optimal point for the given optimization problem. *Optimality conditions* can be found via different ways, *e.g.*, the Lagrange method. However, solving the set of equations of the *optimality conditions* is not easy. The power iteration algorithm is often used to solve the *optimality conditions* when no analytical solution can be found. We will use the power iteration algorithms to solve the *optimality conditions* for the complex μ calculation in this chapter and for structured total least square (STLS) in next chapter.

For *general optimization algorithms*, geometric optimization algorithms are often used. A gradient based geometric optimization algorithm mainly consists of the following two steps in each iteration:

- Find a direction at point x_k, say s_k.
- Solve a one dimensional optimization problem, called *line search*, to find a point x_{k+1} that minimizes $f(x)$ along a line in $\mathbf{S}$ in the direction of s_k.

The simplest structure of the set $\mathbf{S}$ is a Euclidean space $\mathbb{R}^n$ endowed with the natural metric defined by the normal vector inner product. The problem in this case is just an unconstrained optimization problem. A direction is just a vector in $\mathbb{R}^n$, a line is just a straight line in $\mathbb{R}^n$. The way to choose a direction classifies the ways of optimization such as the steepest descent direction, the Newton direction and the conjugate gradient direction. We refer to [35] for details on these methods.

Recently these optimization algorithms over Euclidean spaces have been generalized to Riemannian manifolds [79]. The constraint set $\mathbf{S}$ is now a Riemannian manifold which is a differential manifold endowed with a Riemannian metric. A direction s_k at x_k in this case is a vector on the tangent space of the manifold $\mathbf{S}$ at x_k (note that the tangent space of $\mathbb{R}^n$ is coincides with itself at any point). A "line" in a manifold now is a geodesic, the projection of which onto the tangent space at any point x is of zero curvature at that point (this is actually a generalization of a straight line in Euclidean space: a straight line is a curve with zero curvature at any point). One can now equally consider a steepest descent method and a conjugate gradient method corresponding to the algorithms for Euclidean spaces. However, the calculations of the directions and geodesics are more complicated than those for Euclidean spaces. For computation of directions (steepest descent and conjugate gradient), the gradient of $f(x)$ and the tangent vector parallel translation (not for the steepest descent method) have to be calculated. For an n dimensional manifold, the computation for a parallel translation and a geodesic requires the solution of a system of $2n$ nonlinear and n linear ordinary differential equations. Nevertheless, many optimizations are posed on the manifold that have an underlying structure that may be exploited to greatly reduce the complexity of these computations. Formulae for geodesics and parallel translations of tangent vectors along geodesics for some special Riemannian manifolds are summarized in [79]. Fortunately, our optimization problem for calculation of complex μ can use those formulae with little modifications. We will pursue this further later in this chapter.

2.2 Continuous time power iteration

2.2.1 Optimality condition

The power iteration algorithm [61] is actually an iterative scheme to solve the optimality condition for the optimization problem:

$$\max_{Q\in\mathbf{Q}} \rho(QM). \tag{2.9}$$

The optimality condition for (2.9) is based on the following two theorems [61].

Let x and y be the right and left eigenvectors of the matrix QM. Partition x and y compatible with the block structure:

$$x = \begin{bmatrix} x_{r_1} \\ \vdots \\ x_{r_s} \\ x_{m_1} \\ \vdots \\ x_{m_f} \end{bmatrix}, \; y = \begin{bmatrix} y_{r_1} \\ \vdots \\ y_{r_s} \\ y_{m_1} \\ \vdots \\ y_{m_f} \end{bmatrix},$$

where the block component $x_{r_k}, y_{m_k} \in \mathbb{C}^{r_k}$ and $x_{m_l}, y_{m_l} \in \mathbb{C}^{m_l}$ for each k and l. The x and y are said to be non-degenerate if for every k, $y_{r_k}^H x_{r_k} \neq 0$ and for each l, $x_{m_l} \neq 0, y_{m_l} \neq 0$. The following theorem gives a necessary condition for (2.9) to have a global maximum:

Theorem 6 *Let $Q_0 \in \mathbf{Q}$ achieve the global optimum for (2.9) and suppose that the eigenvalue associated with $\rho(Q_0M)$ is distinct, real and positive, and call it μ. If x and y are the right and left eigenvectors of this eigenvalue, and their block components satisfy the non-degeneracy assumption, then there exists a $D \in \mathbf{D}$ such that:*

$$Q_0DMD^{-1}(Dx) = \mu Dx, \quad x^HDQ_0DMD^{-1} = \mu x^HD. \tag{2.10}$$

Proof: see [61].

Now the problem is to find $Q \in \mathbf{Q}, D \in \mathbf{D}, \beta > 0$ and x with $\|x\| = 1$ such that:

$$QDMD^{-1}(Dx) = \beta Dx, \quad x^HDQDMD^{-1} = \beta x^HD. \tag{2.11}$$

The following theorem eliminates Q and D.

Theorem 7 *Let $M \in \mathbb{C}^{n\times n}$ be given, and let Δ be a two block structure of $s = 1$ and $f = 1$ with block sizes r_1 and m_1, where $r_1+m_1 = n$, Suppose $\beta > 0$ is given, then there exist $Q \in \mathbf{Q}$, $D \in \mathbf{D}$, $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathbb{C}^n$, such that (2.11) holds iff there exist nonzero vectors z_1^0, w_1^0, b_1^0, $a_1^0 \in \mathbb{C}^{r_1}$ and z_2^0, w_2^0, b_2^0, $a_2^0 \in \mathbb{C}^{m_1}$ with $(a_1^0)^Hw_1^0 \neq 0$ and*

$$\begin{aligned} &\beta a^0 = Mb^0, \quad z_1^0 = \tfrac{(w_1^0)^Ha_1^0}{|(w_1^0)^Ha_1^0|}w_1^0, \quad z_2^0 = \tfrac{\|w_2^0\|}{\|a_2^0\|}a_2^0, \\ &\beta w^0 = M^Hz^0, \quad b_1^0 = \tfrac{(a_1^0)^Hw_1^0}{|(a_1^0)^Hw_1^0|}a_1^0, \quad b_2^0 = \tfrac{\|a_2^0\|}{\|w_2^0\|}w_2^0. \end{aligned} \tag{2.12}$$

Proof: see [61].

Note that a^0 and w^0 defined as

$$a^0 = \begin{bmatrix} a_1^0 \\ a_2^0 \end{bmatrix}, \; w^0 = \begin{bmatrix} w_1^0 \\ w_2^0 \end{bmatrix},$$

are the right eigenvector of the matrix MQ and the left eigenvector of the matrix QM associated with the eigenvalue β. The restriction to the two block structure with $s = 1$ and $f = 1$ is just for simplicity of the formulas. It can be easily relaxed.

In [61] (2.12) is solved in an iterative fashion, the so-called *power iteration*, denoted by PIA, in which vectors a_{i_k}, b_{i_k}, z_{i_k} and w_{i_k} $(i = 1, 2)$, and positive scalars $\tilde{\beta}_k$ and $\hat{\beta}_k$ evolve as:

Algorithm 1 (Power Iteration Algorithm)

1. *initialize vectors b_0 and w_0*
2. *k=0*
3. **repeat**
4. $\tilde{\beta}_{k+1}a_{k+1} = Mb_k$ with $\|a_{k+1}\| = 1$
   $z_{1_{k+1}} = \frac{w_{1_k}^Ha_{1_{k+1}}}{|w_{1_k}^Ha_{1_{k+1}}|}w_{1_k}$, $z_{2_{k+1}} = \frac{\|w_{2_k}\|}{\|a_{2_{k+1}}\|}a_{2_{k+1}}$
   $\hat{\beta}_{k+1}w_{k+1} = M^Hz_{k+1}$ with $\|w_{k+1}\| = 1$
   $b_{1_{k+1}} = \frac{a_{1_{k+1}}^Hw_{1_{k+1}}}{|a_{1_{k+1}}^Hw_{1_{k+1}}|}a_{1_{k+1}}$, $b_{2_{k+1}} = \frac{\|a_{2_{k+1}}\|}{\|w_{2_{k+1}}\|}w_{2_{k+1}}$
5. *k=k+1*
6. **until** *convergence or a specified maximum number of steps is reached.*

where $\tilde{\beta}_k$ and $\hat{\beta}_k$ are chosen > 0, so that $\|a_{k+1}\| = \|w_{k+1}\| = 1$.

2.2.2 Continuous time power iteration algorithm

Now by eliminating vectors z_0 and b_0 and separating real and imaginary parts, we can rewrite (2.12) as:

$$AD(x^0)x^0 = \beta x^0, \tag{2.13}$$

where the vector $x^0 \in \mathbb{R}^{4n}$ and the matrix $A \in \mathbb{R}^{4n\times 4n}$ are defined as

$$x_0 \triangleq \begin{bmatrix} a_{1r}^0 \\ a_{2r}^0 \\ w_{1r}^0 \\ w_{2r}^0 \\ a_{1i}^0 \\ a_{2i}^0 \\ w_{1i}^0 \\ w_{2i}^0 \end{bmatrix}, \quad A \triangleq \begin{bmatrix} 0 & M_r & 0 & -M_i \\ M_r^t & 0 & M_i^t & 0 \\ 0 & M_i & 0 & M_r \\ -M_i^t & 0 & M_r^t & 0 \end{bmatrix}, \tag{2.14}$$

with $a_{1r}^0 = \text{real}(a_1^0)$, $a_{2r}^0 = \text{real}(a_2^0)$, $w_{1r}^0 = \text{real}(w_1^0)$, $w_{2r}^0 = \text{real}(w_2^0)$, $a_{1i}^0 = \text{imag}(a_1^0)$, $a_{2i}^0 = \text{imag}(a_2^0)$, $w_{1i}^0 = \text{imag}(w_1^0)$, $w_{2i}^0 = \text{imag}(w_2^0)$, $M_r = \text{real}(M)$ and $M_i = \text{imag}(M)$, and the matrix function $D(x_0) \in \mathbb{R}^{4n\times 4n}$ is defined as:

$$D(x^0) \triangleq \begin{bmatrix} 0 & 0 & d_r^r(x_0)I_r & 0 \\ 0 & d_f(x_0)I_m & 0 & 0 \\ d_r^r(x_0)I_r & 0 & 0 & 0 \\ 0 & 0 & 0 & d_f^{-1}(x_0)I_m \\ 0 & 0 & d_r^i(x_0)I_r & 0 \\ 0 & 0 & 0 & 0 \\ -d_r^i(x_0)I_r & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & -d_r^i(x_0)I_r & 0 \\ 0 & 0 & 0 & 0 \\ d_r^i(x_0)I_r & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & d_r^r(x_0)I_r & 0 \\ 0 & d_f(x_0)I_m & 0 & 0 \\ d_r^r(x_0)I_r & 0 & 0 & 0 \\ 0 & 0 & 0 & d_f^{-1}(x_0)I_m \end{bmatrix}, \tag{2.15}$$

with $d_r^r(x^0) \triangleq \text{real}(d_r(x^0))$, $d_r^i(x^0) \triangleq \text{imag}(d_r(x^0))$, and

$$d_r(x^0) \triangleq \frac{(w_1^0)^H a_1^0}{|(w_1^0)^H a_1^0|}, \quad d_f(x^0) \triangleq \frac{\|w_2^0\|}{\|a_2^0\|}. \tag{2.16}$$

Note that matrices A and $D(x_0)$ are both symmetric and the matrix $D(x_0)$ here contains all the structure information. This formulation can be easily generalized to the cases of $s > 1$ and $f > 1$ by modifying the matrix function $D(x^0)$. From (2.13), β is an eigenvalue of the matrix $AD(x^0)$ and x^0 is the eigenvector associated with β.

The famous power iteration algorithm [86, 39] is an iterative scheme to calculate the spectral radius of a matrix, say, $B \in \mathbb{R}^{n\times n}$ and the eigenvector associated with the eigenvalue of the spectral radius. A continuous time realization of the power iteration can be described by the following differential system:

$$\dot{x} = Bx - x\tau, \ \tau = x^H Bx, \tag{2.17}$$

where $x \in \mathbb{R}^n$. Suppose that B has a real eigenvalue ρ_r that is no less than the real part of any other eigenvalues. Then x in (2.17) will converge to the eigenvector associated with the eigenvalue ρ_r. We refer to [15, 43] for all details on this differential system (flow).

Now we do the same thing for our "eigenvalue problem" in (2.13). To do that, define the following state vectors a_{1r}, w_{1r}, a_{1i}, $w_{1i} \in \mathbb{R}^r$, and a_{2r}, w_{2r}, a_{2i}, $w_{2i} \in \mathbb{R}^m$, and $a_1 := a_{1r} + ja_{1i}$, $a_2 := a_{2r} + ja_{2i}$, $w_1 := w_{1r} + jw_{1i}$ and $w_2 := w_{2r} + jw_{2i}$. Finally we define another state vector x:

$$x \triangleq \begin{bmatrix} a_{1r} \\ a_{2r} \\ w_{1r} \\ w_{2r} \\ a_{1i} \\ a_{2i} \\ w_{1i} \\ w_{2i} \end{bmatrix}. \tag{2.18}$$

By replacing B with $AD(x)$, the continuous time power iteration, denoted by CPIA, for our "eigenvalue" problem of (2.13) is now given as the following differential equations:

$$\boxed{\dot{x} = AD(x)x - \tau(x)x, \ \tau(x) = x^T AD(x)x,} \tag{2.19}$$

where matrix A and matrix function $D(x)$ are the same as defined in (2.14) and (2.15) . If we find an equilibrium point of (2.19), x^0, with

$\|x^0\| = 1$, then (2.19) becomes (2.13), with $\beta = (x^0)^T AD(x^0)x^0$, and thus we find a lower bound on the structured singular value of the matrix M.

One property of the differential equation of (2.19) is that vector x has a constant norm of 1 if the initial value of the norm is 1. We can see this by taking the derivative of $\|x\|^2$:

$$\frac{d\|x\|^2}{dt} = 2x^t\dot{x} = 2x^t(I - xx^t)AD(x)x = 2(1 - x^tx)x^tAD(x).$$

So we have

$$\left.\frac{d\|x\|^2}{dt}\right|_{\|x\|=1} = 0.$$

Thus if we take the initial value $\|x(0)\| = 1$, $\|x\|$ is constant and equal to 1. Actually we can see this property by another way. Suppose vector $y(t)$ satisfies the following differential equation:

$$\frac{dy}{dt} = AD(y)y. \tag{2.20}$$

Then $x = y/\|y\|$ satisfies the differential equation (2.19), for

$$\frac{dx}{dt} = \frac{d}{dt}\frac{y}{\sqrt{y^ty}} = \frac{AD(y)y}{\|y\|} - \frac{yy^tAD(y)y}{\|y\|^3} = AD(x)x - xx^tAD(x)x.$$

Here we use the equality $D(y/\|y\|) = D(y)$. So the norm of x is constant and equal to 1. In all of the analysis, we assume $\|x\| = 1$. Further we also assume that $D(x)$ is well defined, by which we mean that a_1, a_2, w_1, w_2 are nonzero vectors and $w_1^*a_1 \neq 0$ (non-degeneracy assumption).

Now we can see that a power iteration can be formed by discretizing the differential equation of (2.19) plus normalizing the norm of the state vector in each step. Let (2.19) be discretized as:

$$\frac{x_{k+1} - x_k}{\delta t_k} = AD(x_k)x_k - x_k(x_k^tAD(x_k)x_k), \tag{2.21}$$

where δt_k is the integration step at time t_k and x_k is the vector x at time t_k. Suppose $x_k^tAD(x_k)x_k > 0$ and let $\Delta t_k = 1/(x_k^tAD(x_k)x_k)$, then by normalizing the norm of x_{k+1} to 1, (2.21) can be arranged as

$$x_{k+1} = \frac{AD(x_k)x_k}{\|AD(x_k)x_k\|}. \tag{2.22}$$

The power iteration of (2.22) is actually the same as PIA (step 4), if some intermediate results in each iteration are not used and if $\beta_{k+1} = \|AD(x_k)x_k\|$. Note that if $x_k^tAD(x_k)x_k < 0$ the above explanation does not hold. However it can be proved that if the continuous time power iteration is convergent, $x^tAD(x)x > 0$ always near an equilibrium point. Though numerical simulation of the differential equation of (2.19) is also discrete, it is different from PIA, as the former can choose integration steps according to the desired accuracy by, for example, automatic step Runge-Kutta integration, the latter only uses a special integration law of (2.21), for which the integration steps can be possibly very large. One result is that limit cycling which can occur in the discrete time power iteration might be eliminated in the continuous time power iteration, which will be shown by an example later.

2.2.3 Numerical examples

We have simulated the continuous time dynamic system of (2.19) using MATLAB function 'ode23', which uses the automatic step Runge-Kutta-Fehlberg integration method. The desired accuracy of the solution is set to 10^{-5}. For most of the examples we have tested with nonzero initial vectors a_r, a_i, w_r and w_i, the simulations were convergent, by which we mean that both the state vector x and the scalar β evolve to constant values, even in the case where PIA had limit cycling. Here is one of the examples.

The matrix M is a 3×3 random matrix, generated by MATLAB 'rand' function with format 'normal', as:

$$M = \begin{bmatrix} 1.0308 & 0.7611 & -0.3225 \\ -0.7599 & -0.1659 & -0.3684 \\ 0.8741 & 0.3009 & 1.1479 \end{bmatrix}.$$

It has eigenvalues $0.8678 \pm 0.7474i$ (corresponding to a spectral radius $\rho(M) = 1.1453$) and 0.2773. The block structure consists of one repeated scalar block ($s = 1$) with dimension $r_1 = 2$ and one full block ($f = 1$) with dimension $m_1 = 1$. We first use PIA to calculate a lower bound of μ with the MATLAB function 'mu' in μ toolbox [4]. The initial values of the vectors b and w are equal and generated randomly as: $-0.7896 - 0.0098i$, $0.5648 - 0.0515i$, $0.0321 - 0.2317i$. For this special case, limit cycling

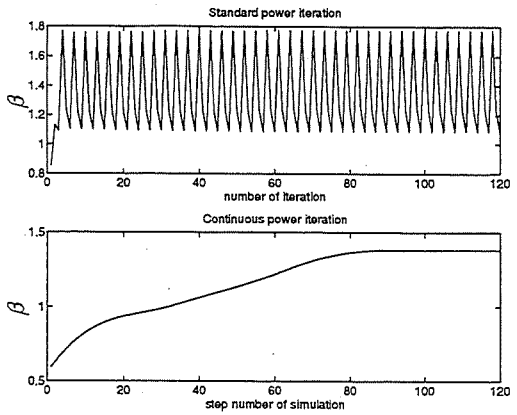


Figure 2.1 Upper plot: Power iteration (using the MATLAB function 'mu' in μ-toolbox [4]). Limit cycling occurs in the iterations. Bottom plot: Continuous power method. β evolves rather smoothly to an equilibrium point 1.384 in 102 steps.

appears during the iterations, as shown in the upper plot of Fig. 2.1. Real matrices with complex eigenvalues, often give rise to a similar situation. It should be emphasized that these examples are relevant for practical applications. Now we calculate β by solving the differential equation (2.19) with the MATLAB function 'ode23'. The matrix M is exactly the same as above. The initial value of the vector x is calculated from initial values of the vectors b and w via (2.19). That is, we use the same initial values as those for the power iteration as shown above. The bottom plot of Fig. 2.1 shows the simulation result. β now evolves rather smoothly and converges to an equilibrium value 1.384 in 102 steps (the number of all forward and backward steps is 306).

2.3 Geometric optimization algorithms

Though the power iteration algorithm (PIA) computes a lower bound for μ and is computationally efficient if it converges, up to this date, there is no formal theory of convergence. As a matter of fact, current implementations do not necessarily converge as has been demonstrated with an example shown above already. The continuous time counter part of PIA, CPIA, seems to have better convergence properties but also lacks a convergence proof.

Now we choose the approach of a *general optimization algorithm*. Actually, such an approach may be regarded as just another iterative algorithm to find a point that satisfies the equilibrium conditions, but now one with guaranteed convergence. The prize for this guaranteed convergence, may be the loss of computational efficiency. However, the computational efficiency of the algorithms presented below, is still acceptable, certainly when one takes into account that the structured singular value problem is mainly a problem of system analysis for which efficiency is not the main issue. Moreover combination of the heuristic PIA with our algorithms, may lead to algorithms that are more useful than both of them separately. Our approach also takes into account the highly regular nature of the constraint in the optimization problem (2.9), by applying optimization methods on Riemannian manifolds, making use of intrinsic properties of the manifold. The discussion whether the value of these differential geometric concepts merely lies in theoretical elegance and new theoretical insight or also in improved quality of the algorithm, has not been conclusively decided yet. The present result at least proves that the approach can provide simple solutions to seemingly difficult problems.

2.3.1 Geometric properties of structured unitary matrices

The optimization methods discussed by Smith [79] generalize optimization methods over Euclidean spaces to optimization methods over Riemannian manifolds by making use of intrinsic properties of the manifold. Straight lines are replaced by geodesics. Tangent vectors at one point are adapted to other points by parallel translations along geodesics. In this section we establish some properties of the manifold $\mathbf{Q}$ of structured unitary matrices and define a bi-invariant Riemannian metric on $\mathbf{Q}$.

From the definition:

$$\mathbf{Q} = \{Q \in \Delta : Q^H Q = I_n\} = U(n) \cap \Delta$$

where $U(n)$ is the unitary group, $\mathbf{Q}$ is a continuous manifold (see [96] for details) with the dimension $s + \sum_{i=1}^{f} m_i^2$, where s and f are the number of the repeated scalar blocks and the number of the full blocks, m_l is the corresponding full block dimension. Let $T_Q\mathbf{Q}$ be the tangent space of $\mathbf{Q}$ at Q. The tangent space $T_Q\mathbf{Q}$ is formed by tangent vectors of all curves on $\mathbf{Q}$ passing through Q. Let $Q(t)$ be any curve on $\mathbf{Q}$ passing through Q, then the tangent vector $\dot{Q}(t)$ of $Q(t)$ at $Q(0) = Q$ satisfies 1) $\dot{Q}(t) \in \Delta$, and 2) $\dot{Q}(t)Q(t)^H + Q(t)\dot{Q}(t)^H = 0$. The first condition ensures that $Q(t)$ preserves the structure, the second condition corresponds with the fact that $\frac{d}{dt}(Q(t)Q(t)^H) = \frac{d}{dt}(I) = 0$, which ensures that $Q(t)$ is unitary. Thus, we have:

$$T_Q\mathbf{Q} = \{\dot{Q}(t) \mid \dot{Q}(t) \in \Delta,\ \dot{Q}(t)Q(t)^H + Q(t)\dot{Q}(t)^H = 0_n.\}$$

Let $S = \dot{Q}Q^H$, then

$$T_Q\mathbf{Q} = \{SQ \mid S \in \mathcal{L}\},\ \mathcal{L} \triangleq \{S \mid S \in \Delta, S^H + S = 0_n\}. \tag{2.23}$$

Now define an inner product in the tangent space $T_Q\mathbf{Q}$: Let $A = RQ \in T_Q\mathbf{Q}$ and $B = SQ \in T_Q\mathbf{Q}$ with $S, R \in \mathcal{L}$, then

$$\langle A, B\rangle \triangleq \mathrm{Tr}(AB^H). \tag{2.24}$$

This inner product is induced by the standard inner product in the space $\mathbb{C}^{n\times n}$. It is easy to verify the following equations:

$$\begin{aligned}\langle A, B\rangle &= \langle R, S\rangle = \mathrm{Tr}(R^H S) = \mathrm{Tr}(A^H B)\\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} \mathrm{Re}\,(R_{i,j})\mathrm{Re}\,(S_{i,j}) + \mathrm{Im}\,(R_{i,j})\mathrm{Im}\,(S_{i,j})\\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} \mathrm{Re}\,(A_{i,j})\mathrm{Re}\,(B_{i,j}) + \mathrm{Im}\,(A_{i,j})\mathrm{Im}\,(B_{i,j}),\end{aligned}$$

which are based on the fact that $\mathrm{Tr}(R^H S)$ is real because R and S are skew Hermitian.

The inner product of (2.24) defines a metric on $\mathbf{Q}$. Now with this metric, we can calculate geodesics in $\mathbf{Q}$ and parallel translations of tangent

vectors along geodesics. As already mentioned, in general we have to solve 2n second order nonlinear differential equations for a geodesic calculation and n first order linear differential equations for a parallel translation of a tangent vector (n here is the dimension of the manifold). However in this special manifold we can calculate the geodesics and parallel translations quite easily. It is due to that fact that $\mathbf{Q}$ has a nice structure: 1) $\mathbf{Q}$ is a Lie group[2] and 2) the metric defined by the inner product of (2.24) is a bi-invariant Riemannian metric on $\mathbf{Q}$[3](see our paper [96] for the proofs). Now from Smith[79, p.19], the geodesic, parametrized by the real parameter t, emanating from from $Q = Q(0)$ in a direction SQ is given by

$$Q(t) = \exp(St)Q. \tag{2.25}$$

And the parallel translation of a tangent vector $RQ = R(0)Q(0) \in T_Q\mathbf{Q}$ along the geodesic (2.25) is given by

$$R(t)Q(t) = \exp(\frac{1}{2}St)R(0)\exp(-\frac{1}{2}St)Q(t). \tag{2.26}$$

The reason to choose the general inner product as (2.24) is to find the gradient of the objective function $\phi(Q) = \rho^2(QM)$ on $\mathbf{Q}$, which will be clear in the next section.

2.3.2 Gradient of spectral radius

To implement gradient based optimization algorithms, like the steepest ascent algorithm or the conjugate gradient algorithm, one should first calculate the gradient of the objective function which is going to be maximized. To simplify some of the calculations, $\Phi(Q) = (\rho(QM))^2$ will be used as an objective function, instead of $\rho(QM)$. However, $\Phi(Q)$ is not differentiable at all points $Q \in \mathbf{Q}$. Therefore, theoretically, three cases are to be distinguished.

case a: the spectral radius $\rho(QM)$ corresponds to one eigenvalue of QM with algebraic multiplicity 1.

[2] A Lie group is a group, which is also an analytic manifold, on which the group operations are analytic. We refer to [76, 42] for more about Lie groups.

[3] By the bi-invariance, we mean: $\langle UA, UB\rangle = \langle A, B\rangle = \langle AU, BU\rangle$ for any $U \in \mathbf{Q}$.

case b: the spectral radius $\rho(QM)$ corresponds to $k > 1$ eigenvalues of QM with algebraic multiplicity 1.

case c: the spectral radius $\rho(QM)$ corresponds to an eigenvalue with algebraic multiplicity greater than 1.

Case a is generic. Case b is likely to occur at isolated points of some geodesics (when the absolute value of one eigenvalue crosses the absolute value of another eigenvalue). However, since $\rho(QM)$ is defined as $\max_i |\lambda_i|$ where λ_i are the eigenvalues of QM, and since these absolute values are smooth functions of Q in a neighborhood of points of case b, the maximum of $\rho(QM)$ cannot occur at a point of case b unless $\rho(QM)$ and $\Phi(QM)$ are differentiable at that point. Case c can be considered pathological and can in general be avoided by some heuristics (remark 2). The following theorem gives a formula for the gradient in differentiable points, and shows how the same formula gives an ascent direction (or 0 in a local maximum) for points of case b. The use of non steepest ascent directions, could theoretically prevent the ascent algorithm from converging to a local maximum, but only if the ascent direction would converge to a direction of no change. However, this scenario is excluded by the exceptional nature of both cases b and c.

Theorem 8 *i) Let the spectral radius of $X = QM$ correspond to one eigenvalue λ of X with algebraic multiplicity 1 (case a), and let p be a corresponding right eigenvector and v a corresponding left eigenvector v, satisfying $v^H p = 1$. The gradient of $\Phi(Q) = \rho^2(QM)$, at the point $Q \in \mathbf{Q}$, with respect to the inner product (2.24) is given by*

$$\nabla_Q \Phi = \Phi(Q)GQ \text{ where } G = \pi(vp^H - pv^H), \tag{2.27}$$

where π denotes the orthogonal projection on Δ and is given by

$$\pi(A) = \mathbf{diag}[\alpha_1 I_{r_1}, \ldots, \alpha_s I_{r_s}, A_{f_1,f_1}, \ldots, A_{f_f,f_f}] \text{ for any } A \in \mathbb{C}^{n\times n},$$

where $\alpha_k = \frac{1}{r_k}\mathrm{Tr}(A_{r_k,r_k}), k = 1,\ldots,s$, and $A_{r_k,r_k}, k = 1,\ldots,s$ and $A_{f_l,f_l}, l = 1,\cdots,f$ are the submatrices of A with the same size and position as $\delta_k I_{r_k}$ and Δ_l of $\Delta \in \mathbf{\Delta}$ respectively. That is, π zeros elements that have to be zero in $\mathbf{\Delta}$, and averages elements that have to be equal in $\mathbf{\Delta}$.

ii) If the spectral radius $\rho(QM)$ corresponds to $m > 1$ eigenvalues of QM with algebraic multiplicity 1, then $\rho(QM)$ is the maximum of m differentiable functions, corresponding to the different eigenvalues. For any choice λ among these eigenvalues, and the corresponding right and left eigenvectors p and v, satisfying $v^H p = 1$, $\Phi(Q)GQ$ is the gradient of one of these m functions, and gives an ascent direction of $\rho(QM)$ or gives 0 in a critical point. One of these choices gives the steepest ascent direction.

Proof: i) Clearly, $\Phi(Q)GQ$ belongs to the tangent space $T_Q\mathbf{Q}$, since $\Phi(Q)\pi(vp^H - pv^H)$ is skew Hermitian and belongs to $\mathbf{\Delta}$.

The only thing left to prove is that the directional derivative of Φ at Q in a direction SQ is given by $\langle \nabla_Q \Phi, SQ\rangle$. Or equivalently, if $Q(t)$ describes some path in $\mathbf{Q}$ through $Q = Q(0)$, with velocity $\dot{Q}(0) = SQ \in T_Q\mathbf{Q}$ (thinking of t as time), and $\Phi(t) = \Phi(Q(t))$ that

$$\dot{\Phi}(0) = \langle \nabla_Q \Phi, SQ\rangle. \tag{2.28}$$

From $X = QM$ it is clear that $\dot{X} = \dot{Q}M = SQM = SX$. And since, by assumption the spectral radius of X corresponds to only one eigenvalue λ, $\lambda(t)$ is differentiable at $t = 0$ and $\dot{\lambda}(0) = v^H \dot{X} p = v^H SXp = \lambda v^H Sp$ [49, 61]. Finally, $\dot{\Phi}(0) = \frac{d}{dt}(\lambda^*\lambda) = \dot{\lambda}^*\lambda + \lambda^*\dot{\lambda} = \lambda^*\lambda v^H Sp + \lambda^*\lambda p^H S^H v = \Phi(Q)(v^H Sp - p^H Sv) = \langle S, \Phi(Q)(vp^H - pv^H)\rangle = \langle \pi(S), \Phi(Q)(vp^H - pv^H)\rangle = \langle S, \Phi(Q)\pi(vp^H - pv^H)\rangle = \langle SQ, \Phi(Q)\pi(vp^H - pv^H)Q\rangle$, which proves (2.28).

ii) In a neighborhood of Q, $\Phi(Q)$ can now be regarded as the maximum of m functions, that are equal at Q. The gradient of each of these functions separately is given by $\Phi(Q)GQ$, with one possible choice for λ, and the corresponding p and v. These gradients give steepest ascent directions (or 0 in critical points) for the individual functions, and therefore certainly give ascent directions for the maximum $\Phi(Q)$. One of the gradients corresponds to the steepest ascent direction. □

2.3.3 The steepest ascent algorithm

In this section we derive the steepest ascent algorithm (SAA) on the manifold $\mathbf{Q}$, solving the optimization problem as defined by (2.9). The steepest ascent algorithm (SAA) is an iterative algorithm, calculating

in each step $k = 1, 2, \ldots$ a new point Q_k on the manifold $\mathbf{Q}$, given a point Q_{k-1}. To this end, a (one dimensional) optimization problem, often referred to as *line search*, is solved, maximizing $\rho(QM)$ along a geodesic on $\mathbf{Q}$, emanating from Q_{k-1} in the direction of the gradient given in Theorem 8, which now yields the following algorithm.

Algorithm 2 (Steepest Ascent)

1. *initialize* Q_0
2. $k = 0$
3. **repeat**
4. *compute* $\rho = \rho(Q_k M)$ *and corresponding right and left eigenvectors* p *and* v, *such that* $v^H p = 1$
5. $G = \pi(vp^H - pv^H)$
6. $t = \arg\max_{t>0} \rho(\exp(Gt) Q_k M)$
7. $Q_{k+1} = \exp(Gt) Q_k$
8. $k = k + 1$
9. **until** *convergence*

Remark 1 Note that G in the algorithm and G in Theorem 8 differ by a factor $\Phi(Q)$. Since only the direction of G is important for the algorithm this makes no difference. The exponentiation of $\exp(Gt) = \exp(\Phi(Q)\pi(vp^H - pv^H)t)$ can be calculated from the block components separately,

$$\begin{aligned} \exp(Gt) \;=\; \mathbf{diag}\big[&\exp(G_{r_1,r_1}t), \ldots, \exp(G_{r_s,r_s}t), \\ &\exp(G_{f_1,f_1}t), \ldots, \exp(G_{f_f,f_f}t)\big]. \end{aligned} \tag{2.29}$$

For the calculation of $\exp(G_\square t)$ for different t, where the index $\square$ denotes one of the block components in (2.29), the eigenvalue decomposition of $G_\square$ needs to be calculated only once. If the eigenvalue decomposition of $G_\square$ is given by $G_\square = EjDE^H$, where $E^H E = I$ and D is real diagonal, $\exp(G_\square t)$ is found as:

$$\exp(G_\square t) = E \exp(jDt) E^H.$$

Note that for repeated scalar blocks one finds $E = I$. For full blocks, more efficient formulas can be obtained also, by exploiting the fact that the block components of G have rank 2 (if not 0), but the computational

benefit is rather small compared to the cost of the repeated computation of ρ, p and v.

Remark 2 For points of case c, calculation of an ascent direction is much more difficult than for other points. In addition the effort is not worthwhile, as this case does not normally occur and if it occurs, it can generally be avoided. In most cases, when a point of case c is met during the line search, one can simply take another point. In many cases points of case c can be avoided by multiplying Q with $U \in \mathbf{Q}$, for which $Up = p$ and $U^H v = v$. In that case $\Phi(UQ) = \Phi(Q)$ but in general UQ will no longer be of case c. If none of this works the algorithm is stopped (and possibly restarted from another initial value). After all, the point of case c, can be a local optimum in contrived examples.

The computationally most expensive steps of the algorithm are the evaluations of the spectral radius of X and the corresponding right and left eigenvector. These do not only occur in line 4 of the algorithm but mainly in line 6, during the line search. (Actually, except for the first step in the iteration, nothing needs to be calculated in line 4, since ρ, p and v were calculated already during the last execution of line 6). In order to minimize this cost, two things can be done.

1. The number of eigenvalue/eigenvector calculations is kept low, using an efficient line search algorithm by Fletcher [35, pp.33]. This algorithm finds a point, satisfying the (modified) Wolfe-Powell conditions (normally given for minima instead of maxima)

$$\begin{aligned} \Phi(t) &\geq \Phi(0) + t\,\rho_F \tfrac{d\Phi}{dt}(0), \\ \|\tfrac{d\Phi}{dt}(t)\| &\leq \sigma_F \tfrac{d\Phi}{dt}(0), \end{aligned} \tag{2.30}$$

where ρ_F and $\sigma_F \geq \rho_F$ are fixed parameters. Typically $\sigma_F = 0.5$ and $\rho_F = 0.01$, for a moderately accurate line search [35, p.30]. Fletcher's algorithms are designed for smooth objective functions, but can be easily adapted for the present case, because no optima occur at non differentiable points.

2. For the calculation of ρ, p and v, some profit can be taken from the fact that only the principal eigenvalue and the corresponding eigenvectors are needed, and the fact that previous calculations for nearby X provide good initial estimations. Both the well known power algorithm (not to

be confused with the PIA for the structured singular value, called power algorithm for its resemblance with the classical power algorithm) and the inverse iteration algorithm can save a lot of computations. However, the power algorithm is not efficient if the maximal eigenvalue of X is not well separated (in absolute value) from the other eigenvalues, and the inverse iteration algorithm can converge to the wrong eigenvector with a bad initial guess for the eigenvalue. Therefore, the prize of the computational improvement, is an amount of heuristic rules to handle the distinction between different cases, or the loss of theoretically guaranteed convergence to a local optimum. However, our experience shows that problems are rare and efficiency can be increased, when the line search is made more robust against inaccurate gradient information. In most numerical experiments however we have used the MATLAB function "eig" [54], to calculate a full eigenvalue decomposition. Also in that case the efficiency of the algorithm is fully acceptable.

2.3.4 The conjugate gradient algorithm

In this section we derive the conjugate gradient algorithm (SAA) on the manifold $\mathbf{Q}$, solving the optimization problem as defined by (2.9). For the conjugate gradient algorithm (CGA) the search direction at a point Q_k is determined in most steps by the gradient at Q_k and the previous search direction at Q_{k-1}. In most versions, the gradient direction itself is taken every n-th step (where n is the problem dimension). Smith proposes the following formula for the search direction H_kQ, at a point Q_k using the gradients G_kQ and $G_{k-1}Q$ and the previous search direction $H_{k-1}Q$. It is very similar to the classical formula for the CGA in Euclidean spaces, but tangent vectors at the point Q_{k-1} from the previous iteration step, are adapted to Q_k by parallel translation along the geodesic through Q_{k-1} and Q_k.

$$H_k = G_k + \gamma_k \tau H_{k-1},$$

where

$$\gamma_k = \frac{\langle G_k - \tau G_{k-1}, G_k \rangle}{\langle G_{k-1}, H_{k-1} \rangle},$$

where τ denotes parallel translation from Q_{k-1} to Q_k, applied to matrices $S \in \mathcal{L}$ representing a tangent vector SQ at Q. It follows from theorem 1

that $\tau H_{k-1} = H_{k-1}$. To calculate τG_{k-1} formula (2.26) is needed. This leads to the following algorithm.

Algorithm 3 (Conjugate Gradient)

1. *initialize* Q_0
2. $k = 0$
3. **repeat**
4. *compute* $\rho = \rho(Q_kM)$ *and corresponding right and left eigenvectors* p *and* v, *such that* $v^Hp = 1$
5. $G_k = \Phi(Q)\pi(vp^H - pv^H)$
6. *if* $(k \bmod n) = 0$ *then* $H_k = G_k$
7. *else* $\tau G_{k-1} = \exp(\frac{1}{2}H_{k-1}t)G_{k-1}\exp(-\frac{1}{2}H_{k-1}t)$
8. $\gamma = \frac{\langle G_k - \tau G_{k-1}, G_k \rangle}{\langle G_{k-1}, H_{k-1} \rangle}$
9. $H_k = G_k + \gamma H_{k-1}$
10. $t = \arg\max_{t>0} \rho(\exp(H_kt)Q_kM)$
11. $Q_{k+1} = \exp(H_kt)Q_k$
12. $k = k + 1$
13. **until** *convergence*

Remark 3 For the calculation of $\exp(Ht)$ for different values of t, the same remark holds as for the SAA (remark 1).

Remark 4 For the line search of line 10, the same strategy as for the SAA is used. However, since the performance of the CGA is more sensitive to the quality of this line search, the conditions (2.30) are made stronger by taking $\sigma_F = 0.1$ ($\rho_F = 0.01$) [35, p.85].

The CGA is known to have better convergence properties than the SAA. The additional cost of computing the conjugate gradient search direction is small compared to the cost of the function evaluations and gradient calculations, especially when the structured matrices consist of small blocks, since these computations can be performed on the block components separately. However, due to the stronger requirements for the line search (remark 4), individual iterations may require more computations. In section 2.3.5 we show that for the present problem, both algorithms are more or less equivalent. CGA can be preferred if accuracy is important.

2.3.5 Numerical experiments

In this section we support our theoretical results with numerical experiments, and give an idea of the efficiency of our algorithms and compare them with the PIA as implemented in the MATLAB μ toolbox [4].

The derivation of the power iteration algorithm [61] starts from conditions for a locally optimal value of μ, to obtain a computationally cheap iteration scheme, which, if it converges, finds a local optimum. Convergence however is not guaranteed. Our algorithms have been designed in the first place as convergent algorithms, but require more computations. We propose to use both algorithms together, for example first try the power iteration algorithm, and if it doesn't converge within a preset time limit, use the SAA or CGA. In this way, we obtain a hybrid algorithm which seems to combine the best of both worlds.

In experiments below we have always used the same convergence test. The algorithm stops if the norm of the gradient drops below 1E-3, or if the relative increase of Φ drops below 1E-3 for 2 consecutive steps.

One example of a simple problem for which the PIA does not converge has already been shown in section 2.2.3. The SAA and the CGA, with MATLAB complete eigenvalue/eigenvector calculation or with power iterations or inverse iterations (see section 2.3.3), all give answers in the range [1.3840 1.3846] and, as expected, there is no limit cycling.

Fig. 2.2 compares the computational cost of SAA, CGA and PIA, for complex matrices with different sizes and a similar block structure, consisting of two repeated scalar blocks of size z and two full blocks of size z, where $z = 1, \ldots, 10$. The matrices were generated at random and for each size the computational cost was averaged over 10 matrices. Experiments with different block structures give no significant difference.

The SAA requires less computations than the CGA, but also gives slightly less accurate results. Closer examination reveals that in the majority of the cases this is not due to convergence to another local optimum, but because the convergence criterion (based on the relative increase per step) tends to stop at lower values in the case of the SAA. If one is interested in accurate results, this also means that Fig. 2.2 presents a flattered picture for the SAA, since obtaining the same accuracy as the CGA would require more computations. Fig. 2.3 shows the evolution of $\rho(QM)$ as a function of the number of iterations used already during iterations for the

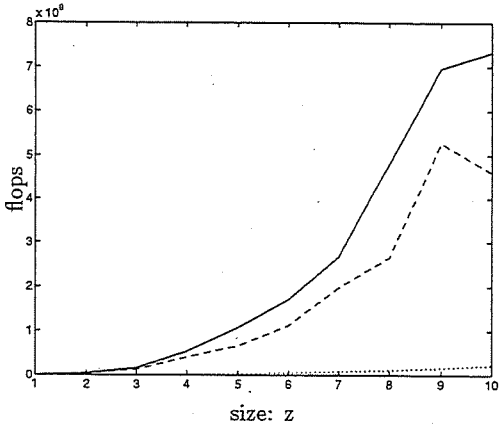


Figure 2.2 Number of floating point operations for CGA (full line —), SAA (dashed line – – –) and PIA (dotted line · · ·), for matrices with different sizes and a block structure, consisting of two repeated scalar blocks of size z and two full blocks of size z, where $z = 1, \ldots, 10$.

SAA and the CGA, for a randomly generated matrix $M \in \mathbb{C}^{10\times 10}$, with a block structure consisting of a full block of size 4 and two repeated scalar blocks of size 3. One clearly sees the better convergence properties of the CGA.

Both the SAA and the CGA are much less efficient than the PIA. However we are comparing an experimental version of SAA and CGA with a fine tuned commercial version of PIA. A considerable improvement is already realized by calculating the eigenvectors and eigenvalues with power iterations or inverse iterations (see section 2.3.3). For both power iterations and inverse iterations the same convergence test was used, stopping the iterations if the change in one iteration of the eigenvector being calculated drops below 1E-3. For power iterations a (seldomly reached) upper limit of 1000 iterations was set. For inverse iterations this limit was set to 100 iterations. If the limit is reached the eigenvectors and eigenvalues are calculated with MATLAB function "eig". Fig. 2.4 shows

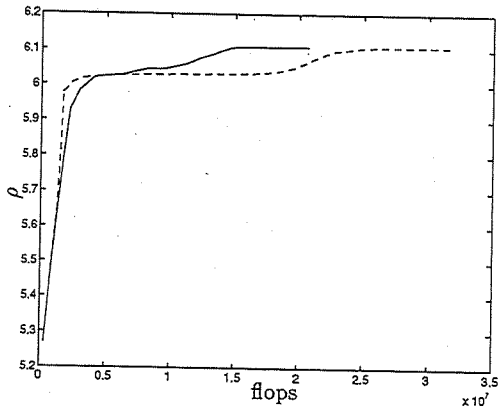


Figure 2.3 Evolution of $\rho(QM)$ as a function of the number of floating point operations used already during iterations for the CGA (full line —) and the SAA (dashed line – – –), for a randomly generated matrix M, with a block structure consisting of a full block of size 4 and two repeated scalar blocks of size 3.

the result for matrices of different sizes, generated in the same way as for the comparison of the SAA and the CGA (figure 2.2). Note that power iterations and inverse iterations require approximately the same number of operations.

2.4 Conclusions

In this chapter, the computation of the complex structured singular value (μ) is studied. The μ problem, which is cast as an optimization problem, can be tackled either by solving the related *optimality conditions* in an iterative way or by applying *general optimization algorithms* using function evaluation and gradient information. The existing iterative scheme PIA (power iteration algorithm), which tries to solve the related *optimality conditions*, sometimes suffers the problem of limit cycling. The

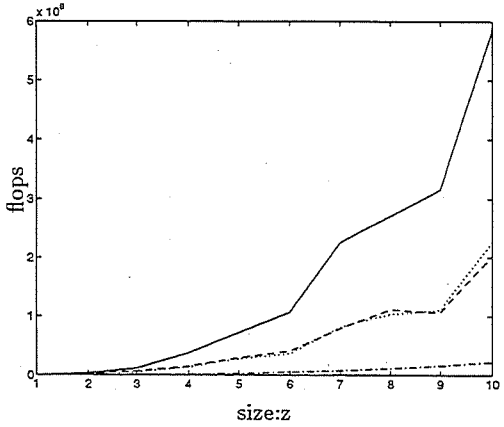


Figure 2.4 Number of floating point operations for SAA, with full eigenvector/eigenvalue computation with the MATLAB function "eig" (full line —), with power iterations (dashed line – – –), with inverse power iterations (dotted line · · ·) and for PIA (dash-dot line · – ·–).

continuous time counter part, CPIA (continuous time power method), of PIA is proposed for getting inside analysis of PIA. It is shown that CPIA has better convergence properties than PIA. However, even CPIA has no proof for guaranteed convergence. The second main result of this chapter is that using the theory of gradient based optimization on Riemannian manifolds, we have derived steepest ascent and conjugate gradient algorithms for the structured singular value problem, with acceptable efficiency and with guaranteed convergence to a locally optimal lower bound for the structured singular value.

Chapter 3

Real parametric μ calculation

Robustness analysis for control systems with real parametric uncertainties is of particular interests because of its practical relevance. There is much literature which tackled this problem. Basically they can be cataloged into two strings: one is based on Kharitonov's type analysis; the other is based on the μ analysis. In Kharitonov's type analysis, the celebrated Kharitonov Theorem [48], introduced to control community by Barmish [7], solves the stability problem for a system when each uncertainty parameter perturbs one coefficient of the characteristic polynomial of the system by checking the stability of merely 4 the so-called Kharitonov's polynomials. The results are generalized to the cases where the coefficients are perturbed by affine combinations of the uncertainty parameters [64]. A comprehensive survey on this method was given by Barmish and Kang [8]. However this Kharitonov's type analysis can hardly find practical applications due to the strict limitation on systems (the parametric uncertainties enter the coefficients of the characteristic polynomial in an affine way). In the μ analysis, the complex μ, as discussed in Chapter 2, is generalized to the mixed μ, where in addition to repeated scalar blocks and full complex blocks, repeated real scalar blocks are included in block structure as well [34, 99], which is used to analyze the control systems with dynamic (repeated scalar, full block) uncertainties and real parametric uncertainties. The real parametric uncertainty problem is a purely real μ problem, which is a special case of the general

mixed μ problems with no complex uncertainty blocks. The complexity analysis has already proved that the calculation of real parametric μ is NP complete [20, 14] in general. The algorithms thus were proposed for calculation of lower bound [99] and upper bound [34]. Both lower and upper bounds are generalized from those for the complex μ. The upper bound problem can be cast as a convex optimization problem, thus can be solved effectively, while the lower bound algorithm generalized from the power iteration for the complex μ has in general poor convergence behavior for calculating the purely real parametric μ. In [30], El Ghaoui defined the l_2 and l_∞ parameter margins. It is the largest 2 or ∞ norm of the parameter vector that ensures stability or H_2 performance. Parameter stability margin problems are equivalent to real parametric μ problems when the l_∞ norm is used. In [31], a quadratically convergent algorithm was proposed to compute the l_2 norm when there are only two uncertain parameters. However, the algorithm does not converge globally. The computation of the l_2 parameter margin can also be formulated in the framework of structured total least squares (STLS) [21]. Though, global convergence is not ensured either.

In this chapter, an algorithm to calculate a lower bound on the real parametric μ, in both l_2 and l_∞ sense, will be described based on the STLS framework. In section 3.1, definitions and formulations for real parametric μ problems will be recapitulated. In section 3.2, the lower and upper bounds and the relations between the bounds for different norms will be discussed. In section 3.3, the main algorithm will be described based on the STLS framework and a variant of the power iteration. Section 3.4 gives some numerical examples. The concluding remarks for this chapter are given in section 3.5.

3.1 Preliminaries

In this section, the definitions of real parametric μ are recapitulated. It is also shown how the real parametric μ is used as a robust stability measure to real parametric uncertainty systems.

3.1.1 Definitions

Real parametric μ

Like the complex μ defined in chapter 2, there is an underlying structure for real parametric μ. Given a real parameter vector $p \in \mathbb{R}^q$ and $p = [p_1\ p_2\ , \cdots,\ p_q]$. The underlying block structure Δ is given as:

$$\Delta(p) \triangleq \left\{ \begin{bmatrix} p_1 I_{r_1} & 0 & 0 & 0 \\ 0 & p_2 I_{r_2} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & p_q I_{r_q} \end{bmatrix}, p \in \mathbb{R}^q \right\}. \tag{3.1}$$

Note that this block structure can also be represented by a block structure vector $bs = [r_1\ r_2\ , \cdots,\ r_q]$. Let $n = \sum_{i=1}^{q} r_i$. Then the real parametric μ is defined as follows.

Definition 10 *Given a matrix $M \in \mathbb{R}^{n\times n}$ and a block structure $\Delta(p)$ defined in (3.1), the real parametric μ of the matrix M with l_k norm is defined as:*

$$\mu_k(M) \triangleq \frac{1}{\min\{\|p\|_k :\ \Delta(p) \in \Delta,\ \det(I + M\Delta(p)) = 0\}}, \tag{3.2}$$

if there is no p such that $\det(I + M\Delta(p)) = 0$, then $\mu_k(M) = 0$.

We only consider $k = 2$ or $k = \infty$. Note that this definition is equivalent to:

$$\mu_k(M) \triangleq \frac{1}{\min\{\|W_k\Delta(p)\|_k :\ \Delta(p) \in \Delta,\ \det(I + M\Delta(p)) = 0\}}, \tag{3.3}$$

where $W_2 = \mathbf{diag}(r_i^{-1/2} I_{r_i})$ for $k = 2$ and $W_\infty = I_n$ for $k = \infty$. Definition (3.3) has the same format as the standard μ definition (2.2) for the case of $k = \infty$.

Parameter margin

Consider a linear, time-invariant closed-loop system of order n:

$$\dot{x} = A(p)x.$$

where x is the state and $A(p)$ is the $n \times n$ dynamic matrix of a plant. The entries of $A(p)$ are rationally dependent on the parameter vector p.

Definition 11 *The parameter stability margin of $A(p)$ is defined as:*

$$pm_k(A(p)) \triangleq \min_{p\in\mathbb{R}^q}\{\|p\|_k \mid A(p) \text{ is unstable}\}. \tag{3.4}$$

Only the 2-norm and ∞-norm will be considered.

3.1.2 Reformulation of the parameter margin problem

The parameter margin problem of (3.4) can be reformulated as a real parametric μ problem.

Without loosing generality, assume:

$$A(p) = A_0 + \sum_{i=1}^{l} f_i(p) A_i,$$

where $A_0 = A(0)$ is a nominal matrix. $f_i(p)$ are scalar rational functions such that $f_i(0) = 0$. Define a matrix function

$$L : \mathbb{R}^{n\times n} \to \mathbb{R}^{n(n+1)/2\times n(n+1)/2} : L(M) \triangleq E(M \otimes M)SE^T,$$

where $\otimes$ is the Kronecker product. The matrix $E \in \mathbb{R}^{(n+1)n/2\times n^2}$ has the partitioned form:

$$E = \begin{bmatrix} E_1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & E_n \end{bmatrix},$$

where $E_i \in \mathbb{R}^{i\times n}$ is given by

$$E_i = [e_1,\ \cdots,\ e_i]^T,$$

the matrix $S \in \mathbb{R}^{n^2\times n^2}$, with

$$s_{n(i-1)+j} = l_{n(i-1)+j} + (1 - \delta_{ij}) l_{n(j-1)+i}, i, j = 1, \cdots, n,$$

where e_i and l_i are the ith basis vectors of the natural bases for $\mathbb{R}^n$ and $\mathbb{R}^{n^2}$ respectively, δ_{ij} is the Kronecker delta. Then we have the following theorem [81]:

Theorem 9 *For any $p \in \mathbb{R}^q$, $A(p)$ is a stable matrix if and only if*

$$\det\left[I + \sum_{i=1}^{l} f_i(p) L(A_i) L(A_0)^{-1}\right] > 0,$$

and the maximal connected subset of the domain of stability containing $p = 0$ is bounded by the set of $p \in \mathbb{R}^q$ such that

$$\det\left[I + \sum_{i=1}^{l} f_i(p) L(A_i) L(A_0)^{-1}\right] = 0. \tag{3.5}$$

The proof of the theorem can be found in [81] based on the construction of a Lyapunov matrix. Thus the parameter margin of $A(p)$ is:

$$pm_k(A(p)) = \min_{p \in \mathbb{R}^q} \{\|p\|_k \mid \det[I + \sum_{i=1}^{l} f_i(p) L(A_i) L(A_0)^{-1}] = 0\}. \tag{3.6}$$

With the multidimensional realization method presented in Chapter 4, we can always find a matrix M and block structure $\Delta(p)$ or bs such that

$$\det\left[I + \sum_{i=1}^{l} f_i(p) L(A_i) L(A_0)^{-1}\right] = 0 \Leftrightarrow \det\left(I + M\Delta(p)\right) = 0,$$

and thus

$$pm_k(A(p)) = \min_{p \in \mathbb{R}^q} \{\|p\|_k \mid \det(I + M\Delta(p)) = 0\}. \tag{3.7}$$

The right hand side is obviously equal to the inverse of $\mu_k(M)$, that is:

$$pm_k(A(p)) = \mu_k(M)^{-1}.$$

Note the parameter margin pm defined above is for system matrix $A(p)$, while μ is defined on a constant matrix M.

3.2 Lower and upper bounds

Calculation of the real parametric μ is not a convex problem. For $k = \infty$, it is already proved to be NP complete [14]. Thus lower and upper bounds are used. In this section, we will focus on an upper bound of μ_∞ denoted by $\bar{\mu}_\infty$, which can be cast as an LMI optimization problem and a lower bound of μ_2 denoted $\underline{\mu}_2$, which can can be cast as a Structured Total Least Squares problem (STLS). But first we discuss some relations between μ_∞ and μ_2 bounds.

3.2.1 Relations of μ_∞ and μ_2 bounds

Given a matrix $M \in \mathbb{R}^{n \times n}$ and a block structure bs. Suppose that $\bar{\mu}_\infty$ and $\underline{\mu}_\infty$ are upper and lower bound of μ_∞, $\bar{\mu}_2$ and $\underline{\mu}_2$ are upper and lower bound of μ_2. That is:

$$\begin{aligned} \underline{\mu}_\infty \le \mu_\infty(M) \le \bar{\mu}_\infty, \\ \underline{\mu}_2 \le \mu_2(M) \le \bar{\mu}_2. \end{aligned} \tag{3.8}$$

Then it can be easily proved that

$$\underline{\mu}_2 \le \mu_\infty \le \bar{\mu}_\infty, \ \underline{\mu}_2 \le \mu_2 \le \bar{\mu}_\infty. \tag{3.9}$$

Actually, suppose $P_e = \arg\min \|p\|_2$ such that $\det(I + M\Delta(P_e)) = 0$ then from the definitions of μ_∞ and μ_2 in (3.2):

$$\min \|p\|_\infty \le \max_i(\|P_{ei}\|_\infty) \le \|P_e\|_2 = \min \|p\|_2,$$

thus

$$\mu_2(M) \le \mu_\infty(M).$$

From (3.8), (3.9) follows. Note that when the point p, which reaches the lower bound $\underline{\mu}_2$, has been found, we can use $1/\|p\|_\infty$ as a lower bound for μ_∞. The significance of (3.9) to use the μ_∞ upper bound and μ_2 lower bound for both μ_∞ and μ_2 is that the μ_∞ upper bound is an LMI optimization problem and the μ_2 lower bound is an STLS problem. Algorithms to solve this problem are available which will be discussed in the next two sections.

3.2.2 Upper bound: LMI optimization

The upper bound of (2.7) for complex μ can certainly be used as an upper bound of the real parametric μ with real repeated scalar blocks replaced by complex repeated scalar blocks. However by using this bound we are not exploring the phase information that is present in real perturbations. The upper bound presented in [34] does exploit this phase information and gives a bound which is never worse than the standard upper bound from complex μ theory, and is frequently much better. The bound in [34] is given for mixed μ. For the real parametric μ_∞, the upper bound can be formulated as a special case of the mixed μ, given as follows:

$$\mu_\infty \leq \sqrt{\max(0, \alpha_*)},$$

where

$$\alpha_* = \inf_{D\in\mathbf{D},G\in\mathbf{G}} \min\{\alpha : (M^H DM + j(GM - M^H G) - \alpha D) \leq 0\}, \quad (3.10)$$

$$\mathbf{D} \triangleq \{\mathbf{diag}(D_1 I_1, \cdots, D_q I_q), D_i \in \mathbb{C}^{r_i \times r_i},\ D_i = D_i^H, D_i > 0\},$$

$$\mathbf{G} \triangleq \{\mathbf{diag}(G_1 I_1, \cdots, G_q I_q), G_i \in \mathbb{C}^{r_i \times r_i},\ G_i = G_i^H\}.$$

The nice observation here is that the minimization of (3.10) involves an LMI (Linear Matrix Inequality) optimization, it is (quasi) convex and hence this bound is computationally attractive.

3.2.3 Lower bound: STLS

Here we give a lower bound for μ_2, which turns out to be an STLS problem.

Given a matrix $M \in \mathbb{R}^{n\times n}$ and a block structure $\mathbf{\Delta}(p)$ with $bs = [r_1, \cdots, r_q]$, the μ_2 problem

$$\mu_2(M)^{-1} = \min_{p\in\mathbb{R}^q}\{\|p\|_2 | \det(I + M\Delta(p) = 0\}$$

can be reformulated as

$$\mu_2(M)^{-2} = \min \|p\|_2^2 \text{ subject to } \begin{matrix} B(p)y = 0 \\ y^T y = 1 \end{matrix}, \quad (3.11)$$

where $B(a) = I + M\Delta(p) = I + p_1 B_1 + p_2 B_2 + \cdots + p_q B_q$ and

$$B_i = MP_i,\ P_i = \text{block } \mathbf{diag}(0_{\sum_{k=1}^{i-1} r_k}, I_{r_i}, 0_{\sum_{k=i+1}^{q} r_k}),$$

where "0_n" denotes a zero matrix with dimension $n \times n$. This problem is the special case of the Structured Total Least Squares Problem [21] with $B_0 = I$. Now we use a Lagrange multiplier method to find necessary optimality conditions for the minimization problem in the right hand side of (3.11). The Lagrange is

$$\mathcal{L} = \|p\|_2^2 + 2l^T B(p)y + \lambda(y^T y - 1),$$

where l is a vector of Lagrange multipliers and λ is a scalar Lagrange multiplier. Differentiation with respect to the several unknowns yields necessary conditions for the solution:

$$\frac{\partial \mathcal{L}}{\partial p_k} = 0 \ \Rightarrow\ p_k + l^T B_k y = 0, \quad (3.12)$$

$$\frac{\partial \mathcal{L}}{\partial y} = 0 \ \Rightarrow\ B(p)^T l + y\lambda = 0, \quad (3.13)$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = 0 \ \Rightarrow\ y^T y = 1, \quad (3.14)$$

$$\frac{\partial \mathcal{L}}{\partial l} = 0 \ \Rightarrow\ B(p)y = 0. \quad (3.15)$$

Now, we can eliminate the parameter vector p as follows. Premultiplying (3.13) with y^T and using (3.14) and (3.15) yields:

$$y^T B(p)^T l + y^T y\lambda = 0 \ \Rightarrow\ \lambda = 0.$$

With $\lambda = 0$, (3.13) becomes:

$$B(p)^T l = 0. \quad (3.16)$$

Substituting (3.12) in (3.16), we have

$$\left[I - \sum_{k=1}^{p} B_k^T (l^T B_k y)\right] l = 0.$$

This can be rewritten as

$$\begin{aligned} l &= \sum_{k=1}^{q} B_k^T l (l^T B_k y) = \sum_{k=1}^{q} (B_k^T l)(B_k^T l)^T y \\ &= \left[\sum_{k=1}^{q} (B_k^T l)(B_k^T l)^T \right] y. \end{aligned} \tag{3.17}$$

Similarly, when (3.12) is substituted in (3.15), we have

$$\left[I - \sum_{k=1}^{q} B_k (y^T B_k^T l) \right] y = 0,$$

which can be rewritten as

$$\begin{aligned} y &= \sum_{k=1}^{q} B_k y (y^T B_k^T l) = \sum_{k=1}^{q} (B_k y)(B_k y)^T l \\ &= \left[\sum_{k=1}^{q} (B_k y)(B_k y)^T \right] l. \end{aligned} \tag{3.18}$$

The last step is to normalize the length of vector l. Let $x = \frac{l}{\|l\|_2}$ and $\sigma = \|l\|_2$. Substituting x and σ in (3.17) and (3.18), we finally have

$$\begin{aligned} &x = D_x y \sigma, \; x^T x = 1, \\ &y = D_y x \sigma, \; y^T y = 1. \end{aligned} \tag{3.19}$$

with

$$D_x = \sum_{k=1}^{q} (B_k^T x)(B_k^T x)^T,$$

$$D_y = \sum_{k=1}^{q} (B_k y)(B_k y)^T.$$

Suppose a triplet (x, y, σ) solves (3.19), then parameter vector p which satisfies $B(p) = 0$ can be obtained from (3.12)

$$p_k = -\sigma x^T B_k y$$

and thus

$$\|p\|_2^2 = \sum_{k=1}^{q} |p_i|^2 = \sigma^2 \sum_{k=1}^{q} (x^T B_k y)(x^T B_k y),$$

which can be further written as

$$\|p\|_2^2 = \sigma^2 x^T D_y x = \sigma^2 y^T D_x y. \tag{3.20}$$

Thus the μ_2 problem now has been reformulated as a "non-linear" SVD problem in (3.19), which is also the so called Riemannian SVD problem [22]. Note that even the problem of (3.19) can be solved, we still can not ensure that a global maximum, pm_2, of $\|p\|_2$ with $\det(I+M\Delta(p)) = 0$ is reached, since we only find a *necessary* optimality condition. Thus in general we only can get an upper bound of pm_2, or a lower bound for μ_2.

3.3 Algorithms for lower bounds

In this section, an algorithm based on the power method for calculating lower bound for μ_2 will be presented. For comparison, the algorithms for lower bounds on μ_∞ will be recapitulated.

3.3.1 Power method for μ_2

Like for the complex μ discussed in the last chapter, the power method is once again used to solve the optimality condition of (3.19). That is we attempt to find solutions to (3.19) in an iterative way. Let the superscript $[k]$ denote the k-th iteration, then the power iteration algorithm is given as follows:

Algorithm 4 (Power iteration for μ_2 (PMU2))

1. *Initialization: chose* $x^{[0]}, y^{[0]}$ *and normalize* $\|x^{[0]}\| = 1, \|y^{[0]}\| = 1$, *k=1*
2. **Repeat**
3. *Compute a new* $x^{[k]}$: $x^{[k]} = D_{x^{[k-1]}} y^{[k-1]}$
4. *Normalize* $x^{[k]}$: $x^{[k]} \leftarrow \frac{x^{[k]}}{\|x^{[k]}\|}$
5. $\sigma_x^{[k]} = \|x^{[k]}\|$
6. *Compute a new* $y^{[k]}$: $y^{[k]} = D_{y^{[k-1]}} x^{[k]}$
7. *Normalize* $y^{[k]}$: $y^{[k]} \leftarrow \frac{y^{[k]}}{\|y^{[k]}\|}$
8. $\sigma_y^{[k]} = \|y^{[k]}\|$
9. *Convergence test with* $x^{[k]}, y^{[k]}, \sigma_x^{[k]}$ *and* $\sigma_y^{[k]}$
10. *k=k+1*

11. **Until** *convergence*

The convergence can be checked by *e.g.* computing the smallest singular value of $B(p)$. When it is small enough, stop the iteration. But first we can always check the difference between $\sigma_x^{[k]}$ and $\sigma_y^{[k]}$, and only when the difference is small enough, then check the smallest singular value of $B(p)$. This may save some computations in the earlier iteration. The convergence properties are good in most of the practical cases. Typically if neither D_x nor D_y is singular, then, the asymptotic convergence rate of this algorithm is linear if there exists a solution at all [21]. However, if D_x or D_y is singular in the kth step, that is the null space of $D_{x^{[k-1]}}$ or $D_{y^{[k-1]}}$ is not empty, and $y^{[k-1]}$ or $x^{[k]}$ is in the null space of $D_{x^{[k-1]}}$ or $D_{y^{[k-1]}}$ respectively, then step 5 or step 7 will not be well defined. The iteration will break down. Though numerically, it does hardly happen that $y^{[k-1]}$ or $x^{[k]}$ is exactly in the null space of $D_{x^{k-1}}$ or $D_{y^{[k-1]}}$, but $\|x^{[k]}\|$ or $\|y^{[k]}\|$ in this case is very small, and the algorithm could eventually run into a limit cycling. The situation of rank deficiency of D_x or D_y happens when the number of parameters in p is small compared to the size of the matrix M, typically when there is only one parameter, then the rank of D_x and D_y is always 1. When the limit cycling happens due to the rank deficiency of D_x or D_y, we can however always restart the iteration with another initial value such that x or y is not in the null space of D_y or D_x before it is convergent. As mentioned already, the convergence can only be locally. If the initial point is close enough to a local optimum it will converge to it. Therefore the algorithm should be invoked several times from different starting points. The resulting parameter vector with the lowest norm is taken to be an upper bound of μ_2^{-1}.

3.3.2 Continuous time algorithm for μ_2

Let $z \in \mathbb{R}^{2n}$ with $z = \begin{bmatrix} x \\ y \end{bmatrix}$, and $x, y \in \mathbb{R}^n$. The optimality condition of (3.19) can be written as:

$$Az = D(z)z\sigma,\ z^Tz = 1,\ x^Tx = y^Ty, \tag{3.21}$$

where

$$A = \begin{bmatrix} & I_n \\ I_n & \end{bmatrix},\ D(z) = \begin{bmatrix} D_x & \\ & D_y \end{bmatrix}.$$

Like for the continuous time power method for μ in section 2.2.2, one possible differential system to solve (3.21) is as follows:

$$\boxed{\dot{z} = -Az + D(z)z\sigma, \sigma = z^TAz/(z^TD(z)z).} \tag{3.22}$$

At an equilibrium point (3.22) becomes equal to (3.21). (3.22) can be proved to be a gradient flow in the case that $D(z)$ is full rank [21] on the manifold formed by two spheres $x^Tx = 1/2$ and $y^Ty = 1/2$, so asymptotic convergence is always guaranteed. This asymptotic convergence property ensures that there is always at least one solution to the STLS problem of (3.11), thus, on the other hand, it proves that: If a STLS problem of (3.11) is not feasible, then there is a pair of x and y such that D_x or D_y are rank deficient and x or y is in the null space of D_y or D_x respectively. However, the opposite results are not true. It happens (actually often) that there is a pair of x and y such that D_x or D_y are rank deficient and x or y is in the null space of D_y or D_x, but the corresponding STLS is feasible. The main disadvantage of this flow is that σ converges to the maximum eigenvalue of $D(z)^{-1}A$ (suppose $D(z)$ is invertible), which is equal to the maximum singular value of D_x^{-1} or D_y^{-1}, instead of the minimum one. One possible improvement is to use the following differential system [22]:

$$\dot{z} = B\left[-Az + D(z)z\sigma\right],\ \sigma = \frac{z^TAz}{z^TD(z)z}, \tag{3.23}$$

where

$$B = \begin{bmatrix} I_n & \\ & -\phi I_n \end{bmatrix},\ \text{with}\ \ 0 < \phi < 1\ \ \text{fixed}.$$

Obviously at an equilibrium point, (3.23) becomes (3.21), and we have a solution. More than that it can be proved [22] that σ at the equilibrium point is always equal to the minimum eigenvalue of the matrix $D(z)^{-1}A$. However, unlike (3.22), the convergence property of (3.23) is currently not yet understood, though numerical experiences show that it is always convergent when $D(z)$ is not rank deficient. We will label (3.23) as CPMU2 (continuous time power method for μ_2).

3.3.3 Algorithms for μ_∞

For comparison, we briefly recapitulate two lower bound algorithms for μ_∞.

Power iteration for μ_∞

The lower bound of complex μ, discussed in Chapter 2, may not be used as a lower bound of real parametric μ_∞ since otherwise that would include perturbations from outside the permissible set $\mathbf{\Delta}(p)$, and so would not yield a valid lower bound. Hence the algorithms discussed in chapter 2 can not be used for a lower bound calculation of μ_∞. Young and Doyle [99] generalized the power iteration method for the mixed μ case. Here we recapitulate the algorithm for the purely real case. The real parametric μ_∞ (purely real μ) is cast as the following maximization problem:

$$\max_{Q(p)\in\mathbf{Q}(p)} \rho_R(MQ(p)), \tag{3.24}$$

where

$$\mathbf{Q}(p) \triangleq \{Q(p) : \|Q(p)\| \le 1, Q(p) \in \mathbf{\Delta}(p)\}.$$

Note the differences between the real and complex μ: first, in the complex case, the objective function is the spectral radius ($\rho(MQ)$), while here is the maximum absolute real eigenvalue ($\rho_R(MQ(p))$); second, in the complex case, $\mathbf{Q}$ is the sphere of a structured unit ball (structured unitary matrix), while in real case it is a unit box in the parameter space. The optimality condition of this maximization problem is as follows. Suppose $Q(p) \in \mathbf{Q}(p)$ achieves a (local) maximum of (3.24), and the eigenvalue, β, is distinct and positive, and the corresponding left and right eigenvectors satisfy the non-degeneracy assumption (see section 2.2.1), then there exists vector a, b, w and z partitioned as:

$$a = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_q \end{bmatrix}, \; b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_q \end{bmatrix}, \; w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_q \end{bmatrix}, \; z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_q \end{bmatrix},$$

with a_i,b_i,w_i and $z_i \in \mathbb{C}^{bs(i)}$, such that:

$$\begin{array}{lll} Mb = \beta a, & z_i = q_i w_i, & i = 1, \cdots, q, \\ M^H z = \beta w, & b_i = q_i a_i, & i = 1, \cdots, q, \end{array} \tag{3.25}$$

for some real scalars $q_i \in [-1\ 1]$ with

$$\mathrm{Re}(a_i^H w_i) \ge 0, \text{ for } q_i = 1,$$

$$\mathrm{Re}(a_i^H w_i) \le 0, \text{ for } q_i = -1,$$

$$\mathrm{Re}(a_i^H w_i) = 0, \text{ for } |q_i| < 1.$$

Now the following power iteration is then used to find a solution for this optimality condition [99]:

Algorithm 5 (Power iteration for μ_∞ (PMU_∞))

1. *Initialization: chose* $b^{[0]}, w^{[0]}$ *and normalize*
   $\|b^{[0]}\| = 1, \|w^{[0]}\| = 1; q_i^{[0]} = 1, i = 1, \cdots, q,$
   $k=0$
2. **Repeat**
3. $\tilde{\beta}^{[k+1]} a^{[k+1]} = M b^{[k]}$, with $\tilde{\beta}^{[k+1]} > 0$ and $|a^{[k+1]}| = 1$
4. $z_i^{[k+1]} = \tilde{q}_i^{[k+1]} w_i^{[k]}, \; i = 1, \cdots, q$
   $\tilde{\alpha}_i^{[k+1]} = \mathrm{sign}(\hat{q}_i^{[k]}) \frac{|b_i^{[k]}|}{|a_i^{[k+1]}|} + \mathrm{Re}((a_i^{[k+1]})^H w_i^{[k]})$
   $\tilde{q}_i^{[k+1]} = \frac{\tilde{\alpha}_i^{[k+1]}}{\max\{1, |\tilde{\alpha}_i^{[k+1]}|\}}$
5. $\hat{\beta}^{[k+1]} w^{[k+1]} = M^H z^{[k+1]}$, with $\hat{\beta}^{[k+1]} > 0$ and $|w^{[k+1]}| = 1$
   $\hat{\alpha}_i^{[k+1]} = \mathrm{sign}(\tilde{q}_i^{[k+1]}) \frac{|b_i^{[k]}|}{|a_i^{[k+1]}|} + \mathrm{Re}((a_i^{[k+1]})^H w_i^{[k+1]})$
   $\hat{q}_i^{[k+1]} = \frac{\hat{\alpha}_i^{[k+1]}}{\max\{1, |\hat{\alpha}_i^{[k+1]}|\}}$
6. $b_i^{[k+1]} = \hat{q}_i^{[k+1]} a_i^{[k+1]}, \; i = 1, \cdots, q$
7. $k = k + 1$
8. **Until** *convergence*

It is easy to verify that if the algorithm converges to some equilibrium then a solution for the optimality condition is found, which can be further used to calculate a Q which achieves a local maximum. However the convergence properties are significantly poorer than in the complex cases as indicated in [100].

Corners checking

One simple method to get a lower bound for μ_∞ is to find a box with one of its corners touching the constraints. El Ghaoui [31] pointed out that, in most practical cases, the largest box (the length of its edges is equal

to $2\mu_\infty^{-1}$) touches the constraints in one of its corners. In this case, μ_∞ can be easily found by solving 2^{q-1} eigenvalue problems.

Suppose we are trying to find μ_∞ along a line in the parameter space with direction, say, p_0: $p = \lambda^{-1}p_0$, $\lambda \in \mathbb{R}$. Then

$$\begin{aligned}\mu_\infty(M)^{-1} &= \min\{\|p\|, |\det(I + M\Delta(p)) = 0\} \\ &= \|p_0\|_\infty \min\{|\lambda^{-1}|, |\det(\lambda + M\Delta(p_0)) = 0\}.\end{aligned}$$

That is

$$\mu_\infty(M) = \max \frac{\rho_R(M\Delta(p_0))}{\|p_0\|_\infty} = \max \rho_R(M\Delta(\frac{p_0}{\|p_0\|_\infty})).$$

Note that this equation is exactly the same as (3.24) with $p = \lambda^{-1}p_0$, since $\Delta(\frac{p_0}{\|p_0\|_\infty}) \in \mathbf{Q}(p)$. To find the largest box, check all the corner directions:

$$\left[\begin{array}{cccc} p_1, & p_2, & \cdots, & p_q \end{array}\right] = \left[\begin{array}{cccc} 1, & \pm 1, & \cdots, & \pm 1 \end{array}\right].$$

Note that we can only find a local maximum, that is, a lower bound of μ_∞ in general. It is also possible that there is no solution found at all, that is, none of the corners of any box touches the constraints, while μ_∞ is not zero.

3.4 Numerical examples

In this section we will give some numerical examples based on the algorithms discussed on this chapter. The upper bound discussed in section 3.2.2 for both μ_∞ and μ_2 will be calculated. The lower bounds are also calculated with different algorithms: the STLS power iteration algorithm (PMU2) and the continuous algorithm (CPMU2) for the lower bounds of μ_2, the μ power iteration algorithm (PMU∞) and the corner checking (CORNER) for μ_∞.

Example 1 *This example is just for numerical purposes, there is no direct physical interpretation. Given matrices*

$$B_1 = \left[\begin{array}{cc} 1.6287 & -0.6498 \\ -0.9805 & 1.3541 \end{array}\right],$$

$$B_2 = \left[\begin{array}{cc} -0.0630 & 2.2362 \\ -1.8017 & -0.3202 \end{array}\right],$$

$$B_3 = \left[\begin{array}{cc} -0.1298 & -0.3380 \\ 0.5481 & 0.3840 \end{array}\right].$$

A matrix M, satisfying

$$I_2 + p_1B_1 + p_2B_2 + p_3B_3 = 0 \Leftrightarrow M\Delta(p) = 0,$$

can be found by the algorithm described in chapter 4 as:

$$M = \left[\begin{array}{cccccc} 2.3130 & -0.1380 & 2.1777 & 0.6631 & -1.1673 & -0.2972 \\ 0.1380 & 0.6698 & -0.4044 & 1.0340 & -0.3210 & 0.3130 \\ -1.1791 & -1.0650 & -0.2154 & -2.0032 & 1.0159 & -0.4023 \\ 1.7463 & -0.5599 & 2.0032 & -0.1677 & -0.7124 & -0.4467 \\ 0.1082 & -0.7030 & 0.6507 & -0.9907 & 0.2036 & -0.3537 \\ -0.6510 & -0.0290 & -0.5594 & -0.2861 & 0.3537 & 0.0506 \end{array}\right].$$

The upper bound calculated based on convex optimization algorithm (use MATLAB function "mu" in μ-toolbox[4]) is

$$\bar{\mu}_\infty = 2.5475.$$

For the lower bounds, we first calculate the lower bound based on algorithm PMU2. Fig. 3.1 shows the evolution of $\|p\|_2$, calculated by (3.20) in each iteration step. It converges rather fast (less than 6 steps). The corresponding lower bound:

$$\text{PMU2}: \quad \underline{\mu_2} = 2.3449,$$

reached at point:

$$p_{PMU2} = \left[\begin{array}{ccc} -0.4237 & 0.0476 & 0.0079 \end{array}\right].$$

The continuous algorithm gives the evolution of $\|p\|_2$ shown as in Fig. 3.2 with $\phi = 0.5$. It converges eventually but not fast to a lower bound:

$$\text{CPMU2}: \quad \underline{\mu_2} = 2.3449,$$

at point

$$p_{CPMU2} = \left[\begin{array}{ccc} -0.4238 & 0.0471 & 0.0079 \end{array}\right].$$

Table 3.1 Eigenvalues of $MQ(p)$ at all corners for Example 1.

corner	(1, −1, −1)	(1, −1, 1)	(1, 1, −1)	(1, 1, 1)
eigen−value	$1.56+0.79j$	$1.81+2.09j$	$1.17+2.48j$	$1.43+1.67j$
	$1.56-0.79j$	$1.81-2.09j$	$1.17-2.48j$	$1.43-1.67j$
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00
	0.00	0.00	0.00	0.00

There is no essential difference between these two points, actually they are the same if using the same stop criteria. Note that now we can give a lower bound for μ_∞ already:

$$\text{CPMU2}: \underline{\mu_\infty} = 1/\|p_{PIMU2}\|_\infty = 1/0.4237 = 2.3602.$$

Fortunately, the power method for a lower bound of μ_∞ also gives a convergent result in this example, Fig. 3.3 shows the evolutions of $\tilde{\beta}$ and $\hat{\beta}$. They converge in about 8 steps. The lower bound is

$$\text{PMU}\infty: \quad \underline{\mu_\infty} = 2.4861,$$

reached at point:

$$p_{PIMU\infty} = \begin{bmatrix} 0.4022 & -0.1582 & -0.4022 \end{bmatrix}.$$

This lower bound is a little bit tighter than that given by PIMU2. Finally we check eigenvalues at all the corners of box $\mathbf{Q}(p)$, and the maximum real eigenvalue in the corners is

$$\text{CORNER}: \underline{\mu_\infty} = 0.$$

This result shows that the corner checking fails. Actually, as shown in Table 3.1, there is no corner at which there is at least one non zero real eigenvalues. We can check all of the corners directly for $B(p) = 0$, the result is of course the same. But in this case, the eigenvalues at the corners are the first two eigenvalues in Table 3.1.

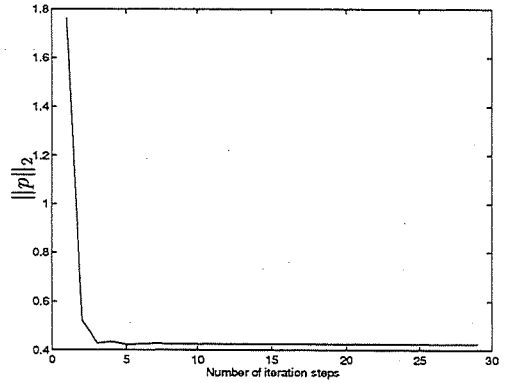


Figure 3.1 $\|p\|_2$ evolution in STLS power iteration.

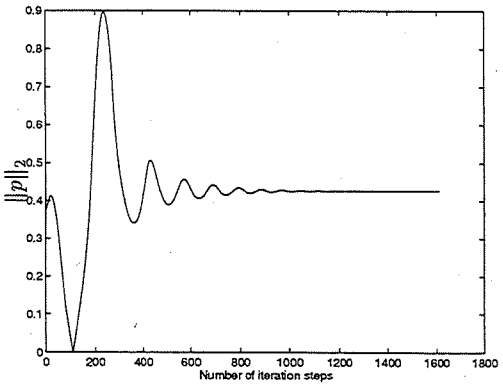


Figure 3.2 $\|p\|_2$ evolution in continuous time STLS power iteration.

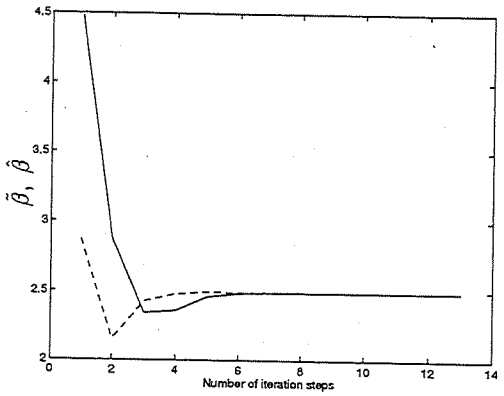


Figure 3.3 $\tilde{\beta}$ and $\hat{\beta}$ evolution in μ power iteration.

Example 2 *This example is from [30] to solve parameter margin problem for a three mass/spring system, which is also a benchmark problem of the 1991 ACC (American Control Conference) [85]. Given the open loop transfer function for a three mass/spring system as:*

$$\frac{(p_1+1)(p_2+1)/6}{s^2[s^4+(\frac{5}{6}p_1+\frac{3}{2}p_2+\frac{7}{3})+(p_1+1)(p_2+1)]}.$$

where p_1 and p_2 are two uncertain parameters and $\|p_1\| < 1$ and $\|p_2\| < 1$. The compensator's transfer function as

$$\frac{0.7466s^3-0.1132s^2+0.5385s+0.0416}{s^3+0.8233s^2+1.2384s+0.3330}.$$

The closed loop system matrix $F(p)$ has the following form:

$$A(p) = A_0 + p_1p_2A_1 + p_1A_2 + p_2A_3,$$

where

$$A_0 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -0.124 & 0 & -1 & 0 & -2.33 & 0 & 0.121 & 0.0643 & 0.0345 \\ 1 & 0 & 0 & 0 & 0 & 0 & -0.823 & -0.124 & -0.333 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix},$$

$$A_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -0.124 & 0 & -1 & 0 & 0 & 0 & 0.121 & 0.0643 & 0.0345 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$A_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -0.124 & 0 & -1 & 0 & -0.833 & 0 & 0.121 & 0.0643 & 0.0345 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$A_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -0.124 & 0 & -1 & 0 & -1.5 & 0 & 0.121 & 0.0643 & 0.0345 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

In order to solve the parameter margin problem of (3.4), we convert (3.4) to (3.5) by the algorithm given by Theorem 9 in Chapter 4, with

$$f_1(p) = p_1p_2,\ f_2(p) = p_1,\ f_3(p) = p_2,$$

and obtain 45×45 *matrices* $L(A_1)L(A_0)^{-1}$, $L(A_2)L(A_0)^{-1}$ *and* $L(A_3)L(A_0)^{-1}$ *(which are obviously too large to write down here). Then using the multidimensional realization algorithm in Chapter 4, we find the matrix* M *and* $\Delta(p)$ *with block bs as follows (see also Example 6):*

$$M_b = \left[\begin{array}{cccccc}
1.9929 & -0.2391 & 0.0747 & 0.1245 & -0.0838 & -0.1007 \\
-9.5041 & 1.1314 & 0.1896 & 0.4209 & 0.1793 & 0.0069 \\
1.7670 & 0.4742 & 0.9628 & -0.1175 & 0.0902 & -0.0457 \\
15.3116 & 1.7784 & -0.3696 & 0.0746 & 0.2095 & -0.1161 \\
10.1655 & -2.6222 & 0.0265 & 0.0984 & 0.1719 & -0.0678 \\
18.9315 & 0.5319 & 0.2194 & 0.0654 & -0.2201 & 0.1460 \\
-0.0671 & -0.0419 & 0.0054 & 0.0259 & -0.0112 & -0.0106 \\
-1.1655 & -0.1205 & -0.2577 & 0.1151 & -0.0387 & 0.0133 \\
0.0472 & -0.1152 & -0.6470 & 0.1433 & -0.0927 & -0.0002 \\
-0.3916 & -0.0731 & -0.2299 & 0.0749 & -0.0355 & 0.0026 \\
1.2426 & -0.0007 & -0.4926 & 0.0432 & -0.0653 & -0.0105 \\
-1.4522 & -0.1311 & -0.7770 & 0.1830 & -0.0823 & 0.0467
\end{array}\right.$$

$$\left.\begin{array}{cccccc}
3.7298 & -1.6699 & -0.8111 & -0.3755 & 0.0991 & 0.5748 \\
-27.5936 & 2.1338 & -1.9431 & -0.4148 & 0.7559 & -0.1398 \\
6.1184 & 1.9612 & 0.5615 & -1.7542 & -0.1803 & -0.0402 \\
48.4273 & 5.5715 & 0.3985 & 0.2696 & 0.6219 & 0.1086 \\
24.2354 & -9.4436 & 0.7867 & -0.1839 & 0.4058 & -0.0602 \\
53.4500 & -0.9555 & -1.8426 & -0.0551 & -0.4325 & -0.1086 \\
1.5411 & -0.0981 & -0.0474 & -0.0314 & 0.0058 & 0.0619 \\
-1.9742 & 1.6895 & 0.1627 & 0.3244 & 0.0620 & -0.2106 \\
1.2129 & -0.3618 & 0.9928 & 0.7967 & -0.1273 & -0.2007 \\
-0.3014 & 0.1342 & 0.0450 & 1.2813 & -0.0066 & -0.1051 \\
3.2295 & -1.0823 & -0.1579 & 0.6067 & 0.7888 & -0.0390 \\
-3.4990 & 1.0193 & 0.3525 & 1.0321 & -0.0049 & 0.4138
\end{array}\right]$$

$bs = [6\ 6]$.

Now the lower and upper bounds of the parameter margins of $pm_\infty(A(p))$ *and* $pm_2(A(p))$ *can be obtained by calculating the upper and lower bounds of* $\mu_\infty(M)$ *and* $\mu_2(M)$. *The lower bound of* $pm_\infty(A(p))$ *is equal to the inverse of the upper bound of* $\mu_\infty(M)$ *and is calculated based on convex optimization algorithm:*

$$\underline{pm}_\infty(A(p)) = \bar{\mu}_\infty^{-1} = 0.42118.$$

The STLS power iteration is again successful and $\|p\|_2$ *converges in about 30 steps, as shown in Fig. 3.4, and*

$$\overline{pm}_2((A(p)) = 0.50171,$$

reached at point:

$$p_{PMU2} = \left[\begin{array}{cc} -0.49953 & -0.046697 \end{array}\right].$$

Now we have an upper bound for $pm_\infty(A(p))$ *already:*

$$\overline{pm}_\infty((A(p)) = 0.49953,$$

which is acceptable compared with the lower bound. Both CPMU2 and PMU∞ *fail in this example. CPMU2 fails due to rank deficiency of* D_x *and* D_y *and* $\|p_2\|$ *eventually goes to infinity. While PMU*∞ *fails due to limit cycling. However corner checking gives a very good upper bound:*

$$\overline{pm}_\infty((A(p)) = 0.42322,$$

reached at corner:

$$p_{CORNER} = \left[\begin{array}{cc} -0.42322 & -0.42322 \end{array}\right].$$

Actually it can be verified, by griding the unit box in the parameter space, then calculating the maximum real eigenvalues at all the points, that p_{PIMU2} *and* p_{CORNER} *reach* $pm_2(A(p))$ *and* $pm_\infty(A(p))$ *respectively.*

3.5 Conclusions

In this chapter, the real parametric μ is studied. The real parametric μ is used as a robust stability measure for parametric uncertainty systems. The parameter stability margin problem has been cast as the real parametric μ problem.

The study is focussed on the calculation of a lower bound of μ_2, which turns out to be an STLS problem. A new algorithm is thus proposed to solve the lower bound problem. The algorithm is an iterative process based on a variant of the power iteration. In most of the numerical examples, the algorithm shows good convergence. However there is no guarantee for convergence, typically, in the case that D_x or D_y is rank deficient which happens when the number of parameters is small compared with the size of the matrices B_i. The continuous time counterpart of the algorithm is also proposed. Unfortunately it seems that the convergence behavior is even worse in the case of D_x or D_y being rank deficient. For

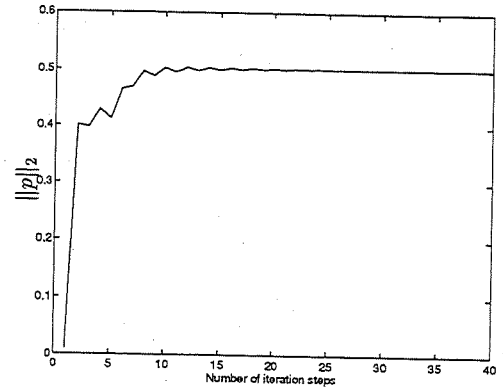


Figure 3.4 $\|p\|_2$ evolution in STLS power iteration.

the first time, we also use an upper bound of μ_∞, which turns out to be an LMI optimization problem and therefore computationally attractive, as an upper bound of μ_2. The tightness of this upper bound is acceptable as shown in two examples. We expect that the upper bound can also be used in other STLS applications for systems and control [21].

Chapter 4

Parametric uncertainty modeling

In this chapter we shall focus on the parametric uncertainty modeling problem, which is related to the multidimensional realization problem, while the full block modeling problem will be discussed in Chapter 7. In section 4.1, we discuss some general concepts on the system modeling and the uncertainty modeling. In section 4.2, we give a definition for the multidimensional realization problem, and relations between this problem and parametric uncertainty modeling and formulation of multi parameter margin problems. The main results will be presented in section 4.3 on the realization algorithm. Section 4.4 discusses two size reduction methods. Numerical examples and concluding remarks will be given in section 4.5 and 4.6 respectively.

4.1 System modeling and uncertainty modeling

For model based control, a physical plant is represented by a mathematical model which is built based on physical laws or experimental data via identification techniques. For mathematical reasons, such models are simple, and in many of cases, they are linear and time invariant (LTI), we refer to [53] for a comprehensive understanding of the methodology of modeling. However a real plant is always complex, and a mathematical model may never be an exact description of the plant. Thus all phenom-

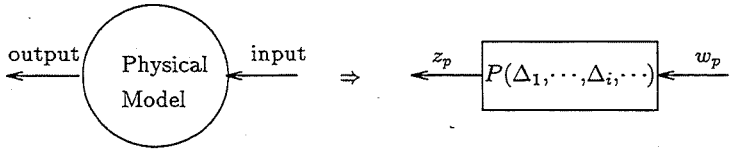


Figure 4.1 System modeling.

ena that can not be explained by the model are represented (or explained) by uncertainties. Errors in design data, variations in environmental data (*e.g.* temperature) and nonlinearities are typical uncertainties. In the step of system modeling, these uncertainties are represented mathematically by some unknown operators, as shown in Fig. 4.1 where P is a linear system but with some Δ_i's which are the uncertain operators. Such kind of mathematical models with uncertain operators are difficult to be obtained by identification techniques based on experimental data, especially when there exists more than one uncertainty. Up to now identification techniques are available for the case where only one uncertainty exists [41]. Modeling of this kind of uncertainty systems is generally based on available physical laws. Bond graph method offers a good way to tackle the modeling problem. However this is out of the scope of the dissertation, we refer to [47].

For robustness analysis and synthesis for the control systems with structured uncertainties, the first step which has to be taken is to extract all uncertainties out of the real plant model, such that the real plant model can be described a linear fractional transformation (LFT), as shown in Fig. 4.2. There, Δ_i is a uncertainty block defined in (1.8), z_δ and w_δ are the input and the output of the uncertainty; z_p and w_p are the regulated output and exogenous (command or disturbance) input of the plant. $G(s)$ is just a LTI system. In this plant-uncertainty setup, $\mathcal{T}(0,G) = G_{22}$ may be thought of as the nominal plant model. G_{11}, G_{12}, G_{21} together with the formula $\mathcal{T}(\cdot,\cdot)$ reflect prior knowledge on how the uncertainty affects the nominal plant. This LFT-type plant-uncertainty model is natural for many control problems. Separating out the uncertainty Δ_i from the real plant model P is called uncertainty modeling [51], and is a main topic of

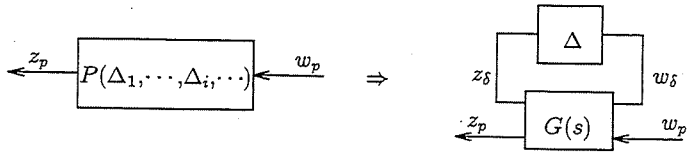


Figure 4.2 Uncertainty modeling.

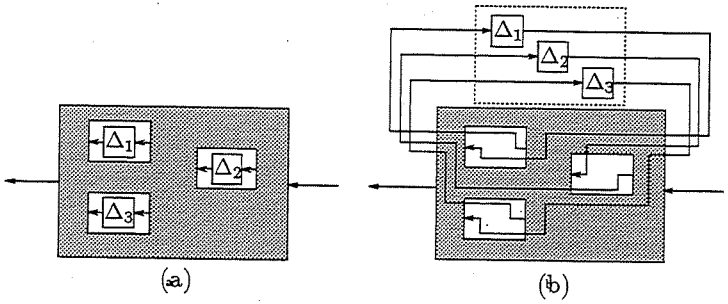


Figure 4.3 Pulling out the Δs.

this chapter.

The general philosophy for uncertainty modeling with LFT is often referred to as "pulling the Δs out" [11]. Consider a structure with three substructures interconnected linearly in some known way (see Fig. 4.3(a)). We can redraw it as a standard LFT, (see Fig. 4.3(b)) via "pulling the Δs out". The matrix G of the LFT can now be obtained by computing the corresponding interconnection matrix in the shadowed box.

4.2 Multidimensional realization method

The parametric uncertainty modeling problem is the uncertainty modeling problem when all uncertainties Δ_i are parametric uncertainties. When these parametric uncertainties enter the real plant in a rational way, the

parametric uncertainty problem turns out to be a multidimensional realization problem. We first give a definition of the realization problem, which is based on the concept of the Linear Fractional Transform (LFT). Then the relations to parametric uncertainty modeling and multiparameter margin formulation will be discussed.

4.2.1 Definition

Definition 12 *Given a multivariable rational matrix (or a multidimensional system transfer matrix) $M(p)$ with dimension $n \times m$, where p contains q parameters $p_1 \cdots p_q$, the multidimensional realization problem is to find a LFT with $\mathcal{T}(\Delta(p), L)$, such that*

$$\boxed{M(p) = \mathcal{T}(\Delta(p), L).} \tag{4.1}$$

where $\Delta(p) \triangleq \mathrm{diag}(p_1 I_{r_1}, \cdots, p_q I_{r_q})$, which can also be represented by a block structure vector $bs = [r_1\ r_2\ \cdots\ r_q]$, r_i being the dimension of the identity matrix I_{r_i}, and

$$\mathcal{T}(\Delta(p), L) = L_{22} + L_{21}\Delta(p)(I - L_{11}\Delta(p))^{-1}L_{12}.$$

$L \in \mathbb{C}^{(r+n)\times(r+m)}$, $r = \sum_{i=1}^{q} r_i$, is a coefficient matrix and partitioned as

$$L = \begin{bmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{bmatrix}.$$

The dimensions of L_{11}, L_{12}, L_{21} and L_{22} are $r \times r$, $r \times m$, $n \times r$ and $n \times m$ respectively.

Note that the elements of the vector p can be integral (or delay) operators or uncertain parameters or both of them. The realization problem is merely to find the coefficient matrix L and block structure bs such that (4.1) holds.

The multidimensional realization problem is originally the problem of realizing a state space model for a multidimensional system transfer function [9]. A lot of work has been done for one and two dimensional realization problems in systems theory [46, 50]. For the case where the entries of $M(p)$ are linear or affine in the components of parameter vector p, a realization algorithm was given by [29, 30]. In general, however, there

is no systematic way to solve this problem when the entries of $M(p)$ are rational functions.

Here we will describe a systematic multidimensional realization algorithm. The algorithm will not necessarily provide a minimal realization from the system point of view, but it does solve the parametric uncertainty modeling problems and the multiparameter margin formulation problems as well. Before going into the details of the algorithm, we show now how parametric uncertainty modeling and multiparameter margin formulation problems can be phrased as multidimensional realization problems.

4.2.2 State space parametric uncertainty modeling

Consider a vector $p = [p_1 \ \cdots \ p_q] \in \mathbb{R}^q$, containing q scalar uncertainty parameters. Let the model of the perturbed system be given as a state space realization in which the entries of the state space matrices are multidimensional rational functions of the parameter vector p:

$$\begin{bmatrix} \dot{x} \\ y \end{bmatrix} = \begin{bmatrix} A(p) & B(p) \\ C(p) & D(p) \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix}. \tag{4.2}$$

To analyze the robustness of such systems by the μ analysis method [29, 80], one should first solve the parametric uncertainty modeling problem. That is, the system of (4.2) must be presented by a LFT. To do that, define the matrix:

$$M(p) = \begin{bmatrix} A(p) & B(p) \\ C(p) & D(p) \end{bmatrix}.$$

The nominal part of the state space model is given by $M_{nom} \triangleq M(0)$. The uncertainty part of the state space model is defined as $M_\Delta(p)$ with entries $[M_\Delta]_{ij}(p) = M_{ij}(p) - [M_{nom}]_{ij}$. By such uncertainty extraction, the perturbed state space model (4.2) can be written as:

$$\begin{bmatrix} \dot{x} \\ y \end{bmatrix} = M_{nom} \begin{bmatrix} x \\ u \end{bmatrix} + M_\Delta(p) \begin{bmatrix} x \\ u \end{bmatrix}. \tag{4.3}$$

Define a new input u_Δ and a new output y_Δ, and let the output y_Δ be fed back to the input u_Δ through a diagonal perturbation

$$\Delta(p) = \text{blockdiag}(p_1 I_{r_1}, \ \cdots \ , p_q I_{r_q}).$$

Furthermore, suppose that matrices B_Δ, C_Δ and D_Δ are constructed such that the following equations hold:

$$\begin{bmatrix} \dot{x} \\ y \\ y_\Delta \end{bmatrix} = \begin{bmatrix} M_{nom} & B_\Delta \\ C_\Delta & D_\Delta \end{bmatrix} \begin{bmatrix} x \\ u \\ u_\Delta \end{bmatrix}, \tag{4.4}$$

$$u_\Delta = \Delta(p) y_\Delta.$$

B_Δ, C_Δ and D_Δ contain information how the uncertainties affect the nominal model. Obviously it must be possible to reduce (4.4) to (4.2). Therefore eliminating u_Δ and y_Δ in (4.4), we have

$$\begin{bmatrix} \dot{x} \\ y \end{bmatrix} = M_{nom} \begin{bmatrix} x \\ u \end{bmatrix} + B_\Delta \Delta(p)[I - D_\Delta \Delta(p)]^{-1} C_\Delta \begin{bmatrix} x \\ u \end{bmatrix}. \tag{4.5}$$

(4.5) is equivalent to (4.2), if the following equation holds:

$$M(p) = M_{nom} + B_\Delta \Delta(p)[I - D_\Delta \Delta(p)]^{-1} C_\Delta. \tag{4.6}$$

If we let $L_{22} = M_{nom}$, $L_{21} = B_\Delta$, $L_{12} = C_\Delta$ and $L_{11} = D_\Delta$, (4.6), compared with (4.2), shows that the state space parametric uncertainty modeling problem coincides with a multidimensional realization problem.

4.2.3 Formulation of the multiparameter margin problem

In this section, we solve the formulation of the multiparameter margin problem left hanging in section 3.1.2. From Theorem 9, let $B_i = L(A_i)L(A_0)^{-1}$, $i = 1, \cdots, l$, $B_0 = I$, and denote $B(p) = \sum_{i=0}^{l} f_i(p) B_i$. Now we try to find a constant matrix M_b and a block structure $bs = [r_1, \cdots, r_q]$ such that:

$$\det(B(p)) = 0 \Leftrightarrow \det(I + M_b \Delta(p)) = 0, \tag{4.7}$$

where

$$\Delta(p) = \text{blockdiag}(p_1 I_{r_1}, \ \cdots \ , p_q I_{r_q}).$$

For the case where the $f_i(p)$ are linear in p, the algorithm for constructing M_b and bs is given in [30]. In [30] an example was also given to show

how to construct M_b and bs when $f_i(p)$ are polynomials in the elements of vector p.

The problem of constructing M_b and bs in the right side of (4.7) from the left side (4.7) is actually nothing more than a multidimensional realization problem, which can be completely solved by the algorithm to be described below. To see this, assume that the following realization problem be solved:

$$\sum_{i=1}^{l} f_i(p)B_i = L_{22} + L_{21}\Delta(p)[I - L_{11}\Delta(p)]^{-1}L_{12}. \tag{4.8}$$

Then the determinant in the left side of (4.7) becomes:

$$\det\{I + L_{22} + L_{21}\Delta(p)[I - L_{11}\Delta(p)]^{-1}L_{12}\} = 0. \tag{4.9}$$

If $\det(I + L_{22}) \neq 0$ (from what follows, it will be clear that L_{22} is a zero matrix in this case) and $\det[I - L_{11}\Delta(p)] \neq 0$ (which must be checked after the parameter margin is obtained), and by manipulating the matrices in (4.9) and using the equality: $\det(I + AB) = \det(I + BA)$. (4.9) is equivalent to

$$\det\{I + (-L_{11} + L_{12}[I + L_{22}]^{-1}L_{21})\Delta(p)\} = 0.$$

The problem is clearly solved if we let:

$$M_b = -L_{11} + L_{12}(I + L_{22})^{-1}L_{21}. \tag{4.10}$$

4.2.4 Preliminary conditions

In the rest of the chapter, we assume that each entry $[M(p)]_{ij}$ of the rational matrix $M(p)$ satisfies the following condition:

$$[M(p)]_{ij} = c_{ij} + f_{ij}(p), \tag{4.11}$$

with c_{ij} a constant and $f_{ij}(p)$ is of the form:

$$\frac{k(p)}{1 + l(p)}, \quad k(0) = l(0) = 0.$$

This assumption is based on the following facts: For the elements in p which are integral operators(s^{-1}) or delay operators(z^{-1}), (4.11) means that the corresponding system is physically causal (the transfer function is proper). For the elements in p which are parametric uncertainties, (4.11) shows that the uncertainty part of $M(p)$ diminishes when the uncertainty parameters are equal to zero.

4.3 Algorithm

The main idea of the algorithm for the multidimensional realization is to realize the affine part of a rational matrix at first, then the rational function part in the second step. The two realizations are finally "cascaded" via the Redheffer Star Product.

It is obvious that $M(p)$ can be written as an affine combination in rational functions:

$$M(p) = M_0 + \sum_{i=1}^{l} f_i(p)M_i, \tag{4.12}$$

where
$M_0 \in \mathbb{C}^{n\times m}$: constant matrix with entries $[M]_{ij} = c_{ij}$ (see (4.11)).
$f_i(p)$, $i = 1, \cdots, l$: multivariable rational functions, each of $f_i(p)$ is just one of $f_{ij}(p)$ in (4.11), l ($\leq nm$) is the number of the rational functions $f_{ij}(p)$ which are different.
$M_i \in \mathbb{C}^{n\times m}$, $i = 1, \cdots, l$: constant matrices, which tell how the $f_i(p)$ enter the matrix $M(p)$.

The first step of the algorithm is to find a coefficient matrix L_a and a block structure bs_a such that:

$$M(p) = L_{a22} + L_{a21}\Delta(f)[I - L_{a11}\Delta(f)]^{-1}L_{a12}, \tag{4.13}$$

where

$$L_a = \begin{bmatrix} L_{a11} & L_{a12} \\ L_{a21} & L_{a22} \end{bmatrix},$$

$\Delta(f) = \mathrm{diag}[f_1(p)I_{r_{a1}}, \cdots, f_l(p)I_{r_{al}}]$, and $bs_a = [r_{a1}, \cdots, r_{al}]$.

The second step is to find a coefficient matrix L_b and a block structure bs such that:

$$\Delta(f) = L_{b22} + L_{b21}\Delta(p)[I - L_{b11}\Delta(p)]^{-1}L_{b12}, \tag{4.14}$$

where

$$L_b = \begin{bmatrix} L_{b11} & L_{b12} \\ L_{b21} & L_{b22} \end{bmatrix},$$

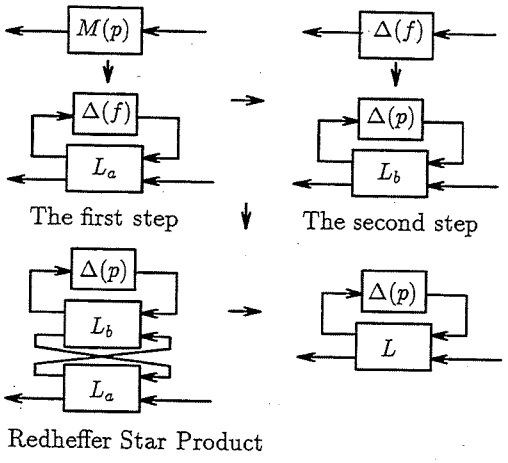


Figure 4.4 Description of the two step realization algorithm. First $M(p)$ is realized as a LFT with a diagonal rational $\Delta(f)$. $\Delta(f)$ is then realized as a LFT with a block diagonal $\Delta(p)$. The two realizations are then "cascaded" via a Redheffer Star product.

$$\Delta(p) = \text{diag}(p_1 I_{r_1}, \cdots, p_q I_{r_q}) \text{ and } bs = [r_1, \cdots, r_q].$$

Based on the coefficient matrices L_a and L_b, the coefficient matrix L in the final realization of (4.1) can be easily constructed by the Redheffer Star-Product [29]: $L = \mathcal{T}(L_b, L_a, r_a, r_a)$ with $r_a = \sum_{i+1}^{l} r_{ai}$ and:

$$\mathcal{T}(L_b, L_a, r_a, r_a) = \begin{bmatrix} L_{b11} + L_{b12}L_{a11}(I - L_{b22}L_{a11})^{-1}L_{b21} & L_{b12}(I - L_{a11}L_{b22})^{-1}L_{a12} \\ L_{a21}(I - L_{b22}L_{a11})^{-1}L_{b21} & L_{a22} + L_{a21}L_{b22}(I - L_{a11}L_{b22})^{-1}L_{a12} \end{bmatrix}. \quad (4.15)$$

The corresponding block structure vector bs for L is the same as that obtained in the second step. This two step procedure is depicted in Fig. 4.4.

4.3.1 Realization: affine combination

The first realization step is just to realize the affine part of $M(p)$, taking $f_i(p)$ as independent variables or parameters. Because the entries of $M(p) = M(f(p))$ are linear in the vector $f(p) = [f_1(p), \cdots, f_l(p)]$, we can use the same method as that used in the case where the entries in $M(p)$ are linear in the vector p. For this linear case, the algorithm is based on Safonov and Athans' Internal Feedback Loop (IFL) parameter representation [66, 30, 29]. To do that, first, take the LR decomposition for each $n \times m$ matrix of M_i, $i = 1, \cdots, l$, from its singular value decomposition:

$$M_i = U_i \Sigma_i V_i^T, \quad (4.16)$$

where U_i and V_i are $n \times r_{ai}$ and $r_{ai} \times m$ matrices respectively, $r_{ai} = rank(M_i)$. Σ_i is a $r_{ai} \times r_{ai}$ positive definite diagonal matrix. Then let:

$$L_i = U_i \Sigma_i^{\frac{1}{2}}, \; R_i = \Sigma_i^{\frac{1}{2}} V_i^T, \quad (4.17)$$

such that $M_i = L_i R_i$ or $f_i(p) M_i = L_i [f_i(p) I_{r_{ai}}] R_i$. Finally construct:

$$U = [L_1, \cdots, L_l] \; and \; V = [R_1^T, \cdots R_l^T]^T. \quad (4.18)$$

Now (4.1) becomes:

$$M(p) = M_0 + \sum_{i=1}^{l} L_i [f_i(p) I_{r_{ai}}] R_i = M_0 + U \Delta(f) V^T,$$

where $\Delta(f)$ is defined in (4.13). Comparing this equation with (4.13), it is easy to find that:

$$L_{a11} = 0, \; L_{a12} = V^T, \quad (4.19)$$

$$L_{a21} = U, \; L_{a22} = M_0.$$

The corresponding block structure is $bs_a = [r_{a1}, \cdots, r_{al}]$ with r_{ai} found as above. Note that L in (4.15) can be simplified significantly as a result of $L_{a11} = 0$:

$$L = \begin{bmatrix} L_{b11} & L_{b12}L_{a12} \\ L_{a21}L_{b21} & L_{a22} + L_{a21}L_{b22}L_{a12} \end{bmatrix}. \quad (4.20)$$

4.3.2 Realization: rational function

The second step is to realize the rational function part. This step is not so trivial as the first one. The algorithm for this step consists of three sub-steps. Before going into detail, we define a permutation matrix which is related to another integer matrix and will be frequently used later. Let B be an $h \times q$ matrix with integer entries b_{ij}, and

$$s_{ij} = \sum_{k=1}^{(i-1)} \sum_{l=1}^{q} b_{kl} + \sum_{l=1}^{j} b_{il}, \quad i = 1, \cdots, h; \; j = 1, \cdots, q,$$

$$s_{i0} = s_{(i-1)q}, \; i = 2, \cdots, h, \; s_{10} = 0,$$

and $s_{hq} \times (s_{ij} - s_{i(j-1)})$ matrices:

$$E_{ij} = [e_{s_{i(j-1)}+1}, \cdots, e_{s_{ij}}], \quad i = 1, \cdots h; \; j = 1, \cdots q,$$

where e_i is a column vector with zero entries except that the ith entry is 1. Now define a permutation matrix P_B as follows:

$$P_B = [E_{11} \cdots E_{h1} \cdots \cdots E_{1q} \cdots E_{hq}]. \tag{4.21}$$

Obviously, the matrix P_B depends on the matrix B, the rows of which are the block structure vectors (see later). The following example is given to show clearly how the permutation matrix P_B is formed.

Example 3 *Let a 2×3 matrix B be given as:*

$$B = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}.$$

Then from the definition above:

$$s_{10} = 0, \quad s_{11} = 1, \quad s_{12} = 3, \quad s_{13} = 6,$$
$$s_{20} = 6, \quad s_{21} = 10, \quad s_{22} = 15, \quad s_{23} = 21,$$

and

$$\begin{aligned}
E_{11} &= [e_{s_{10}+1}, \cdots, e_{s_{11}}] = e_1, \\
E_{12} &= [e_{s_{11}+1}, \cdots, e_{s_{12}}] = [e_2 \; e_3], \\
E_{13} &= [e_{s_{12}+1}, \cdots, e_{s_{13}}] = [e_4 \; e_5 \; e_6], \\
E_{21} &= [e_{s_{20}+1}, \cdots, e_{s_{21}}] = [e_7 \; e_8 \; e_9 \; e_{10}], \\
E_{22} &= [e_{s_{21}+1}, \cdots, e_{s_{12}}] = [e_{11} \; e_{12} \; e_{13} \; e_{14} \; e_{15}], \\
E_{23} &= [e_{s_{22}+1}, \cdots, e_{s_{23}}] = [e_{16} \; e_{17} \; e_{18} \; e_{19} \; e_{20} \; e_{21}].
\end{aligned}$$

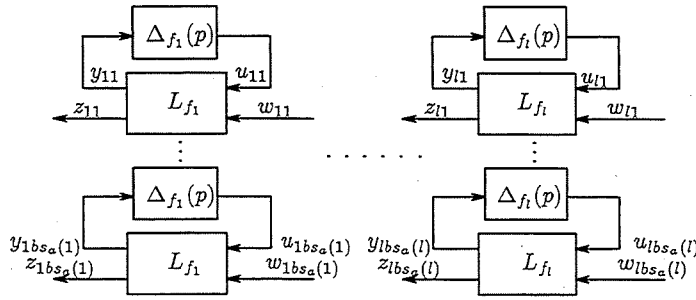


Figure 4.5 The LFT for $\Delta(f)$ by lumping all LFTs for rational functions (f_i).

Finally we can form the corresponding permutation matrix as:

$$\begin{aligned}
P_B \;=\; & [e_1 \; e_7 \; e_8 \; e_9 \; e_{10} \; e_2 \; e_3 \; e_{11} \; e_{12} \; e_{13} \\
& e_{14} \; e_{15} \; e_4 \; e_5 \; e_6 \; e_{16} \; e_{17} \; e_{18} \; e_{19} \; e_{20} \; e_{21}].
\end{aligned}$$

L_b and bs for $\Delta(f)$

Assume that the rational functions $f_i(p)$, $i = 1, \cdots l$, have their LFTs with corresponding coefficient matrices L_{f_i} and block structure vectors bs_{f_i} respectively. Assume also that the block structure vector bs_a is already known from the last section. Then we can form the LFT for $\Delta(f)$ by lumping all LFTs for the $f_i(p)$, each repeated $bs_a(i)$ times, see Fig. 4.5. First, we write the aggregate system consisting of the appended LFTs as:

$$\begin{bmatrix} y_{11} \\ z_{11} \\ \vdots \\ y_{1bs_a(1)} \\ z_{1bs_a(1)} \\ \vdots \\ \vdots \\ y_{l1} \\ z_{l1} \\ \vdots \\ y_{lbs_a(l)} \\ z_{lbs_a(l)} \end{bmatrix} = L_b' \begin{bmatrix} u_{11} \\ w_{11} \\ \vdots \\ u_{1bs_a(1)} \\ w_{1bs_a(1)} \\ \vdots \\ \vdots \\ u_{l1} \\ w_{l1} \\ \vdots \\ u_{lbs_a(l)} \\ w_{lbs_a(l)} \end{bmatrix},$$

$$u_{ij} = \Delta_{f_i}(p) y_{ij}, \quad i = 1, \cdots, l; \; j = 1, \cdots bs_a(i).$$

where $\Delta_{f_i}(p)$ has the block structure bs_{f_i} and

$$L_b' = \text{block diag}(I_{bs_a(i)} \otimes L_{f_1}, \cdots I_{bs_a(l)} \otimes L_{f_l}), \tag{4.22}$$

where $\otimes$ denotes the Kronecker Product. However L_b' is not equal to L_b, as its rows (and columns) corresponding to the repeated feedback inputs(outputs) and exogenous inputs(outputs) are mixed together. But L_b can be obtained by a permutation as follows. Define an $l \times (q+1)$ integer matrix:

$$B = \begin{bmatrix} b_1 \\ \vdots \\ b_l \end{bmatrix}, \text{ with } b_i = \left.\begin{bmatrix} bs_{f_i} & 1 \\ \vdots & \vdots \\ bs_{f_i} & 1 \end{bmatrix}\right\} bs_a(i) \text{ rows.} \tag{4.23}$$

Then

$$L_b = P_B^T L_b' P_B. \tag{4.24}$$

P_B is defined in (4.21) with B in (4.23). Note that permuting L_b' by P_B is nothing more than putting together rows (columns) which correspond to the repeated feedback inputs(outputs) or exogenous inputs(outputs). It is easy to construct the block structure vector bs:

$$bs = \sum_{i=1}^{l} bs_a(i) bs_{f_i}. \tag{4.25}$$

Note that from the preliminary condition on the matrix $M(p)$, $f_i(0) = 0$, so $\Delta(f)_{p=0} = 0$, this leads to L_{b22}=0, see (4.14). Then the coefficient matrix L can be further simplified from (4.20) to

$$L = \begin{bmatrix} L_{b11} & L_{b12}L_{a12} \\ L_{a21}L_{b21} & L_{a22} \end{bmatrix}. \tag{4.26}$$

L_f and bs_f for one rational function

The coefficient matrix L_f and block structure vector bs_f of the LFT for one rational function $f(p)$ are constructed here in the assumption that all LFTs of monomials in its numerator and denominator polynomials $k(p)$ and $l(p)$ are known. First, the coefficient matrix L_k and block structure vector bs_k for $k(p)$ will be constructed. Let the monomials of $k(p)$ be $m_j(p)$, $j = 1, \cdots s$, s is the number of the monomials in $k(p)$, that is:

$$k(p) = \sum_{j=1}^{s} m_j(p),$$

and L_{m_j} and bs_{m_j}, $j = 1, \cdots s$ be the coefficient matrices and the block structure vectors of LFTs for the corresponding monomials $m_j(p)$. So what we try to do now is to find a LFT with L_k and bs_k for the linear combination of all LFTs for $m_j(p)$, see Fig. 4.6. It is easy to verify:

$$\begin{bmatrix} y_1 \\ \vdots \\ y_s \\ z \end{bmatrix} = L_k' \begin{bmatrix} u_1 \\ \vdots \\ u_s \\ w \end{bmatrix},$$

$$y_j = \Delta_j(p) u_j, \quad i = 1, \cdots, s,$$

where

$$L_k' = \begin{bmatrix} (L_{m_1})_{11} & 0 & \cdots & \cdots & 0 & (L_{m_1})_{12} \\ 0 & \ddots & \ddots & & \vdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ \vdots & & \ddots & \ddots & 0 & \vdots \\ 0 & \cdots & \cdots & 0 & (L_{m_s})_{11} & (L_{m_s})_{12} \\ (L_{m_1})_{21} & \cdots & \cdots & \cdots & (L_{m_s})_{21} & \sum_{j=1}^{s} (L_{m_j})_{22} \end{bmatrix}, \tag{4.27}$$

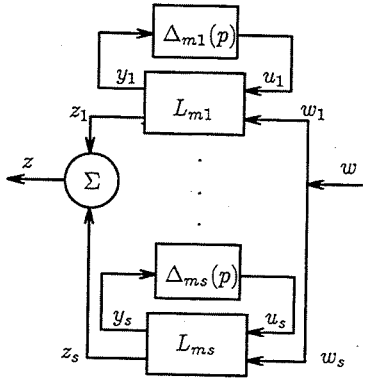


Figure 4.6 A polynomial LFT as a linear combination of monomial LFTs.

and $\Delta_j(p)$ has the block structure vector bs_{m_j}. The next step is to permute L'_k such that the repeated feedback inputs (outputs) are arranged together. Define the $s \times (q+1)$ integer matrix:

$$B = (b_1^T, \cdots, b_s^T)^T, \tag{4.28}$$

with

$$b_j = [bs_{m_j}\ 0],\ \ j = 1, \cdots, (s-1), b_s = [bs_{m_s}\ 1].$$

Using the permutation matrix P_B defined in (4.21) with B in (4.28), we have:

$$L_k = P_B^T L'_k P_B. \tag{4.29}$$

It is trivial to get the block structure vector bs_k:

$$bs_k = \sum_{j=1}^{s} bs_{m_j}. \tag{4.30}$$

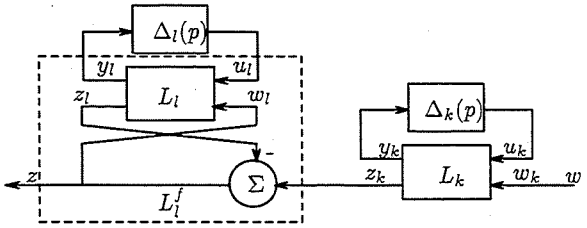


Figure 4.7 A rational function LFT is a linear combination of two polynomial LFTs.

We can use the same method to construct the coefficient matrix L_l and the block structure vector bs_l of the LFT for the polynomial $l(p)$.

Now the LFT of L_f and bs_f can be constructed as a linear combination of LFTs of L_k, bs_k, L_l and bs_l, see Fig. 4.7, for $f(p)$ is the product of two rational functions:

$$f(p) = k(p)[\frac{1}{1+l(p)}],$$

and $\frac{1}{1+l(p)}$ is just a closed loop function with the feedback gain $-l(p)$ and forward gain 1. It is easy to verify that the coefficient matrix of the LFT for $\frac{1}{1+l(p)}$ is:

$$L_l^f = \begin{bmatrix} L_{l11} - L_{l12}EL_{l21} & L_{l12}E \\ -EL_{l21} & E \end{bmatrix},\ \ E = (I + L_{l22})^{-1}, \tag{4.31}$$

and the corresponding block structure vector is just bs_l. For the series connection of the two LFTs, we have:

$$\begin{bmatrix} y_l \\ y_k \\ z \end{bmatrix} = L'_f \begin{bmatrix} u_l \\ u_k \\ w \end{bmatrix},$$

$$u_l = \Delta_l(p)y_l,\ \ u_k = \Delta_k(p)y_k,$$

where

$$L'_f = \begin{bmatrix} L^f_{l11} & L^f_{l12}L_{k21} & L^f_{l12}L_{k22} \\ 0 & L_{k11} & L_{k12} \\ L^f_{l21} & L^f_{l22}L_{k21} & L^f_{l22}L_{k22} \end{bmatrix}. \tag{4.32}$$

Again, permutation is needed to arrange the repeated feedback inputs (outputs) together. This time, the integer matrix B is defined as:

$$B = \begin{bmatrix} bs_l & 0 \\ bs_k & 1 \end{bmatrix}. \tag{4.33}$$

Then

$$L_f = P_B^T L'_f P_B. \tag{4.34}$$

The block structure vector bs_f is:

$$bs_f = bs_l + bs_k. \tag{4.35}$$

L_m and bs_m for one monomial

Let a monomial have the form: $ap_1^{k_1} \cdots p_q^{k_q}$, where a is the coefficient of the monomial and k_i $(i = 1, \cdots, q)$ is the degree of the monomial with respect to p_i. The total degree of the monomial is

$$k = \sum_{i=1}^{q} k_i.$$

This monomial, as a transfer function, is depicted as a series of blocks in Fig. 4.8. By pulling all variables or parameters out (see the dashed line), it is trivial to construct the $(k+1) \times (k+1)$ coefficient matrix L_m and the block structure vector bs_m of the LFT for the monomial as:

$$L_m = \begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 & a \\ 1 & \ddots & & & \vdots & 0 \\ 0 & \ddots & \ddots & & \vdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ \vdots & & \ddots & \ddots & 0 & \vdots \\ 0 & \cdots & \cdots & 0 & 1 & 0 \end{bmatrix}, \tag{4.36}$$

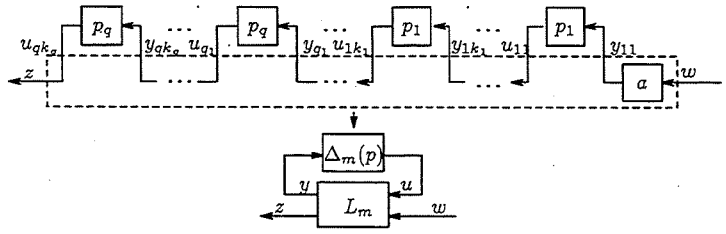


Figure 4.8 A monomial LFT as a linear combination of its variables.

and

$$bs_m = [k_1 \ \cdots \ k_q]. \tag{4.37}$$

The coefficient a and the degrees k_i of a monomial can be directly obtained from the input data. The second realization step for L_b and bs is completed here, for we can construct L_m and bs_m for each monomial from the input data $(a, k_i, \forall i)$ first, then L_f and bs_f for each rational function from the corresponding L_m and bs_m, and finally L_b and bs_b for $\Delta(f)$ from all of L_f and bs_f.

4.3.3 Data structure and program code

In order to make a computer program for the realization problem with the above algorithm, the data structure for representation of the multivariable rational matrix $M(p)$ has to be decided first. Here we give, as an example, the data structure with two matrices. The first one is the $n \times lm$ matrix M defined as(see Sec. 3):

$$M = [M_0 \ M_1 \ \ldots \ M_l].$$

The matrix M tells how each rational function $f_i(p)$ enters the matrix $M(p)$. The second one, which tells the structures of all rational functions, is defined as:

$$N = [n_1^T, \ \ldots \ n_h^T]^T,$$

with each row including all data for one monomial:

$$n_r = [i,\ j,\ a_{ik},\ bs_{mik}],\ \ r = 1,\ \ldots\ , h\ ,$$

where
i: integer, denotes that the monomial is in the ith rational function $f_i(p)$.
j: integer, denotes if the monomial is numerator(j=0) or denominator(j=1).
k: integer, denotes that the monomial is in the kth position.
a_{ik}: $\in \mathbb{C}$, the coefficient of the monomial.
bs_{mik}: integer vector, the elements of it denote the degrees of corresponding variables. It is equal to the block structure vector for the monomial.
h: integer, total number of the monomials.

Here is an example to show how the input matrices are formed.

Example 4 *Assume that a 2×2 rational matrix $M(p)$ is:*

$$M(p) = \begin{bmatrix} 1+\frac{p_2}{1+2p_1} & \frac{5p_1p_2}{1+6p_1p_2} \\ \frac{5p_1p_2}{1+6p_1p_2} & 1+\frac{3p_1}{1+4p_2} \end{bmatrix}.$$

The we have the input matrix $M = [M_0\ M_1\ M_2\ M_3]$, where

$$M_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ M_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix},\ M_2 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix},\ M_3 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

The corresponding rational functions are:

$$f_1(p) = \frac{p_2}{1+2p_1},\ \ f_2(p) = \frac{3p_1}{1+4p_2} \ \text{ and } \ f_3 = \frac{5p_1p_2}{1+6p_1p_2}.$$

Therefore, the input matrix N is:

$$N = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 2 & 1 & 0 \\ 2 & 0 & 3 & 1 & 0 \\ 2 & 1 & 4 & 0 & 1 \\ 3 & 0 & 5 & 1 & 1 \\ 3 & 1 & 6 & 1 & 1 \end{bmatrix}.$$

With the input data M and N defined above , we can now summarize the algorithm[1] as:

[1]The code is available in MATLAB as a function file.

Algorithm 6 (Multidimension realization algorithm)

1. *Input data, M and N.*
2. *Compute L_a and bs_a from M, using (4.16, 4.17, 4.18, 4.19).*
3. *Compute L_b and bs_b from N and bs_a.*
   - 3.1. *Compute L_m and bs_m from N using (4.36, 4.37).*
   - 3.2. *Compute L_k, L_l, bs_k and bs_l from L_m and bs_m using (4.27, 4.28, 4.29, 4.30).*
   - 3.3. *Compute L_f and bs_f from L_k, bs_k and L_l, bs_l using (4.31, 4.32, 4.33, 4.34, 4.35).*
   - 3.4. *Compute L_b and bs from $L_f(s)$, $bs_f(s)$ and bs_a using (4.22, 4.23, 4.24, 4.25).*
4. *Compute L from (4.26).*

4.4 Size reduction

As in the case of one dimensional realization, there is minimum realization problem for multidimensional realization, which is defined as follows:

> Solve the realization problem of Definition 12, such that the dimension of L_{11} (or $\Delta(p)$) is minimized.

As mentioned already, the algorithm above does in general not give the minimal realization, that is, the size of the result $\Delta(p)$ is reducible. If we take each rational function entry in $M(p)$ as a $f_i(p)$, then the rank r_{ai} of M_i is 1 and the algorithm will give a block structure vector:

$$bs = \sum_{i=1}^{h} n_i(4 : q+3),$$

and the size of $\Delta(p)$ is:

$$\sum_{i=1}^{q} bs(i).$$

Clearly this size is larger then the minimal one in the case $j = 1$.

For the case of one dimension, the minimum realization problem can be solved by transforming the original realization to the controllability and observability staircase forms, this transformation isolates the uncontrollable and unobservable states (each appearance of the parameter corresponding one state), which are then removed from the model. The minimum size of L_{11} is equal to the McMillan Degree of the given rational matrix $M(p)$ (suppose it is proper) [46]. The minimal realization problem for multidimensional systems ($j > 1$), however, is known as a very difficult problem in system theory even for 2D systems [50].

Here we give two heuristic methods which can lead to a reduction of the dimension of $\Delta(p)$, but not necessarily minimum.

4.4.1 Reducing size of each block one by one

As mentioned earlier, for the case of one dimension, the minimum realization problem has been solved completely by removing all uncontrollable and unobservable states. Here we show the minimum realization for one block, while keeping the sizes of other blocks constant. The idea was also explored in [51].

Theorem 10 *Given a multidimensional system $M(p)$ and its realization with L and $bs = [r_1, \cdots, r_q]$. If one dimensional system realization with L and $bs = r_1$ has a minimal realization L_m with $bs = r_{1m}$, $r_{1m} < r_1$, then $M(p)$ has the realization L_m and $bs = [r_{1m}, r_2, \cdots, r_q]$.*

Proof: Suppose

$$L = \begin{bmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{bmatrix}.$$

Partition L_{11}, L_{12} and L_{21} as follows:

$$L = \begin{bmatrix} A & B_1 & B_2 \\ C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{bmatrix},$$

where A is of dimension $r_1 \times r_1$, and

$$L_{11} = \begin{bmatrix} A & B_1 \\ C_1 & D_{11} \end{bmatrix}, \; L_{12} = \begin{bmatrix} B_1 \\ D_{21} \end{bmatrix},$$

$$L_{21} = \begin{bmatrix} C_1 & D_{21} \end{bmatrix}, \; L_{22} = D_{22}.$$

Now consider a one dimensional realization of L with block structure r_1, which is just a normal state space realization with state matrix A and input matrix $B = \begin{bmatrix} B_1 & B_2 \end{bmatrix}$, output matrix $C = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix}$ and the direct transfer matrix $D = \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix}$. Form system theory [46], we can always find a similar transformation T such that

$$TAT^{-1} = \begin{bmatrix} A_{uc} & 0 \\ * & A_c \end{bmatrix}, \; TB = \begin{bmatrix} TB_1 & TB_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ B_{1c} & B_{2c} \end{bmatrix},$$

$$CT^{-1} = \begin{bmatrix} C_1T^{-1} \\ C_2T^{-1} \end{bmatrix} = \begin{bmatrix} * & C_{1c} \\ * & C_{2c} \end{bmatrix}.$$

where A_c and B_c is controllable and A_c is of dimension $r_{1c} \times r_{1c}, r_{1c} \leq r_1$, $*$ denotes some matrix. Define a new similarity transformation $\bar{T}$ as:

$$\bar{T} \triangleq \begin{bmatrix} T & 0 \\ 0 & I \end{bmatrix}.$$

Obviously:

$$\bar{T}L_{11}\bar{T}^{-1} = \begin{bmatrix} A_{uc} & 0 & 0 \\ * & A_c & B_{1c} \\ * & C_{1c} & D_{11} \end{bmatrix},$$

$$\bar{T}L_{12} = \begin{bmatrix} 0 \\ B_{2c} \\ D_{12} \end{bmatrix}, \; L_{21}\bar{T}^{-1} = \begin{bmatrix} * & C_{2c} & D_{21} \end{bmatrix},$$

and $\bar{T}$ commutes with $\Delta(p)$: $\Delta(p)\bar{T} = \bar{T}\Delta(p)$. Thus it can be easily verified that

$$\begin{aligned} M(p) &= L_{22} + L_{21}\bar{T}^{-1}\bar{T}\Delta(p)(I - L_{11}\Delta(p))^{-1}\bar{T}^{-1}\bar{T}L_{12} \\ &= L_{22} + L_{21}\bar{T}^{-1}\Delta(P)(I - \bar{T}L_{11}\bar{T}^{-1}\Delta(p))^{-1}\bar{T}L_{12} \\ &= L_{22} + L_{21c}\Delta_c(p)(I - L_{11c}\Delta_c(p))^{-1}L_{12c}, \end{aligned}$$

with

$$L_{11c} = \begin{bmatrix} A_c & B_{1c} \\ C_{1c} & D_{11} \end{bmatrix}, \; L_{12c} = \begin{bmatrix} B_{2c} \\ D_{12} \end{bmatrix}, \; L_{21c} = \begin{bmatrix} C_{2c} & D_{21} \end{bmatrix},$$

and $\Delta_c(p)$ has the block structure $bs = [r_{1c}\, r_2, \cdots, r_q]$. Thus

$$L_c \triangleq \begin{bmatrix} L_{11c} & L_{12c} \\ L_{21c} & L_{22} \end{bmatrix}, bs = [r_{1c}\, r_2, \cdots, r_q],$$

is also a realization of $M(p)$ but with (possibly) smaller size since $r_{1c} \leq r_1$. Note that

$$L_c = \begin{bmatrix} A_c & B_{1c} & B_{2c} \\ C_{1c} & D_{11} & D_{12} \\ C_{2c} & D_{21} & D_{22} \end{bmatrix},$$

together with the block structure $bs = r_{1c}$ is also a size reduced realization of the one-dimensional realization:

$$L = \begin{bmatrix} A & B_1 & B_2 \\ C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{bmatrix},$$

with $bs = r_1$. If (A, C) is not observable, just do the same thing to remove unobservable modes, then the theorem follows. □

Theorem 10 gives a way to reduce the size for one block. It can actually be done for each of the blocks lexicographically. Note that it will not necessarily give a minimum realization for multidimensional systems.

4.4.2 Reducing the size for $\det(I + M\Delta(p)) = 0$

From Chapter 2 and 3, in both μ and μ_2 calculation, we try to calculate some minimum norm of a parameter vector p subject to

$$\det(I + M\Delta(p)) = 0. \tag{4.38}$$

For robustness analysis, we focus on calculating robust measures like μ and μ_2, instead of complete realization. Thus in each realization step in our algorithm, we can introduce some size reduction procedure to find an equivalent condition for (4.38):

$$\det(I + M_r\Delta_r(p)) = 0, \tag{4.39}$$

where M_r and $\Delta_r(p)$ are of same format as M and $\Delta(p)$ but with smaller size. The reduction algorithm based on Singular Value Decomposition (see Algorithm 3.1 of [33] for details) can be used such that each "block row" and "block column" of the size reduced matrix M_r (partitioned conformally to the block structure of $\Delta_r(p)$) is of full rank.

Now we elaborate this size reduction procedure in realization algorithm as follows. To solve the realization problem of (4.8), let the program input matrices be:

$$M_0 = 0, M_i = B_i.$$

After finishing step 1 and step 2 of the algorithm, we obtained L_{a12}, L_{a21} and bs_a and have $L_{a11} = 0$ and $L_{a22} = M_0 = 0$. Before going to step 3, we can reduce the dimension of $\Delta(f)$. As $L_{a11} = 0$ and $L_{a22} = M_0 = 0$, (4.8) is equivalent to $\det[I + L_{a21}\Delta(f)L_{a12}] = 0$, which is further equivalent to:

$$\det[I + M_a\Delta(f)] = 0, \;\; M_a = L_{a12}L_{a21}. \tag{4.40}$$

By reducing the size of $\Delta(f)$ in (4.40), we have:

$$\det[I + M_{ar}\Delta_r(f)] = 0,$$

where $\Delta_r(f)$ has a reduced block structure vector bs_{ar}. After the size of $\Delta(p)$ is reduced, we can go to step 3 with new L_a and bs_a:

$$L_{a11} = 0, \;\; L_{a12} = I, \;\; L_{a21} = M_{ar}, \;\; L_{a22} = 0 \text{ and } \; bs_a = bs_{ar}.$$

Then from step 3 and 4, we obtain L and bs. By constructing M_b with 4.10, we have:

$$\det[I + M_b\Delta(p)] = 0. \tag{4.41}$$

Again the same reduction procedure can be used to reduce the size of $\Delta(p)$. By this double reduction, the final size of $\Delta(p)$ is reduced significantly as demonstrated in the example in the next section. Note that such a reduction procedure may not be used for the general realization problem as $\sum_{i=1}^{l} f_i(p)B_i \neq M_b\Delta(p)$.

4.5 Examples

In this section, we give two examples to show how the algorithm and the size reduction methods works.

Example 5 *This example is to solve a general realization problem with the rational matrix $M(p)$ the same as in Example 4. Using the input matrices M and N obtained in Example 4. We can obtain, from the program, the following LFT coefficient matrices and block structure vectors:*

$$L_a = \left[\begin{array}{cccc|cc} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right] \quad \text{and} \quad bs_a = \left[\begin{array}{ccc} 1 & 1 & 2 \end{array}\right]$$

for the affine realization part and

$$L_b = \left[\begin{array}{cccccccccccc|cccc} -2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & -6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & -6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline -2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & -6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & -6 & 0 & 0 & 0 & 0 \end{array}\right]$$

for the rational realization part. After the Redheffer Star Product of L_a and L_b, the final realization has the coefficient matrix as depicted in Fig. 4.9 with the block structure vector $bs = \left[\begin{array}{cc} 6 & 6 \end{array}\right]$.

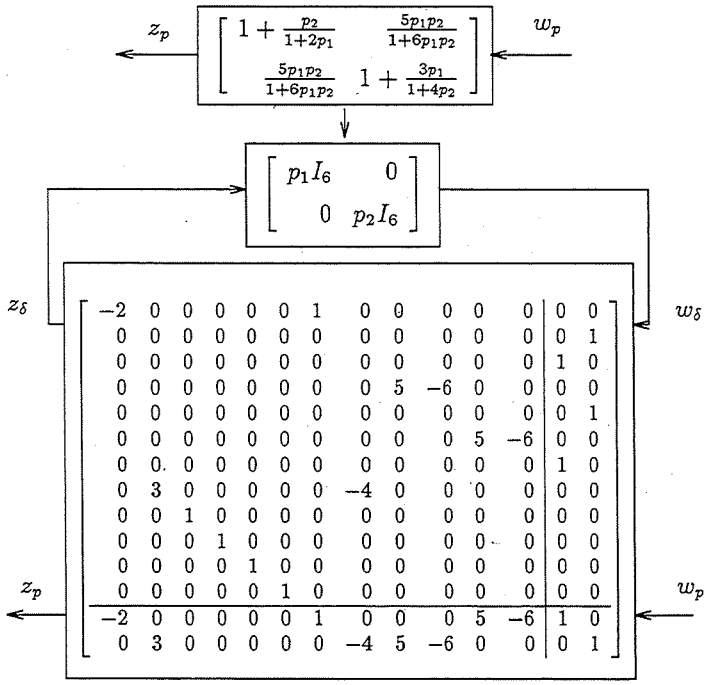


Figure 4.9 The LFT for the realization problem in Example 4.

Now using the size reduction procedure, the size of each block is reduced from 6 to 4, and we have the following realization:

$$bs = \begin{bmatrix} 4 & 4 \end{bmatrix},$$

$$L = \left[\begin{array}{ccccc|cc}
0.0000 & 0.0000 & 0.0000 & 0.0000 & -0.5047 & & \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & & \\
0.0000 & 0.0000 & -2.0000 & 0.0000 & -0.9979 & & \\
0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0203 & & \\
0.0485 & 0.0000 & 0.0000 & -0.0313 & 0.0000 & & \\
-0.0824 & 0.2632 & 0.0000 & -2.0449 & 0.0000 & & \\
-0.7478 & 0.0000 & 0.0000 & 0.4822 & 0.0000 & & \\
-0.0225 & -0.9647 & 0.0000 & -0.5578 & 0.0000 & & \\
\hline
0.0000 & 0.0000 & 2.0000 & 0.0000 & 0.4928 & & \\
0.0854 & 0.0000 & 0.0000 & 2.1196 & 0.0000 & & \\
\hline
0.0000 & 7.7876 & 0.0000 & 0.0000 & 0.0569 & & \\
-1.5791 & 0.0000 & 5.7885 & -0.6402 & 0.0000 & & \\
0.0000 & -0.0647 & 0.0000 & 0.0000 & 0.0000 & & \\
0.0000 & -0.3139 & 0.0000 & 0.0000 & 1.4131 & & \\
0.0000 & 0.0000 & 0.0000 & 0.9979 & 0 & & \\
-3.7230 & 0.0000 & -1.0156 & 0.0000 & 0 & & \\
0.0000 & 0.0000 & 0.0000 & 0.0647 & 0 & & \\
-1.0156 & 0.0000 & -0.2770 & 0.0000 & 0 & & \\
\hline
0.0000 & 7.8586 & 0.0000 & 1.0000 & 0 & & \\
1.8035 & 0.0000 & 8.5876 & 0 & 1.0000 & &
\end{array}\right].$$

Example 6 *This example corresponds to the one step we did not treat in example 2 to solve the formulation problem of the multiparameter margin, see section 4.2.3. From example 2, we have*

$$f_1(p) = p_1 p_2,\ f_2(p) = p_1,\ f_3(p) = p_2,$$

and 45×45 *matrices* B_1, B_2 *and* B_3. *Now start our algorithm with input:*

$$M = \begin{bmatrix} 0_{45\times 45} & B_1 & B_2 & B_3 \end{bmatrix} \text{ and } N = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 2 & 0 & 1 & 1 & 0 \\ 3 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

After step 2, we can form M_a *from* L_a *by (4.40) with* M_a: 27×27 *matrix and* $bs_a = [9\ 9\ 9]$. *The first reduction gives* M_{ar}: 18×18 *matrix and* $bs_{ar} = [6\ 6\ 6]$. *The coefficient matrix* L_b *and the block structure bs are obtained from step 3 with* L_b: 42×42 *matrix and* $bs = [12\ 12]$. *The matrix* M_b *in (4.41) is constructed from L given by step 4 with* M_b: 24×24 *matrix (conformal to* $bs = [12\ 12]$*). Finally by the second reduction,* M_b *and bs are:*

$$M_b = \left[\begin{array}{cccccc|cccccc}
1.9929 & -0.2391 & 0.0747 & 0.1245 & -0.0838 & -0.1007 & & & & & & \\
-9.5041 & 1.1314 & 0.1896 & 0.4209 & 0.1793 & 0.0069 & & & & & & \\
1.7670 & 0.4742 & 0.9628 & -0.1175 & 0.0902 & -0.0457 & & & & & & \\
15.3116 & 1.7784 & -0.3696 & 0.0746 & 0.2095 & -0.1161 & & & & & & \\
10.1655 & -2.6222 & 0.0265 & 0.0984 & 0.1719 & -0.0678 & & & & & & \\
18.9315 & 0.5319 & 0.2194 & 0.0654 & -0.2201 & 0.1460 & & & & & & \\
-0.0671 & -0.0419 & 0.0054 & 0.0259 & -0.0112 & -0.0106 & & & & & & \\
-1.1655 & -0.1205 & -0.2577 & 0.1151 & -0.0387 & 0.0133 & & & & & & \\
0.0472 & -0.1152 & -0.6470 & 0.1433 & -0.0927 & -0.0002 & & & & & & \\
-0.3916 & -0.0731 & -0.2299 & 0.0749 & -0.0355 & 0.0026 & & & & & & \\
1.2426 & -0.0007 & -0.4926 & 0.0432 & -0.0653 & -0.0105 & & & & & & \\
-1.4522 & -0.1311 & -0.7770 & 0.1830 & -0.0823 & 0.0467 & & & & & & \\
\hline
3.7298 & -1.6699 & -0.8111 & -0.3755 & 0.0991 & 0.5748 & & & & & & \\
-27.5936 & 2.1338 & -1.9431 & -0.4148 & 0.7559 & -0.1398 & & & & & & \\
6.1184 & 1.9612 & 0.5615 & -1.7542 & -0.1803 & -0.0402 & & & & & & \\
48.4273 & 5.5715 & 0.3985 & 0.2696 & 0.6219 & 0.1086 & & & & & & \\
24.2354 & -9.4436 & 0.7867 & -0.1839 & 0.4058 & -0.0602 & & & & & & \\
53.4500 & -0.9555 & -1.8426 & -0.0551 & -0.4325 & -0.1086 & & & & & & \\
1.5411 & -0.0981 & -0.0474 & -0.0314 & 0.0058 & 0.0619 & & & & & & \\
-1.9742 & 1.6895 & 0.1627 & 0.3244 & 0.0620 & -0.2106 & & & & & & \\
1.2129 & -0.3618 & 0.9928 & 0.7967 & -0.1273 & -0.2007 & & & & & & \\
-0.3014 & 0.1342 & 0.0450 & 1.2813 & -0.0066 & -0.1051 & & & & & & \\
3.2295 & -1.0823 & -0.1579 & 0.6067 & 0.7888 & -0.0390 & & & & & & \\
-3.4990 & 1.0193 & 0.3525 & 1.0321 & -0.0049 & 0.4138 & & & & & &
\end{array}\right],$$

$bs = [6\ 6]$.

4.6 Conclusions

It was shown in this chapter that the parametric uncertainty modeling can be cast as the realization of a multidimensional system, which is defined in the framework of Linear Fractional Transformations (LFTs). An algorithm to systematically solve the multidimensional realization problem

was given, which is the main contribution of this chapter. The trick we used is to realize the affine part of a rational matrix at first, then the polynomial part in second step. The disadvantage of the algorithm is clearly the large size of the resulting matrix. Consequently, two size reduction procedure were introduced. The first one is based on minimal realization method for one dimensional system and can be used for size reduction of general multidimensional realization; While the second one serves to reduce the size in the problem of $\det(I+M\Delta)=0$, which is used in each realization step. The second reduction procedure is a problem-dependent size reduction and may not be generally used for realization problems. Both size reduction procedures give by no means minimal size. Further research work to investigate the minimality issue is definitely required.

Chapter 5

Uncertainty normalization with LFTs

To analyze the robustness properties of systems with structured uncertainty as shown in Fig. 1.9, we have to normalize all the uncertainty blocks such that they are all non-expansive, that is, the induced 2-norm is smaller than 1. For LTI uncertainty blocks, this requirement reduces to the fact that the H_∞-norm of the uncertainty blocks must be smaller than 1. One motivation for doing this is for extending the range of applicability of the small gain theorem. For example, when the uncertainties are passive or positive real, the small gain condition is not satisfied, thus we need to find a transformation such that the uncertainties are non-expansive (to this end non-expansiveness is not necessary, but L_2 boundedness is sufficient). The other reason is for mathematical convenience in robustness analysis and synthesis when the scaled small gain condition and especially the μ techniques are used for structured uncertainty systems.

This work is based on the sector transformation and loop transformation. In [101], Zames derived a transformation to map a SISO conic sector to a non-expansive sector, while Safonov generalized the transformation for MIMO conic sector and multi-loop cases in [69] and for passive cases in [70]. The loop transformation technique was summarized in [23] by Desoer and Vidyasagar.

In this chapter, we focus on normalization of structured uncertainties via LFT by use of sector transformation and loop transformation techniques. The original feedback system is transformed in such a way that

it is stable if and only if the transformed system is stable and has non-expansive structured uncertainty feedback. In section 5.1, the concepts of a sector, sector transformation, loop shift and multiplier transformations will be recapitulated as preliminaries for the rest of the chapter. In section 5.2, the main results on normalization for unstructured uncertainties in general sectors will be given. In section 5.3 simple methods will be given for normalization for two special sectors: general conic sector and passive sector. Section 5.4 generalizes the results for structured uncertainty cases. Section 5.5 shows by an example how to use the method in this chapter for normalization of uncertainties. Some concluding remarks will be made in section 5.6.

5.1 Preliminaries

In this section, we recapitulate loop shift and multiplier transformations. They will form the basis for our main results. The material is basically from [23] and [84].

5.1.1 Generalized sectors

In practice, uncertainties are not in general given as non-expansive operators. Instead, quite often, the uncertainties are given in some specified sectors. A general sector is defined as follows.

Definition 13 *Let $\Delta : \mathbf{L}_{2e}^n \to \mathbf{L}_{2e}^n$, and $w, z \in \mathbf{L}_{2e}^n$ and $w = \Delta z$. Given four linear causal stable operators S_{11}, S_{12}, S_{21}, S_{22}, if*

$$\langle S_{11}z + S_{12}w, S_{21}z + S_{22}w\rangle_T \le 0$$

for all $T \in \mathbb{R}_+$ and any $z \in \mathbf{L}_{2e}$, then we say

$$\Delta \in \text{sector}[S],$$

where $\langle\cdot,\cdot\rangle_T$ denotes the inner product:

$$\langle z, w\rangle_T \triangleq \int_0^T z^T w dt, \tag{5.1}$$

and

$$S = \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix}. \tag{5.2}$$

This definition actually covers a variety of special sectors. For convenience we will drop the subscript "T" from the inner product notation $\langle \cdot, \cdot \rangle_T$.

5.1.2 Sector transformation

Suppose $\Delta_1 \in \text{sector}[S_1]$ if and only if $\Delta_2 \in \text{sector}[S_2]$, then the map from Δ_1 to Δ_2 is called a sector transformation. A typical format of a sector transformation is an LFT:

$$\Delta_2 = \mathcal{T}(L, \Delta_1),$$

where L is a coefficient matrix or LTI system.

5.1.3 Loop shift transformation

Consider a feedback system of Fig. 5.1(a). By making a suitable transformation of the system, one can significantly expand the range of applicability of the small gain theorem. The idea is to introduce an additional L_2-stable linear operator C which acts as a feed-forward around Δ and, on the other side, as a feedback around G, shown in Fig. 5.1(b). Then the system of Fig. 5.1(a) is L_2-stable if and only if the system of Fig. 5.1(b) is L_2-stable. This idea is summarized in the following lemma:

Lemma 1 *Consider the system of Fig. 5.1(a), and its transformed version of Fig. 5.1(b). Suppose that C is a linear operator and that C is L_2-stable, then the system of Fig. 5.1(a) is L_2-stable if and only if its transformed version of Fig. 5.1(b) L_2-stable.*

The complete proof can be found in [23]. The significance of this lemma is that we have more freedom to use the small gain theorem by playing around with C. Actually suppose that Δ is causal and L_2-stable, by using the small gain theorem to Fig. 5.1(b) and from the loop shift theorem, the system of Fig. 5.1(a) is L_2-stable if there exists a causal linear operator C which is L_2-stable such that 1) $G(I - CG)^{-1}$ is causal and L_2-stable and 2) the small gain condition:

$$\gamma_p[G(I - CG)^{-1}]\gamma_p(\Delta - C) < 1.$$

is satisfied. The causality and linearity requirements on C imply that C has a proper transfer function (suppose that C has a limit dimension).

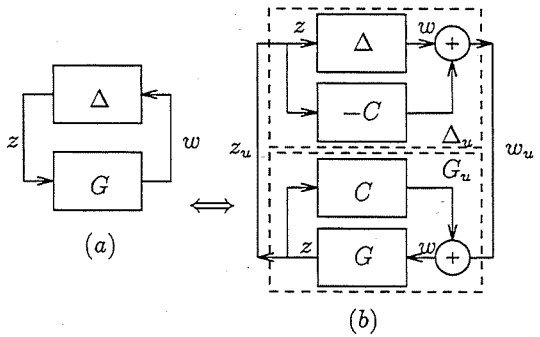


Figure 5.1 Loop shift transformation.

5.1.4 Multiplier transformation

Another way to increase the applicability of the small gain theorem is to use the so-called multiplier transformation. The basic idea is that by multiplying certain operators by appropriately chosen "multipliers", or often called "scalars" or "weights", the products can be doctored to satisfy the small gain conditions.

Lemma 2 *Consider the system of Fig. 5.2(a), and its transformed version of Fig. 5.2(b). Suppose that Δ and G are causal. Suppose also that R, S are causal, linear, and L_2-stable operators, and their inverses R^{-1} and S^{-1} exist and they are causal and L_2-stable, then the system of Fig. 5.2(a) is L_2-stable if and only if the system of Fig. 5.2(b) is L_2-stable.*

The linearity and causality restrictions on R and S are not necessary but we only consider linear and causal cases. Now using the small gain theorem to Fig. 5.2(b), from the multiplier theorem, the system of Fig. 5.2(a) is L_2-stable if there exist R and S, with inverses R^{-1} and S^{-1} that are causal, linear and L_2-stable, such that

$$\gamma_p(R^{-1}\Delta S)\gamma_p(RGS^{-1}) < 1.$$

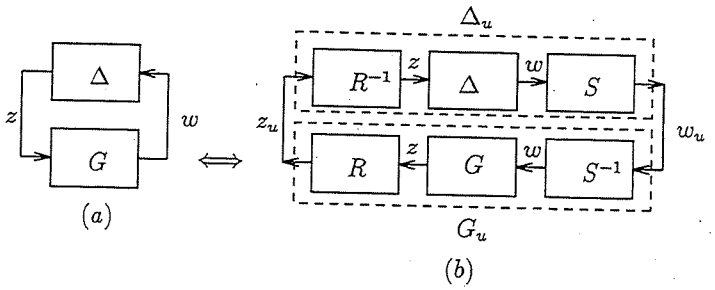


Figure 5.2 Multiplier transformation.

5.2 Normalization of general sectors

In this section we will give our main results. Consider a feedback system of Fig. 5.3(a). Suppose $\Delta \in$ sector[S] and sector[S] are given. Now we are trying to find a transformed system of Fig. 5.3(b), with $\Delta_u = \mathcal{T}(\hat{L}, \Delta)$ and $G_u = \mathcal{T}(L, G)$ such that the system of Fig. 5.3(a) is L_2-stable if and only if Fig. 5.3(b) is L_2-stable and, more than that, Δ_u is non-expansive, that is, $\gamma(\Delta_u) \leq 1$.

Theorem 11 *Consider a feedback system of Fig. 5.3(a) where* $\Delta \in$ sector[S], $G : \mathbf{L}_{2e} \to \mathbf{L}_{2e}$. *Suppose*

$$\begin{bmatrix} I & S_{11} \\ I & S_{21} \end{bmatrix}^{-1}, \begin{bmatrix} -I & -S_{12} \\ I & -S_{22} \end{bmatrix}^{-1} \tag{5.3}$$

are all well-defined, causal and L_2*-stable. Let*

$$\hat{L} = \begin{bmatrix} I & S_{11} \\ I & S_{21} \end{bmatrix}^{-1} \begin{bmatrix} -I & -S_{12} \\ I & -S_{22} \end{bmatrix}, \tag{5.4}$$

$$L = \hat{L}^{-1} = \begin{bmatrix} -I & -S_{12} \\ I & -S_{22} \end{bmatrix}^{-1} \begin{bmatrix} I & S_{11} \\ I & S_{21} \end{bmatrix}. \tag{5.5}$$

If $\hat{L}_{12}^{-1}$, $\hat{L}_{21}^{-1}$, $\mathcal{T}(\hat{L}, \Delta)$ *and* $\mathcal{T}(L, G)$ *are well defined, causal and* L_2*-stable, then (1)* $\mathcal{T}(\hat{L}, \Delta)$ *is non-expansive if and only if* $\Delta \in$ sector[S] *and (2)*

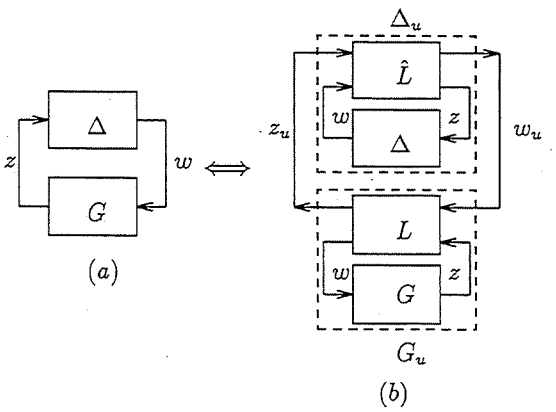


Figure 5.3 General sector normalization.

system of Fig. 5.3(a) is L_2*-stable if and only if system Fig. 5.3(b) is* L_2*-stable.*

Proof: The proof is constructive and consists of two main steps: sector transformation for Δ to find $\hat{L}$, loop transformation to find L.

Now we are trying to find an LFT $\mathcal{T}(\hat{L}, \Delta)$, such that $\Delta \in$ sector[S] iff $\Delta_u = \mathcal{T}(\hat{L}, \Delta)$ is non-expansive or $\gamma_2(\Delta_u) \leq 1$. Let $w_u = \Delta_u z_u$, then

$$\gamma_2(\Delta_u) \leq 1 \Leftrightarrow -\|z_u\|_2^2 + \|w_u\|_2^2 \leq 0.$$

Since $\mathbf{L}_2$ is also a Hilbert space under the inner product of (5.1), thus

$$\langle z_u, w_u \rangle = \langle w_u, z_u \rangle$$

exist, then,

$$-\|z_u\|_2^2 + \|w\|_2^2 = \langle -z_u - w_u, z_u - w_u \rangle,$$

and we have

$$\langle -z_u - w_u, z_u - w_u \rangle \leq 0,$$

that is

$$\Delta_u \in \text{sector}[S_u], \text{ with } S_u = \begin{bmatrix} -I & -I \\ I & -I \end{bmatrix}.$$

The problem now is to find an LFT in Fig. 5.3(b) (in the top) such that

$$\langle -z_u - w_u, z_u - w_u \rangle \leq 0 \Leftrightarrow \langle S_{11}z + S_{12}w, S_{21}z + S_{22}w \rangle \leq 0. \quad (5.6)$$

The condition of (5.6) can be satisfied trivially by letting

$$S_u \begin{bmatrix} z_u \\ w_u \end{bmatrix} = S \begin{bmatrix} z \\ w \end{bmatrix}. \quad (5.7)$$

Now we can find $\hat{L}$ by solving (5.7) as follows. Rearrange (5.7):

$$\begin{bmatrix} I & S_{11} \\ I & S_{21} \end{bmatrix} \begin{bmatrix} w_u \\ z \end{bmatrix} = \begin{bmatrix} -I & -S_{12} \\ I & -S_{22} \end{bmatrix} \begin{bmatrix} z_u \\ w \end{bmatrix}. \quad (5.8)$$

Left multiplying each side of (5.8) with $\begin{bmatrix} I & S_{11} \\ I & S_{21} \end{bmatrix}^{-1}$ (this is well defined from the assumption of the theorem), we have:

$$\begin{bmatrix} w_u \\ z \end{bmatrix} = \begin{bmatrix} I & S_{11} \\ I & S_{21} \end{bmatrix}^{-1} \begin{bmatrix} -I & -S_{12} \\ I & -S_{22} \end{bmatrix} \begin{bmatrix} z_u \\ w \end{bmatrix} = \hat{L} \begin{bmatrix} z_u \\ w \end{bmatrix},$$

where $\hat{L}$ is in (5.4).

The second step is to prove that the system (a) is L_2-stable if and only if the system (b) is L_2-stable. We use the loop transformation techniques here. It is easy to check that the LFT $\mathcal{T}(\hat{L}, \Delta)$ can be formed via the transformation shown in Fig. 5.4. The same transformation now is applied to Fig. 5.3(a). First, a feedback of $\hat{L}_{22}$ and a feedforward of $-\hat{L}_{22}$ are added around Δ and G respectively, see Fig. 5.5(b), based on Lemma 1. Then insert multiplier $\hat{L}_{21}$ and $\hat{L}_{12}$ on the left and right sides of Δ, and the multiplier $\hat{L}_{21}^{-1}$ and $\hat{L}_{12}^{-1}$ on the left and right sides of G, see Fig. 5.5(c), based on Lemma 2. And finally, add a feedforward of L_{11} and a feedback of $-L_{11}$ as shown in Fig. 5.5(d), again based on Lemma 1.

Note that the assumption of well-posedness, causality and stability of the systems in (5.3) and $\hat{L}_{12}^{-1}$, $\hat{L}_{21}^{-1}$, Δ_u and G_u are necessary for all of the

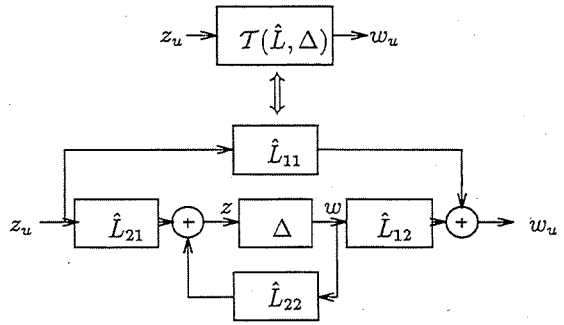


Figure 5.4 Sector transformation for Δ.

transformations to be effective. From Fig. 5.5(d) (bottom dashed box), for the transformed G_u, we have:

$$z_u = \hat{L}_{21}^{-1} z - \hat{L}_{21}^{-1} \hat{L}_{22} w,$$

$$w = -\hat{L}_{12}^{-1} L_{11} z_u + \hat{L}_{12}^{-1} w_u,$$

which can be rearranged as:

$$\begin{bmatrix} \hat{L}_{11} & \hat{L}_{12} \\ \hat{L}_{21} & \hat{L}_{22} \end{bmatrix} \begin{bmatrix} z_u \\ w \end{bmatrix} = \begin{bmatrix} w_u \\ z \end{bmatrix}.$$

That is

$$\hat{L} \begin{bmatrix} z_u \\ w \end{bmatrix} = \begin{bmatrix} w_u \\ z \end{bmatrix},$$

and

$$\begin{bmatrix} z_u \\ w \end{bmatrix} = \hat{L}^{-1} \begin{bmatrix} w_u \\ z \end{bmatrix} = L \begin{bmatrix} w_u \\ z \end{bmatrix},$$

where L is given in (5.5). □

Remarks: 1) For uncertainty modeling, the key issue here is that we have to find an LFT, $\mathcal{T}(L, G)$, such that the corresponding transformed

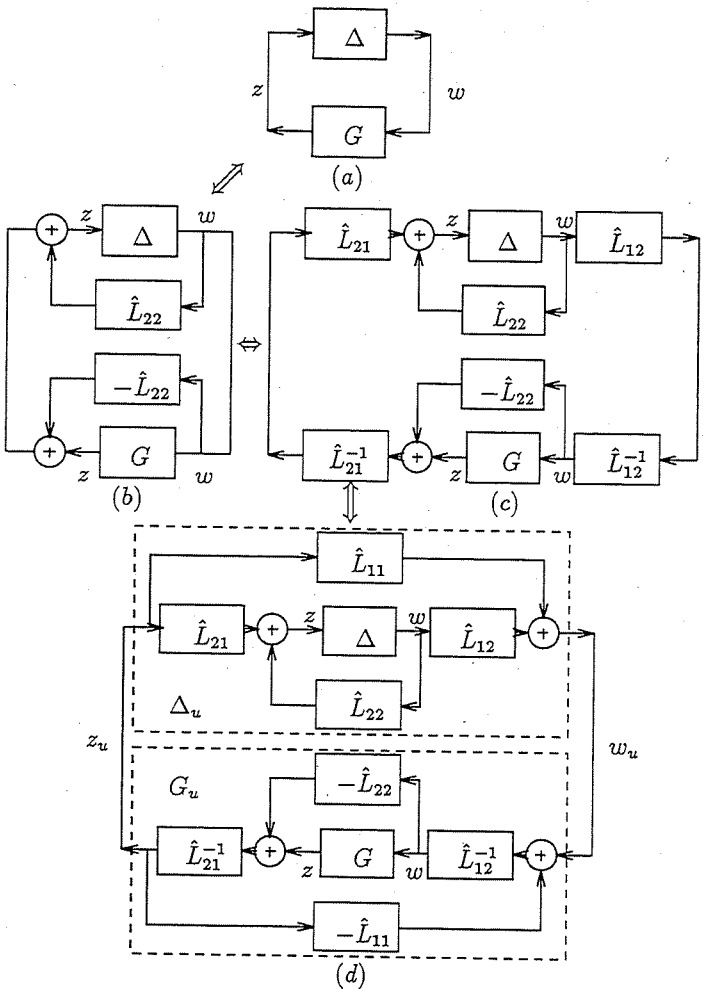


Figure 5.5 Loop transformations.

uncertainty $\Delta_u = \mathcal{T}(\hat{L}, \Delta)$ is non-expansive if and only if Δ is in the sector$[S]$ and that the original system with Δ and G is stable if and only if the transformed system with Δ_u and G_u is stable. In doing so, it is ensured that no conservativeness whatsoever is introduced in the uncertainty modeling step. Note that $\hat{L}$ will not be used in further analysis, since we only need to check the small gain condition for $\mathcal{T}(L, G)$. 2) The verification of well-posedness and stability of $\mathcal{T}(L, G)$, which are the conditions used in the theorem, can be done immediately after L is found for robustness analysis. For control synthesis, however, this verification can only be done after a control is found such that the closed loop system is available. But any control design should always guarantee that the closed loop system ($\mathcal{T}(L, G)$) is well-defined and stable, so in this sense, we do not need to make this verification in the uncertainty modeling step. 3) From the small gain theorem, it is easy to see that under the condition of the theorem the system of Fig. 5.3(b) is stable if $\mathcal{T}(L, G)$ is strict non-expansive or $\gamma(\mathcal{T}(L, G)) < 1$). 4) The existence condition of systems in (5.3) can be often satisfied in practice. Since if $\Delta \in$ sector$[S]$, then $\Delta \in$ sector$[S_d]$ with

$$S_d = \begin{bmatrix} S_{11} & S_{12} \\ dS_{21} & dS_{22} \end{bmatrix}, 0 < d < \infty. \tag{5.9}$$

(this can be directly verified from the sector definition), then by choosing suitable d, the existence condition of systems in (5.3) can be satisfied.

5.3 Normalization of some special sectors

Theorem 11 gives a general way to find the transformed Δ_u and G_u via two LFTs such that Δ is in a general sector $[S]$ if and only if Δ_u is non-expansive. However the formulae for the LFTs might be unnecessary complicated when some special sectors, instead of a general sector $[S]$, are used. Here we will show the normalization for two special sectors: general conic sector and passive sector.

5.3.1 General conic sector

First, we give the definition of a general conic sector.

Definition 14 *Given three linear, causal, stable transfer functions C,P and Q with $P = P^H$ and $Q = Q^H$ and P^{-1} and Q^{-1} exist, and an L_2-stable Δ. Then Δ is said to be in an L_2 conic sector, denoted by $Cone(C, P, Q)$, or*

$$\Delta \in \text{Cone}(C, P, Q),$$

if

$$\|Q^{-1/2}(w - Cz)\|_2 \leq \|P^{1/2}z\|_2, w = \Delta z, \forall z \in \mathbf{L}_2,$$

which is equivalent to

$$\gamma_2\left(Q^{-1/2}(\Delta - C)P^{-1/2}\right) \leq 1. \tag{5.10}$$

Now consider the system of Fig. 5.6(a), where $\Delta \in \text{Cone}(C, P, Q)$. We are trying to find an LFT with a coefficient matrix L for G. Note that (5.10) gives an explicit sector transformation for Δ that is

$$\Delta_u = Q^{-1/2}(\Delta - C)P^{-1/2},$$

which is non-expansive. So we do not need to find $\hat{L}$ in order to find L. By loop shift and multiplier techniques, we can easily find a transformed version of the original system Fig. 5.6(a), shown in Fig. 5.6(b). From Fig. 5.6(b), we have

$$z_u = Q^{1/2}z,$$

$$w = P^{1/2}w_u + Cz,$$

and from these two equations,

$$\begin{bmatrix} z_u \\ w \end{bmatrix} = L \begin{bmatrix} w \\ z \end{bmatrix}, \; L = \begin{bmatrix} 0 & Q^{1/2} \\ P^{1/2} & C \end{bmatrix}. \tag{5.11}$$

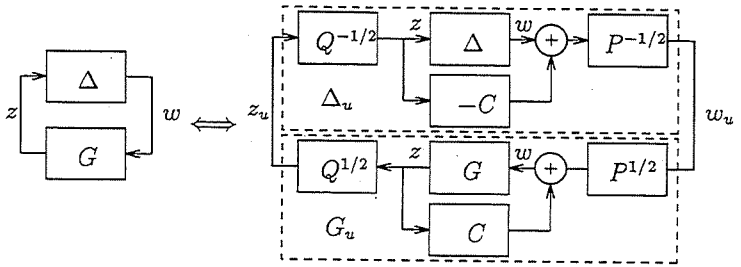


Figure 5.6 General conic sector normalization.

5.3.2 Passive sector

A system $\Delta : \mathbf{L}_{2e} \to \mathbf{L}_{2e}$ is passive if

$$\langle w, z \rangle > 0, \forall w = \Delta z, z \in \mathbf{L}_{2e}.$$

Now suppose $(I + \Delta)^{-1} : \mathbf{L}_{2e} \to \mathbf{L}_{2e}$ exists [1], then Δ is passive if and only if $(\Delta - I)(\Delta + I)^{-1}$ is non-expansive or

$$\gamma_2\left((\Delta - I)(\Delta + I)^{-1}\right) \leq 1. \tag{5.12}$$

We refer to [23] for details. Now consider the system of Fig. 5.7(a), assume that Δ is passive, then from (5.12), we can obviously let

$$\Delta_u = (\Delta - I)(\Delta + I)^{-1}. \tag{5.13}$$

The transformation of (5.13) to Δ is shown in Fig. 5.7(b) in the dashed box denoted by Δ_u. Again, by the loop shift and multiplier techniques, we can find the transformation for G shown in the dashed box denoted by G_u and we can write down the following relations:

$$z_u = w_u + \sqrt{2}z,$$

$$w = \sqrt{2}w_u + z.$$

[1] If Δ is incrementally strictly passive, then $(I + \Delta)^{-1}$ exists [23]

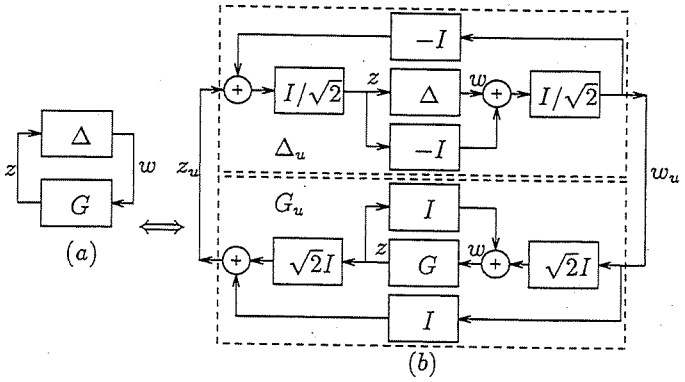


Figure 5.7 Passive sector normalization.

By these two equations, we have

$$\begin{bmatrix} z_u \\ w \end{bmatrix} = L \begin{bmatrix} w_u \\ z \end{bmatrix}, \; L = \begin{bmatrix} I & \sqrt{2} \\ \sqrt{2} & I \end{bmatrix}. \tag{5.14}$$

5.4 Normalization of structured uncertainties

For structured uncertainties, $\Delta \in \mathbf{\Delta}$, from the definitions of the uncertainty structure $\mathbf{\Delta}$, Δ has a diagonal block structure:

$$\Delta = \begin{bmatrix} \Delta_1 & & \\ & \ddots & \\ & & \Delta_q \end{bmatrix}.$$

Normally in practice, the sector conditions are given for all individual uncertainty blocks Δ_i (instead of Δ): $\Delta_i \in \text{sector}[S_i]$. In this case, we can do the sector transformation with the following steps:

1. Find a sector transformation LFT $\Delta_{u_i} = \mathcal{T}(\hat{L}_i, \Delta_i)$, such that $\Delta_i \in \text{sector}[S_i]$ if and only if Δ_{u_i} is non-expansive or $\gamma_2(\Delta_{u_i}) \leq 1$ by (5.4) in Theorem 11 with $\hat{L}$ and S replaced by $\hat{L}_i$ and S_i.

2. Find an LFT coefficient L_i by (5.5) in Theorem 11 with L and $\hat{L}$ replaced by L_i and $\hat{L}_i$.

3. Form L by lumping all L_i.

We summarize this idea in the following corollary for theorem 11

Corollary 1 *Consider the system of Fig. 5.8(a), where $\Delta \in \mathbf{\Delta}$ with each diagonal block $\Delta_i \in \text{sector}[S_i]$, and its transformed version of Fig. 5.8(b), where $\hat{L}_i$ are formed via (5.4), and L_i are formed via (5.5). Define an LFT coefficient L via*

$$L \triangleq \left[\begin{array}{ccc|ccc} [L_1]_{11} & & & [L_1]_{12} & & \\ & \ddots & & & \ddots & \\ & & [L_q]_{11} & & & [L_q]_{12} \\ \hline [L_1]_{21} & & & [L_1]_{22} & & \\ & \ddots & & & \ddots & \\ & & [L_q]_{21} & & & [L_q]_{22} \end{array}\right]. \tag{5.15}$$

Suppose that $[\hat{L}_i]_{lk}$, $\forall i = 1, \cdots, q$, $l, k = 1, 2$, $[\hat{L}_i]_{lk}^{-1}$, $\forall i = 1, \cdots, q$, $l, k = 1, 2$, $l \neq k$, $\mathcal{T}(\hat{L}_i, \Delta_i)$, $\forall i = 1, \cdots, q$ and $\mathcal{T}(L, Q)$ are well-defined, causal and L_2-stable, then (1) $\forall i = 1, \cdots, q$, $\Delta_i \in \text{sector}[S_i]$ if and only if $\mathcal{T}(\hat{L}_i, \Delta_i)$, $\forall i = 1, \cdots, q$ are non-expansive and (2) the system (a) is L_2-stable if and only if system (b) is L_2-stable.

Prove: The proof can be done along the same line as the proof for Theorem 11, the difference is that sector transformations and loop transformations are made for multiloop now, while in the case of theorem 11 only single loops are treated. L in (5.15) is actually formed by lumping all L_i together and then rearranging the inputs and outputs, see the dotted box in Fig. 5.8(b).

Remarks: The LFT coefficient L is not necessary to be formed explicitly. The sub-LFT coefficients $L_i, i = 1, \cdots, q$, can be actually absorbed into G to form $G_u = \mathcal{T}(L, G)$ individually, we will discuss a more practical method in Chapter 7.

5.5 Examples

In this section we take an example to show how the theory above can be used.

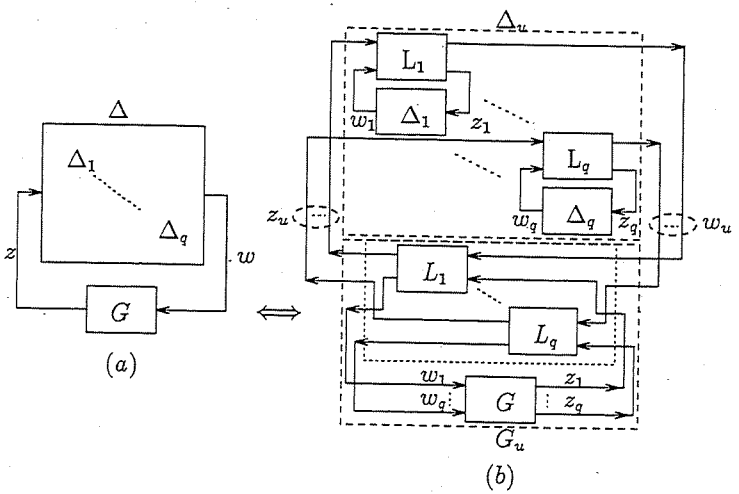


Figure 5.8 Normalization for structured uncertainties.

Example 7 *Consider the system of Fig. 5.3(a) with* $w = \Delta z$ *and* $\Delta \in$ sector$[S]$ *where*

$$S = \begin{bmatrix} a & -I \\ I & -b^{-1} \end{bmatrix}, \ b > a > 0,$$

or equivalently:

$$\langle az - w, z - b^{-1}w \rangle \leq 0, \ b > a > 0.$$

We are trying to find an LFT coefficient L *such that there is an* $\hat{L}$ *and* $\Delta \in$ sector$[S]$ *if and only if* $\mathcal{T}(\hat{L}, \Delta)$ *is non-expansive, and further, Fig. 5.3(a) is* L_2*-stable if and only if Fig. 5.3(b) is* L_2*-stable. As mentioned already, sector*$[S]$ *is equivalent to sector*$[S_d]$ *defined in (5.9), we use* S_d *in what follows.*

Now by the formulae in Theorem 11, we have

$$S_d = \begin{bmatrix} aI & -I \\ dI & -db^{-1}I \end{bmatrix},$$

$$\begin{aligned} \hat{L} &= \begin{bmatrix} I & aI \\ I & dI \end{bmatrix}^{-1} \begin{bmatrix} -I & I \\ I & db^{-1} \end{bmatrix} \\ &= \frac{1}{d-a} \begin{bmatrix} dI & -aI \\ -I & I \end{bmatrix} \begin{bmatrix} -I & I \\ I & db^{-1} \end{bmatrix} \\ &= \frac{1}{d-a} \begin{bmatrix} (-d-a)I & (d-adb^{-1})I \\ 2 & (-1+db^{-1}) \end{bmatrix}, \qquad (5.16) \\ L &= \frac{1}{1+db^{-1}} \begin{bmatrix} (1-db^{-1})I & (d-adb^{-1})I \\ 2I & (a+d)I \end{bmatrix}. \qquad (5.17) \end{aligned}$$

To ensure that the transformation is effective, we need that $\mathcal{T}(\hat{L}, \Delta)$ *is well-defined causal and* L_2*-stable (one obvious condition is that* $d \neq a$*), that is:*

$$-\frac{d+a}{d-a}I + \frac{d(1-ab^{-1})}{d-a}I\Delta(I - \frac{db^{-1}-1}{d-a}\Delta)^{-1}\frac{2}{d-a}I$$

is well-defined, causal and stable. Thus, obviously, $\mathcal{T}(\hat{L}, \Delta)$ *is well-defined, causal and stable if and only if* $(I - \frac{db^{-1}-1}{d-a}\Delta)^{-1}$ *is. If* Δ *is incrementally strictly passive and has finite incremental gain, then it can be guaranteed that* $(I - \frac{-1+b^{-1}}{1-a}\Delta)^{-1}$ *is well-defined, causal and stable.*

Now let $b \to \infty$*, that is* Δ *is passive with index a [70], or* $\Delta - aI$ *is passive, then (5.16) and (5.17)*

$$\hat{L} = \begin{bmatrix} -\frac{d+a}{d-a}I & \frac{d}{d-a}I \\ \frac{2}{d-a}I & \frac{-1}{d-a}I \end{bmatrix},$$

$$L = \begin{bmatrix} I & dI \\ 2I & (d+a)I \end{bmatrix}.$$

If further let $a \to 0$*, that is now* Δ *is passive, then*

$$\hat{L} = \begin{bmatrix} -I & I \\ \frac{2}{d}I & \frac{-1}{d}I \end{bmatrix},$$

$$L = \begin{bmatrix} I & dI \\ 2I & dI \end{bmatrix},$$

and by setting $d = 1$, this recovers L in (5.14) (the LFTs of both L's are the same though there are differences in parameters).

Another way to find L is to use the conic sector transformation. In this case we need the assumption that $b \neq \infty$. Since

$$\langle w - az, b^{-1}w - z\rangle \leq\Leftrightarrow \langle w - az, w - bz\rangle \leq 0,$$

and

$$\langle w - az, w - bz\rangle = \|w - cz\| - r^2\|z\|_2,$$

with

$$c = (a + b)/2,\ r = (b - a)/2.$$

That is

$$\Delta \in \text{Cone}((a + b)/2I, r^2I, I).$$

Now using the formula for L in (5.11), we have

$$L = \begin{bmatrix} 0 & I \\ rI & cI \end{bmatrix}. \tag{5.18}$$

This is equivalent to L given by (5.17) if choosing $d = b$.

5.6 Conclusions

In this chapter, we introduced a method for normalizing structured uncertainties in general sectors. The significance of the results is that we can use a systematic way to normalizing structured uncertainties, which in practice are often not non-expansive or even not norm bounded but belong to some general sectors, and that it is easy to code a computer program to implement the normalization method based on all the formulae in the chapter.

Chapter 6

D-K iteration with mixed D-scalings

Control synthesis for unstructured uncertainty systems to ensure robust stability is successfully tackled by the H_∞ control synthesis method, which turns out to be a problem of just solving two Riccati equations [28], or (two) Riccati inequality problem(s) [37]. However robust control synthesis for structured uncertainty systems is much more complicated than unstructured uncertainty systems. Of course one could use the same H_∞ technique to solve the robust control synthesis problem for structured uncertainty systems by taking the structured uncertainty block as an unstructured uncertainty block, but this would lead to arbitrarily large conservativeness. One basic way to reduce conservativeness is to introduce scalings. The theoretical background for introducing scalings consists of the small gain theorem and multiplier techniques as discussed in Chapter 5. Basically, scalings provide an elegant and efficient way of handling directional and frequency contents of structured uncertainties. This is the main idea of μ synthesis where scalings are introduced to treat robust stability and performance problems in the presence of LTI structured uncertainties.

The basic idea of D-K iteration is to find an optimal controller (K) and an optimal scaling (D) in an iterative way, such that the H_∞ norm of the scaled closed loop system is minimized. The standard D-K iteration [4] with frequency dependent D scaling can treat robust performance in the presence of linear time invariant (LTI) uncertainties, but for non-

linear time varying (NLTV) uncertainties no general conclusion can be reached. Recently, a D-K iteration procedure with constant D-scalings [1] was introduced, which can treat robust performance in the presence of any L_2 norm bounded NLTV uncertainty. In practice, however, LTI and NLTV uncertainties often exist simultaneously, in which case frequency dependent D-scalings can not ensure robust stability, constant D-scalings, on the other hand, can be conservative.

In this chapter, we shall mainly present a procedure of D-K iteration with mixed frequency and constant D-scalings for control synthesis of structured uncertainty systems with both LTI and NLTV uncertainty blocks. In section 6.1, robustness analysis with different D-scalings to different uncertainty blocks will be discussed. Algorithms to find D-scalings will be presented in section 6.2. Algorithms of D-K iterations for different D-scalings will be presented in section 6.3. Section 6.4 gives a practical example to show how the algorithm of D-K iteration with mixed D-scalings works. These results will be used in the next chapter. Some concluding remarks will be given in section 6.5

6.1 D-scaling for structured uncertainty

In this section we develop a framework on robustness analysis with D-scalings and classify different uncertainty blocks according to D-scaling properties.

6.1.1 LTI and NLTV uncertainty structures

For both stability and performance robustness analysis, one can use the feedback system as shown in Fig. 6.1, where $G_c(s)$ is a transfer function for a stable (or stabilized by feedback control) LTI system. Δ is an operator and $\Delta \in \mathbf{\Delta}$, where $\mathbf{\Delta}$ is the structure set defined as:

$$\mathbf{\Delta} \triangleq \{\mathrm{diag}(\Delta_l, \Delta_n) : \Delta_l \in \mathbf{\Delta}_l,\ \Delta_n \in \mathbf{\Delta}_n\}, \tag{6.1}$$

$$\begin{aligned}\mathbf{\Delta}_l \triangleq\ & \{\mathrm{diag}(\delta_1 I_{r_1^l}, \cdots, \delta_{S^l} I_{r_{S^l}^l}, \Delta_{S^l+1}, \cdots, \Delta_{S^l+F^l}) : \\ & \delta_i \in \mathcal{S}_2^1,\ \Delta_{S^l+j} \in \mathcal{S}_2^{m_j^l}\},\end{aligned} \tag{6.2}$$

$$\mathbf{\Delta}_n \triangleq \{\mathrm{diag}(\delta_1 I_{r_1^n}, \cdots, \delta_{S^n} I_{r_{S^n}^n}, \Delta_{S^n+1}, \cdots, \Delta_{S^n+F^n}) :$$

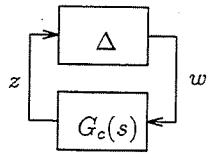


Figure 6.1 Feedback system for robustness analysis.

$$\delta_i : \mathcal{B}_N(\mathbf{L}_2^1),\ \Delta_{S^n+j} : \mathcal{B}_N(\mathbf{L}_2^{m_j^n})\}. \tag{6.3}$$

where $\mathbf{\Delta}_l$ and $\mathbf{\Delta}_n$ are LTI and NLTV uncertainty structures respectively, and S^l, S^n, F^l and F^n are the numbers of repeated scalar and full blocks, r_i^l, r_i^n, m_j^l and m_j^n are dimensions for the corresponding blocks. Both sets $\mathbf{\Delta}_l$ and $\mathbf{\Delta}_n$ are causal, stable and non-expansive, that is

$$\gamma_2(\Delta_l) = \|\Delta_l\|_\infty \le 1,\ \ \gamma_2(\Delta_n) \le 1.$$

Note that compared to the definition of $\mathbf{\Delta}$ in (1.8), the parametric uncertainty block $\mathbf{\Delta}_r$ has been merged into LTI block $\mathbf{\Delta}_l$. Corresponding to the above uncertainty structure set, define a scaling matrix set $\mathbf{D}$:

$$\mathbf{D} \triangleq \{\mathrm{diag}(D_f, D_c) : D_f \in \mathbf{D}_f,\ D_c \in \mathbf{D}_c\}, \tag{6.4}$$

$$\mathbf{D}_f \triangleq \{\mathrm{diag}(D_1, \cdots, D_{S^l}, d_{S^l+1} I_{m_1^l}, \cdots d_{S^l+F^l} I_{m_{F^l}^l}) :$$
$$D_i \in \mathbb{C}^{r_i^l \times r_i^l}, D_i = D_i^* > 0, d_{S^l+j} \in \mathbb{R}, d_{S^l+j} > 0\}, \tag{6.5}$$

$$\mathbf{D}_c \triangleq \{\mathrm{diag}(D_1, \cdots, D_{S^n}, d_{S^n+1} I_{m_1^n}, \cdots d_{S^n+F^n} I_{m_{F^l}^n}) :$$
$$D_i \in \mathbb{C}^{r_i^n \times r_i^n}, D_i = D_i^* > 0, d_{S^n+j} \in \mathbb{R}, d_{S^n+j} > 0\}. \tag{6.6}$$

We separate the uncertainty structure into two sub-structures. In doing so, we can handle LTI and NLTV uncertainties later on. Suppose that $D(s)$ is a transfer function of a stable LTI system and $D(s)^{-1}$ is also stable (that is $D(s)$ is outer). From the multiplier techniques (Lemma 2) used already in the last chapter for uncertainty modeling, the feedback system of Fig. 6.1 is stable if the transformed system of Fig. 6.2 is stable. Now suppose further that $D(s)^{-1}\Delta D(s)$ is non expansive, then from the

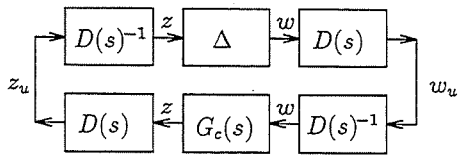


Figure 6.2 Robustness analysis with D-scalings.

small gain theorem, the system of Fig. 6.1 is stable if $D(s)G_c(s)D(s)^{-1}$ is strictly non expansive, or

$$\|D(s)G_c(s)D(s)^{-1}\|_\infty < 1.$$

The problem is now to find a class of transfer functions $D(s)$ such that $D(s)^{-1}\Delta D(s)$ is non expansive if and only if Δ is non-expansive,

6.1.2 Frequency dependent D-scaling for LTI uncertainties

Suppose all $\Delta \in \mathbf{\Delta} = \mathbf{\Delta}_l$ are LTI uncertainties and $D(s) \in \mathbf{D}$[1], then $D(s)$ commutes with Δ:

$$D(s)\Delta = \Delta D(s).$$

Then obviously

$$\|\Delta\|_\infty \le 1 \ \Leftrightarrow \|D(s)^{-1}\Delta D(s)\|_\infty \le 1. \tag{6.7}$$

Thus the stability condition for the system Fig. 6.1 becomes

$$\inf_{D(s)\in\mathbf{D}} \|D(s)G_c(s)D(s)^{-1}\|_\infty < 1,$$

or equivalently

$$\sup_\omega \inf_{D(j\omega)\in\mathbf{D}} \bar{\sigma}(D(j\omega)G_c(j\omega)D(j\omega)^{-1}) < 1. \tag{6.8}$$

[1]We say $D(s) \in \mathbf{D}$ if for any specific $s \in \mathbb{C}$, $D(s) \in \mathbf{D}$, but the condition $D(s)^H = D(s)$ is not needed.

Condition (6.8) gives a sufficient condition of robust stability for all LTI $\Delta \in \mathbf{\Delta}$. In the μ framework, condition (6.8) is the so called the frequency domain upper bound (FDUB) test [29], since for any given frequency ω,

$$\inf_{D(j\omega)\in\mathbf{D}_f} \bar{\sigma}(D(j\omega)G_c(j\omega)D(j\omega)^{-1}) \tag{6.9}$$

is a upper bound of $\mu(G_c(j\omega))$.

The set of uncertainties satisfying (6.7) with frequency dependent D-scalings includes not only LTI uncertainties but some NLTV set as well. In [72], Safonov showed that if Δ is a memoryless time-invariant nonlinearity and is incrementally non-expansive,

$$\frac{\|\Delta(z_1) - \Delta(z_2)\|_2}{\|z_1 - z_2\|_2} \le 1,$$

then

$$\gamma_2(\Delta) \le 1 \;\Leftrightarrow\; \gamma_2(D(s)^{-1}\Delta D(s)) \le 1,$$

where $D(s)$ has a structure compatible to the structure of Δ. So the frequency upper bound test is still valid for memoryless and time invariant nonlinear uncertainties which are incrementally non-expansive. We refer to [72] for details on this type of nonlinearities.

6.1.3 Constant D-scaling for NLTV uncertainties

Now suppose that all $\Delta \in \mathbf{\Delta} = \mathbf{\Delta}_n$ are NLTV, and $D(s) \in \mathbf{D}$. Because of the nonlinearity of Δ, Δ and $D(s)$ do not commute any more. Thus in general no conclusion can be reached on whether or not $D(s)^{-1}\Delta D(s)$ is non-expansive except in the case that $D(s)$ is a constant matrix. In this case, let $D(s) = D$, $D \in \mathbf{D}$. Then D and Δ commute: $D\Delta = \Delta D$, and

$$\gamma_2(\Delta) < 1 \;\Leftrightarrow\; \gamma_2(D^{-1}\Delta D) < 1.$$

Thus the stability condition for the system Fig. 6.1 becomes

$$\inf_{D\in\mathbf{D}} \|DG_c(s)D^{-1}\|_\infty < 1,$$

or equivalently

$$\inf_{D\in\mathbf{D}} \sup_{\omega} \bar{\sigma}(DG_c(j\omega)D^{-1}) < 1. \tag{6.10}$$

Condition of (6.10) is also called frequency domain constant D (FDCD) test. In the μ framework, this is the state space upper bound of μ for a discrete system [29]. Now it is easy to verify that:

$$\inf_{D\in\mathbf{D}_c} \sup_{\omega} \bar{\sigma}(DG_c(j\omega)D^{-1}) < 1$$
$$\Downarrow$$
$$\sup_{\omega} \inf_{D(j\omega)\in\mathbf{D}_f} \bar{\sigma}(D(j\omega)G_c(j\omega)D(j\omega)^{-1}) < 1.$$

This relation shows that if Δ is LTI or memoryless and incrementally non-expansive, then a constant D-scaling will give a conservative result.

6.1.4 Mixed D-scaling for LTI and NLTV uncertainties

In practice, however, LTI and NLTV uncertainties can exist simultaneously in systems. Thus in general constant D-scalings can be conservative, while frequency dependent D-scalings, on the other hand, can not guarantee robustness of stability. The main idea for mixed D-scalings is using frequency dependent D-scalings for LTI (or memoryless and incrementally non expansive) blocks, and constant D-scalings for NLTV blocks.

Now suppose $\Delta \in \mathbf{\Delta}$:

$$\Delta = \begin{bmatrix} \Delta_l & \\ & \Delta_n \end{bmatrix},\ \Delta_l \in \mathbf{\Delta}_l,\ \Delta_n \in \mathbf{\Delta}_n,$$

where the sets $\mathbf{\Delta}_l$ and $\mathbf{\Delta}_n$ contains LTI blocks and NLTV blocks respectively. Let

$$D(s) = \begin{bmatrix} D_f(s) & \\ & D_c \end{bmatrix}, D_f(s) \in \mathbf{D}_f, D_c \in \mathbf{D}_c.$$

Corresponding to Δ, $D(s)$ now has also two separated structures, one is frequency dependent and the other is just constant. Since $D_f(s)$ commutes with Δ_l and D_c commutes with Δ_n, then $D(s)$ commutes with Δ, and thus $D(s)^{-1}\Delta D(s)$ is non expansive if and only if Δ is non expansive. The robust stability condition now becomes:

$$\inf_{D(s)\in\mathbf{D}} \|D(s)G_c(s)D(s)^{-1}\|_\infty < 1,$$

or equivalently

$$\inf_{D_c\in\mathbf{D}_c}\sup_{\omega}\inf_{D_f(j\omega)\in\mathbf{D}_f}\bar{\sigma}\left(\begin{bmatrix} D_f(j\omega) & \\ & D_c\end{bmatrix}G_c(j\omega)\begin{bmatrix} D_f(j\omega) & \\ & D_c\end{bmatrix}^{-1}\right)<1. \quad (6.11)$$

One obvious result is:

$$\inf_{D\in\mathbf{D}_c}\sup_{\omega}\bar{\sigma}(DG_c(j\omega)D^{-1})<1$$
$$\Downarrow$$
$$\inf_{D_c\in\mathbf{D}_c}\sup_{\omega}\inf_{D_f(j\omega)\in\mathbf{D}_f}\bar{\sigma}\left(\begin{bmatrix} D_f(j\omega) & \\ & D_c\end{bmatrix}G_c(j\omega)\begin{bmatrix} D_f(j\omega) & \\ & D_c\end{bmatrix}^{-1}\right)<1$$
$$\Downarrow$$
$$\sup_{\omega}\inf_{D_f(j\omega)\in\mathbf{D}_f}\bar{\sigma}(D(j\omega)G_c(j\omega)D(j\omega)^{-1})<1.$$

So condition (6.11) is weaker than condition (6.10) and meanwhile it ensures robust stability.

6.2 Calculation of D-scalings

Now we discuss the calculation of D-scalings. As mentioned above, there are three types of D-scalings: frequency dependent D-scalings, constant D-scalings and mixed D-scalings.

6.2.1 Frequency dependent D-scalings

Calculation of frequency dependent D-scalings is based on (6.9). First a complex scaling matrix is found at each frequency by solving (6.9), which is an LMI optimization problem as already mentioned, then a frequency sweep is done to find a set of scaling matrices along the frequency ω. Numerically, of course, we can only find the scaling matrices at limited frequencies in a limited sweep range. The number of frequency points and the sweep range are determined by trial-and-error, nevertheless the

bandwidth of the closed loop system $G_c(j\omega)$ is certainly a key factor to be taken into in consideration.

Now we have only a set of frequency dependent D-scaling matrices, while for control synthesis with D-K iteration, a scaling transfer function matrix is needed. Let D_ω denote a scaling matrix at a certain frequency ω. Now the problem is to find an outer transfer matrix $D(s)$ (both $D(s)$ and $D(s)^{-1}$ are stable), such that at a given frequency ω, $D(j\omega)=U_\omega D_\omega$, where U_ω is a unitary matrix: $U_\omega^H U_\omega = I$. That is $D(j\omega)$ has a polar decomposition[2] with the positive definite matrix part equal to the D-scaling matrix calculated from μ upper bound at the same frequency. The reasons why we let $D(j\omega)=U_\omega D_\omega$ are 1) D_ω, which is positive definite, is not corresponding to the frequency response of an outer transfer matrix, 2) if D_ω reaches the upper bound of $\mu(G_c(j\omega))$ at frequency ω, then $U_\omega D_\omega$ reaches either, since

$$\bar{\sigma}(D_\omega G_c(j\omega)D_\omega^{-1})=\bar{\sigma}(U_\omega D_\omega G_c(j\omega)D_\omega^{-1}U_\omega^H).$$

Note that D_ω is block diagonal, therefore we can find each block transfer matrix independently.

For a full uncertainty block, say, Δ_i, a corresponding D-scaling is a repeated scalar matrix, see (6.5). The scalar $d_{i\omega}$ at a certain frequency ω from the μ upper bound calculation is positive and taken as magnitude data. We need to find a transfer function $d_i(s)$ which is stable and minimum phase (corresponding to outer transfer matrix for MIMO systems). Using the complex cepstrum algorithm [60], the phase, associated with a stable, minimum phase, real, rational transfer with the same magnitude as the corresponding magnitude data, can be generated. With these magnitude and phase data, a real, rational transfer function is found, which fits the data in a least squares sense [4, 52].

For a repeated scalar uncertainty block, say Δ_i, a corresponding D-scaling is a full matrix, see (6.5). To form such a scaling transfer matrix, however, is still an open question. The frequency domain identification techniques[45], which emerged very recently, might be helpful for solving this problem. One could also , of course, solve this problem by taking

[2] Suppose that a matrix $A\in\mathbb{C}^{n\times n}$ has a singular value decomposition: $A=USV^T$. Let $Q=UV^T$ and $P=VSV^T$, then $A=QP$ is called a polar decomposition of A. Note that Q is a unitary matrix and P is positive (semi) definite.

the repeated scalar uncertainty block with dimension $r_i \times r_i$ as a block consisting of r_i scalar blocks. In this way D_i is just a diagonal matrix and $D_i(s)$ is a diagonal transfer matrix with each element in the diagonal being a stable and minimum phase transfer function, so that the method for constructing scalar $d_i(s)$ above can be used. But consequently this could lead to a conservative result.

6.2.2 Constant D-scalings

The problem of constant D-scalings is much easier than frequency dependent D-scalings, thanks to the well known Bounded Real Lemma [63, 77]:

Lemma 3 (Bounded Real Lemma)

Consider some (not necessarily minimal) realization $G_c(s) = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$ *of some (not necessarily square) transfer function* $G_c(s)$. *Then the following statements are equivalent:*

(i) A *is stable and* $\|D + C(sI - A)^{-1}B\|_\infty < \gamma$,
(ii) $\bar{\sigma}(D) < \gamma$ *and there exists a symmetric positive definite solution* X *to the Algebraic Riccati Inequality (ARI):*

$$A^TX + XA + \frac{C^TC}{\gamma^2} + (\frac{C^TD}{\gamma^2} + XB)(I - \frac{D^TD}{\gamma^2})^{-1}\frac{C^TD}{\gamma^2} + XB)^T < 0. \quad (6.12)$$

Now given the scaling set $\mathbf{D}$, we are considering the problem of finding a scaling $D_c \in \mathbf{D}$ such that $\|D_cG_c(s)D_c^{-1}\|_\infty$ is minimized. The following theorem characterizes the scaling matrices leading to an H_∞-norm less than a given bound γ [1]:

Theorem 12 *Given a scaling set* $\mathbf{D}$, *the following statements are equivalent:*

(i) $G(s)$ *is stable and there exists* $D_c \in \mathbf{D}$ *such that* $\|D_cG_c(s)D_c\|_\infty < \gamma$,
(ii) there exist $X > 0$ *and* $P > 0$, $P \in \mathbf{D}$ *such that:*

$$\left[\begin{array}{cc} A^TX + XA + \gamma^{-2}C^TPC & XB + \gamma^{-2}C^TPD \\ (XB + \gamma^{-2}C^TPD)^T & \gamma^{-2}D^TPD - P \end{array}\right] < 0. \quad (6.13)$$

Proof: The proof is easy. Since

$$D_cG_c(s)D_c^{-1} = \left[\begin{array}{c|c} A & BD_c^{-1} \\ \hline D_cC & D_cDD_c^{-1} \end{array}\right]$$

from the Bounded Real Lemma, (i) is equivalent to the existence of $X > 0$ and $D_c \in \mathbf{D}$ such that:

$$A^TX + XA + \frac{C^TD_c^TD_cC}{\gamma^2} + (\frac{C^TD_c^TD_cDD_c^{-1}}{\gamma^2} + XBD_c^{-1})\times$$

$$(I - \frac{D_c^{-t}D^TD_c^TD_cDD_c^{-1}}{\gamma^2})^{-1}(\frac{C^TD_c^TD_cD}{\gamma^2} + XBD_c^{-1})^T < 0.$$

Then changing variables $P = D_c^TD_c$ and eliminating D_c^{-T}, the equivalence with (ii) can be obtained by the standard Schur complement argument.

The problem of finding $X > 0$ and $P > 0$ such that (6.13) is satisfied is a generalized eigenvalue problem, which is quasi convex problem as discussed in Appendix A. So some efficient algorithms exist to solve them, see Appendix A.

6.2.3 Mixed D-scalings

The mixed D-scaling problem of (6.11) is more complicated than frequency D-scaling or constant D-scaling problems. For frequency dependent D-scalings, the problem is convex over the frequency response matrices of a scaling transfer matrix, but not over the state space realization matrices of the transfer matrix, while for constant D-scalings, the convexity is obtained in a state space framework. Thus it is difficult to combine the two scaling methods in the same framework. Here we suggest a heuristic method to tackle the problem. The basic idea is that we first find a frequency dependent scaling D_f, then find a constant scaling D_c[3]. And in each step, we have a convex optimization problem. To do that, we rewrite the left hand side of of (6.11) as the following optimization problem:

[3] We choose this sequence due to one technique reason: frequency scaling is much faster than the constant scaling.

$$\inf_{D\in\mathbf{D}} \sup_{\omega} \inf_{D_f(j\omega)\in\mathbf{D}_f} \bar{\sigma}\left(D\begin{bmatrix} D_f(j\omega) & \\ & I \end{bmatrix} G_c(j\omega) \begin{bmatrix} D_f^{-1}(j\omega) & \\ & I \end{bmatrix} D^{-1}\right). \quad (6.14)$$

Now we first solve the optimization problem:

$$\sup_{\omega} \inf_{D_f(j\omega)\in\mathbf{D}_f} \bar{\sigma}\left(\begin{bmatrix} D_f(j\omega) & \\ & I \end{bmatrix} G_c(j\omega) \begin{bmatrix} D_f(j\omega) & \\ & I \end{bmatrix}^{-1}\right). \quad (6.15)$$

Suppose that an optimal solution, say, $D_f^o(s)$, is found, then we can solve another optimization problem:

$$\inf_{D\in\mathbf{D}} \left\| D\begin{bmatrix} D_f^o(s) & \\ & I \end{bmatrix} G_c(s) \begin{bmatrix} D_f^o(s) & \\ & I \end{bmatrix}^{-1} D^{-1} \right\|_\infty. \quad (6.16)$$

The optimization problem of (6.15) is convex in the frequency domain. Actually it is a frequency dependent D-scaling problem for $G_c(s)$ with a scaling structure $\bar{\mathbf{D}}$ defined by

$$\bar{\mathbf{D}} \triangleq \left\{ \begin{bmatrix} D_f & \\ & dI \end{bmatrix} : D_f \in \mathbf{D}_f,\ 0 < d \in \mathbb{R} \right\}. \quad (6.17)$$

It is easy to see this since for any $D_f(j\omega) \in \mathbf{D}_f$ and $d(j\omega) \in \mathbb{R}, d(j\omega) > 0$, the following equation holds:

$$\begin{bmatrix} D_f(j\omega) & \\ & d(j\omega)I \end{bmatrix} G_c(j\omega) \begin{bmatrix} D_f(j\omega) & \\ & d(j\omega)I \end{bmatrix}^{-1}$$

$$= \begin{bmatrix} d(j\omega)^{-1}D_f(j\omega) & \\ & I \end{bmatrix} G_c(j\omega) \begin{bmatrix} d(j\omega)^{-1}D_f(j\omega) & \\ & I \end{bmatrix}^{-1}.$$

Thus it can be solved by use of the frequency scaling procedure in the section 6.2.1. The optimization problem of (6.16) is just a constant D-scaling problem for the scaled system:

$$\begin{bmatrix} D_f^o(s) & \\ & I \end{bmatrix} G_c(s) \begin{bmatrix} D_f^o(s)^{-1} & \\ & I \end{bmatrix}.$$

Thus a mixed D-scaling calculation is proposed as follows:

(*i*) Find an optimal $D_f(s) \in \mathbf{D}_f$, by solving the frequency dependent D-scaling problem of (6.15) for $G_c(s)$ with the modified scaling structure $\bar{\mathbf{D}}$ defined in (6.17).
(*ii*) Then fix this $D_f(s)$ and find an optimal constant D-scaling $D \in \mathbf{D}$ which solves the constant D-scaling problem of (6.16).
(*iii*) Finally $D(s)$ is set to $D\text{diag}(D_f(s), I)$.

Remarks: The difference between the standard frequency dependent (or constant) D-scaling procedure and the mixed scaling procedure proposed here is that the former can find a global minimum, while the latter finds only a local minimum since the frequency scaling D_f and the constant scaling D_c are not jointly optimal though in each step they are. However this disadvantage seems not so serious since the D-K iteration method (including the standard D-K iteration) to be discussed later on leads to a local minimum anyway. In the case of $S^n = F^n = 0$ or $S^l = F^l = 0$, then frequency dependent D-scalings or constant D-scalings will be recovered respectively.

6.3 D-K iteration

6.3.1 Control synthesis and D-K iteration

Now suppose that we have a plant represented in a linear fractional transformation as shown in Fig. 6.3(a) with a structured uncertainty $\Delta \in \mathbf{\Delta}$, control input u and measurement output y. Note that an artificial uncertainty block representing a performance requirement (in H_∞ norm) could be included into the LTI part (Δ_l) of $\mathbf{\Delta}$.

Now we are trying to find a stabilizing controller K shown in Fig. 6.3(b) such that the closed loop transfer function $G_c = \mathcal{T}(G, K)$, shown in Fig. 6.3(c), satisfies the robustness condition (6.8) for LTI (or memoryless incrementally non-expansive) uncertainties, (6.10) for NLTV uncertainties and (6.11) for general uncertainties with both LTI and NLTV blocks. So in general, we have to solve the minimization problem:

$$\inf_{\substack{K \text{ stabilizing} \\ D\in\mathbf{D}}} \|D\mathcal{T}(G,K)D^{-1}\|_\infty. \quad (6.18)$$

The minimization problem of (6.18) is actually a μ-synthesis problem [4]. However, it is not known how to solve this problem exactly since

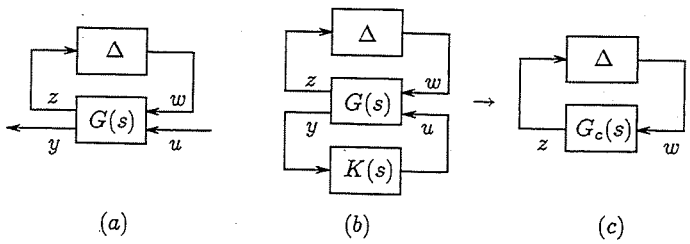


Figure 6.3 (a) plant with uncertainties, (b) control ($K(s)$) design, (c) closed loop system.

the optimization problem is not convex jointly in the control K and the scaling D. The synthesis approach proposed by Doyle in [27] as an approximation to solve the problem is an iterative scheme, referred to as D-K iteration, which involves a sequence of scaling and H_∞ control design problems. The basic idea of the D-K iteration is as follows: If we fix the scaling factor D, (6.18) is

$$\min_{K \text{ stabilizing}} \|DT(G,K)D^{-1}\|_\infty.$$

It is just a H_∞ synthesis problem, so it can be solved by H_∞ control techniques [28]. Then if we fix the controller matrix K, (6.18) is

$$\min_{D \text{ minphase,stable},D\in\mathcal{D}} \|DT(G,K)D^{-1}\|_\infty.$$

It is just a problem for calculating the μ upper bound which is a convex minimization problem for frequency dependent D-scaling or constant D-scaling as we mentioned in previous sections. Though it is not ensured that the global minimum could be reached, it has been successfully used in many applications. However, the standard D-K iteration can only be used for frequency D-scalings.

6.3.2 D-K iteration with mixed D-scalings

Now we propose a procedure for D-K iteration with mixed D-scalings. This D-K iteration scheme combines the standard D-K iteration scheme

and the mixed D-scaling procedure proposed in section 6.2.3. The algorithm now is described as follows.

Algorithm 7 (D-K iteration algorithm)

1. *Initialization:* $D_f(s)_{[0]} = I, D_{c[0]} = I$, *i=0.*
2. **Repeat**, *i=i+1.*
3. H_∞ *control design for the open loop system:*

$$G_1(s) = \begin{bmatrix} D_f(s)_{[i-1]} & 0 & 0 \\ 0 & D_{c[i-1]} & 0 \\ 0 & 0 & I \end{bmatrix} G(s) \begin{bmatrix} D_f(s)_{[i-1]} & 0 & 0 \\ 0 & D_{c[i-1]} & 0 \\ 0 & 0 & I \end{bmatrix}^{-1}.$$

A controller $K_1(s)$ is obtained and the corresponding closed loop system is $G_{c1}(s) = \mathcal{T}(G_1(s), K_1(s))$.

4. *Frequency dependent D-scaling to closed-loop system $G_{c1}(s)$ with the modified scaling structure of (6.17). A set of scaling matrices δD_{fw} at specified frequency points is obtained. Then by interpolating $\delta D_{fw} D_f(j\omega)_{[i-1]}$ at all frequency points, the total frequency dependent D-scaling transfer function $D_{f1}(s)$ can be found.*
5. H_∞ *control design for the open loop system:*

$$G_2(s) = \begin{bmatrix} D_{f1}(s) & 0 & 0 \\ 0 & D_{c[i-1]} & 0 \\ 0 & 0 & I \end{bmatrix} G(s) \begin{bmatrix} D_{f1}(s) & 0 & 0 \\ 0 & D_{c[i-1]} & 0 \\ 0 & 0 & I \end{bmatrix}^{-1}.$$

A controller $K_2(s)$ is obtained and the corresponding closed loop system is $G_{c2}(s) = \mathcal{T}(G_2(s), K_2(s))$.

6. *Constant D-scaling to the closed loop system $G_{c2}(s)$. A scaling matrix $\delta D = \mathrm{diag}(\delta D_f, \delta D_c)$ is obtained.*

7. *Updating:* $D_f(s)_{[i]} = \delta D_f D_{f1}(s)$, $D_{c[i]} = \delta D_c D_{c[i-1]}$, $K_{[i]} = K_2$, $G_{c[i]} = \mathcal{T}(G, K_{[i]})$.

8. **Until** (1) $\|G_{c[i]}\|_D < 1$ *or* (2) $\mid \|G_{c[i]}\|_D - \|G_{c[i-1]}\|_D \mid < \epsilon$, *where*

$$\|G_{c[i]}\|_D \triangleq$$

$$\left\| \begin{bmatrix} D_f(s)_{[i]} & \\ & D_{c[i]} \end{bmatrix} \mathcal{T}(G(s), K_2(s)) \begin{bmatrix} D_f(s)_{[i]} & \\ & D_{c[i]} \end{bmatrix}^{-1} \right\|_\infty ,$$

and ϵ is a small positive number for the convergence test.

where $G(s)$ is the transfer function for the given nominal plant.

Remarks: 1) Step 3 and 5 are the standard H_∞ control designs [28]. Step 4 is for frequency D-scaling, while step 6 is for constant D-scaling. 2) In each step (frequency dependent and constant scaling, and two H_∞ control designs), we find an optimal solution (D_f, D, K_1 or K_2). But these solutions are not jointly optimal. So we can only find a local minimum. 3) In each step (or in each iteration), the objective function, the scaled H_∞ norm of the closed loop system, is no larger than the last step (or the last iteration). So the D-K iteration procedure with mixed D-scalings is guaranteed to converge to some equilibrium point. This is the same as the standard D-K iteration.

6.4 A practical example

Now we use the D-K iteration with mixed D-scalings to design a robust controller for an hydraulic servo system for a continuously variable transmission to be described in Chapter 7. The open loop system $G(s)$ is realized as follows:

$$A = \begin{bmatrix} -1336.90 & -2100.00 & 0.00 & \\ 53.48 & 0.00 & 0.00 & \\ 0.00 & -80000.00 & -25605.85 & \\ -7486.63 & 0.00 & 0.00 & \\ -802.14 & 0.00 & 200.57 & \\ 0.00 & 0.00 & 352357.86 & \\ 0.00 & 0.00 & 0.00 & \end{bmatrix}$$

$$\begin{bmatrix} 7973.28 & 8983.35 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 17966.70 & 300.00 & 0.00 \\ -17085.60 & 0.00 & 0.00 & 0.00 \\ 0.00 & -17966.70 & 0.00 & 0.00 \\ 0.00 & 0.00 & -5270.40 & 0.00 \\ 0.00 & 0.00 & -0.50 & -0.30 \end{bmatrix},$$

$$B = \begin{bmatrix} B_w & B_u \end{bmatrix} =$$

$$\begin{bmatrix} 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 14.0 & 1.5 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & -1.0 & -1.0 & 0.0 & -1.0 & -1.0 & 0.0 & 3.0 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -30.0 & 0.0 & 30.0 \\ 0.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -3.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 17.6 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.5 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix},$$

$$C = \begin{bmatrix} C_z \\ C_y \end{bmatrix} =$$

$$\begin{bmatrix} 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.5 & 29.7 \\ 0.0 & 0.0 & 0.0 & -5695.2 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 133.7 & 0.0 & -11977.8 & 0.0 & 0.0 \\ 0.0 & 0.0 & 13371.2 & 0.0 & 0.0 & -200.0 & 0.0 \\ 0.0 & 0.0 & 153595.0 & 0.0 & 0.0 & -2297.4 & 0.0 \\ 0.0 & 0.0 & 2674.2 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 40000.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 60.9 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 706.4 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -1.0 & 0.0 \end{bmatrix},$$

$$D = \begin{bmatrix} D_{zw} & D_{zu} \\ D_{yw} & D_{yu} \end{bmatrix} =$$

$$\begin{bmatrix}
0.5 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -10.0 & 0.0 & 10.00 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -2.0 & 0.00 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \\
0.0 & 0.0 & 0.0 & 7.7 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \\
1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00
\end{bmatrix}.$$

The last input/output pair is a control/measurement pair. The first input/output pair is a command/regulated-output(frequency weighted tracking-error) pair, the performance specification for this pair is that the L_2 gain should be smaller than 1. The remaining 8 inputs and outputs are connected to uncertainties. The uncertainty structure includes full blocks only with dimension 1×1 for each. All uncertainties are normalized in a unit ball with the loop shift and multiplier techniques. The first 3 uncertainties (connected to the 2nd to 4th input/output pairs) corresponding to hydraulic resistors in orifices are SISO, memoryless and incremental non expansive, which can be considered as LTI uncertainties, and the 4th uncertainty (connected to 5th input/output pair), corresponding to a hydraulic capacitor of a cylinder with fixed but unknown volume, is LTI; The last four uncertainties, corresponding to hydraulic capacitors of cylinders with time varying volume and hydraulic resistors for a valve with time varying opening, are NLTV uncertainties. Thus the dimension of $\mathbf{\Delta}_l$ (and $\mathbf{D}_f$) is 5 and the dimension of $\mathbf{\Delta}_n$ (and $\mathbf{D}_c$) is 4. In the mixed D-K iteration, we used MATLAB, μ toolbox [4] for H_∞ control design and frequency dependent D-scaling and MATLAB, LMIlab toolbox [38] for constant D-scaling. As the order of the scaled open loop system is rather high (order of $G(s)$ is 7, if a 3rd order curve fit is used, then the scaled open loop is of order $7+3\times5\times2=37$), so is the closed loop system (with order $2 \times 37 = 74$), thus the number of decision variables for the LMI optimization is very large ($74 \times 75 + 9 = 5559$), and computation for the LMI optimization is very slow. For this reason, a model reduction procedure is used before the constant D-scaling step. Fig. 6.4 shows the maximum singular value plot of the scaled closed loop systems for each iteration. The H_∞ norm for the closed loop system with the H_∞ controller without scaling is 37.47. For the first 3 iterations, the H_∞ norms of the scaled closed loop systems are 1.908, 1.067, 0.9863, and after the step 5 in

the 4th iteration it goes to 0.950 which satisfies the robust stability and performance requirement. Fig. 6.4 also shows that the maximum singular value plot becomes more and more flat, actually there is no significant decrease if the iteration continues. For comparison, we also used D-K iteration with constant D scaling for this control design problem. After 12 iterations, the H_∞ norm of the (constant) scaled closed loop system is 1.27 and the maximum singular value plot is shown in Fig. 6.5. The result is obviously conservative.

6.5 Conclusions

A D-K iteration procedure with mixed frequency and constant D-scalings was introduced. This procedure can be used to synthesize robust control for systems with both LTI and NLTV uncertainties. An industrially motivated example shows that the result with this D-K iteration is not so conservative as with the D-K iteration with constant D-scalings, and the convergence behavior of the method is guaranteed as the standard D-K iteration. However, this heuristic algorithm can not guarantee the global optimization. The problem we had in running the procedure is that compared with the frequency scaling step (step 4), the constant scaling step (step 6) is very slow, because of the high order of the scaling $D_f(s)$, even when the model reduction is used in this step to keep the closed loop system with order 30 (which will not significantly change the H_∞ norm).

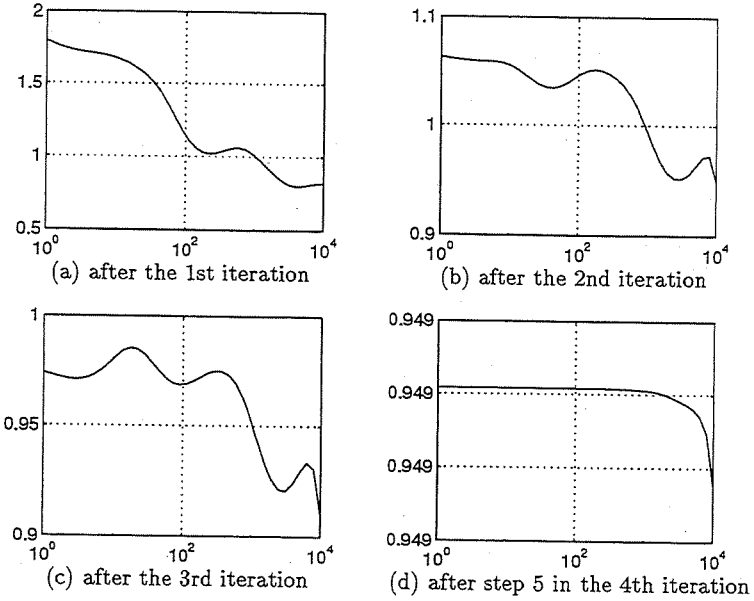


Figure 6.4 Maximum singular value plots for the scaled closed loop system in each D-K iteration with a mixed D-scaling. y-axis is magnitude and x-axis is frequency (rad/s). After the step 5 in the 4th iteration, the H_∞ norm of the scaled closed loop system converges to 0.950.

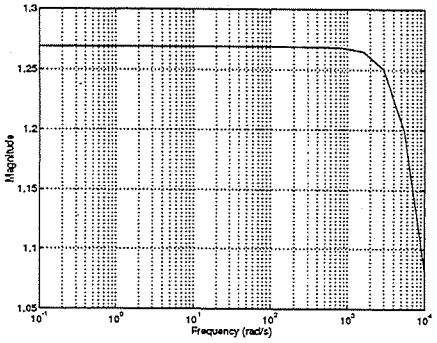


Figure 6.5 Maximum singular value plot for the scaled closed loop system after the 12th iteration with a constant D-scaling

Chapter 7

Application to an hydraulic servo system

In this chapter,we shall present the results on robust control analysis and design for an hydraulic servo system of a continuously varying transmission. The hydraulic system is subject to many uncertainties because of nonlinearities, changes in temperature and errors in specified data. Firstly all these uncertainties are modeled by the methods introduced in Chapter 4 and Chapter 5 and the whole system is represented by a linear fractional transformation. Secondly four control designs are proposed: classical PID control, H_∞ (sub)optimal control, μ synthesis with D-K iteration, μ synthesis with PID parameter search. Finally for each of designs, the robustness of stability and performance is analyzed with methods introduced in Chapter 6.

The structure of this chapter is arranged as follows. In section 7.1, the descriptions of the process and uncertainties are given. In section 7.2, process modeling will be described, especially the focus will be put on modeling uncertainties from hydraulic resistors and capacitors. The performance specification and the open loop interconnection will also given in this section. Four control designs will be presented in section 7.3. Section 7.4 will evaluate the four control designs via simulations, and finally some conclusions are made in section 7.5

7.1 Process and uncertainties

A continuously variable transmission (CVT) scheme is shown in Fig.7.1. The specific CVT-part, variator, is a pair of steel V shaped pulleys (primary and secondary pulleys) connected by a steel belt which is used to transfer torque between the pulleys. Each pulley consists of one fixed sheave (the left one for the secondary pulley) and one axially slidable sheave (the right one for the secondary pulley). The two moving sheaves are connected to two corresponding hydraulic cylinders/pistons. The engine shift is coupled, through a multi-plate wet clutch with the primary pulley. The secondary pulley finally is connected to the vehicle wheel. Clearly the positions of the slidable sheaves, which determine the transmission ratio, and the clipping force on the belt, which determines the maximum torque to be transferred, are determined by the pressures on the slidable sheaves of the pulleys (p^{sc}, for the secondary pulley). The ultimate powertrain control problem (optimal shifting) is to design a control system for the CVT such that some performance specifications for vehicles (such as comfortable driver feeling, lower fuel consumption, small acceleration time) can be satisfied. To do so, the control system is divided up into three levels: powertrain level, powertrain component level and actuator level. At the powertrain level, signals of powertrain components (variator and clutch) are used as control signals (variator ratio and clutch position) to realize optimal shift. At the powertrain component level, actuator signals (pressure signal p^{sc}, p^{pr} and p^{cl}) generated by hydraulic servo systems are acted as control signals to control the powertrain components . Finally at actuator level, the hydraulic servo systems are controlled via electro-hydraulic actuator by electric signals. See [83] for more details on the hierarchical control strategy. Here we are interested only in designing a part of actuator level control system to control the secondary cylinder pressure p^{sc}. The hydraulic circuit, which generates p^{sc}, is shown in Fig. 7.2 together with the symbols that will be used to describe its components, where G_{r1}^{sv}, G_{r2}^{sv}, G_{r3}^{sv} and G_r^{sc} denote hydraulic resistors; C_{c1}^{sv}, C_{c2}^{sv}, C_c^{sc} and C_c^{ch} denote hydraulic capacitors; I_m^{sc} and I_m^{sv} denote mechanical inductors or translational objects (here the pistons); C_k^{sv} and C_k^{sc} denote the mechanical capacitors or linear massless springs; and R_f^{sc}, R_f^{sv} denote sliding friction resistors. The secondary circuit consists of one valve and one cylinder connected with a channel. The inputs

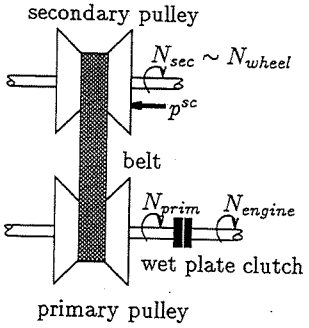


Figure 7.1 A continuously variable transmission scheme.

to it are the oil flow q^{sv} from the oil pump driven by the engine with rotation speed N_{engine}, the force f_1^{sc} from the primary cylinder (not shown in Fig. 7.2), the clutch pressure p^{cl} and the servo pressure p_1^{sv} from the controller. The only output is the secondary pressure p^{sc} in the secondary cylinder, which is the variable to be controlled. Assuming that q^{sv} and p^{cl} are constant and the piston in the secondary cylinder is fixed (but its position remains unknown), we then have a SISO system with the servo pressure p_1^{sv} as the input and the secondary pressure p^{sc} as the output. The process operates as follows: when the servo pressure p_1^{sv} is high, the piston I_m^{sv} in the valve will move towards the left, making a larger opening of the valve or a smaller resistance of the opening, consequently the pressure in the channel is lower, resulting in a lower pressure p^{sc} in the cylinder. When the servo pressure p_1^{sv} is low, the result is just opposite. So we can control the secondary pressure p^{sc} via the servo pressure p_1^{sv}. The actuator, which transfers electrical servo signal (from controller) into hydraulic servo pressure p_1^{sv} , is a pulse width modulated (PWM) valve. The PWM frequency is high, compared to the bandwidth of the hydraulic system. The dynamics of the PWM valve is neglected. The dynamics of the pressure sensor is also omitted as its response time is small (< 1 ms).

Now the setup for our control problem can be described by the so-

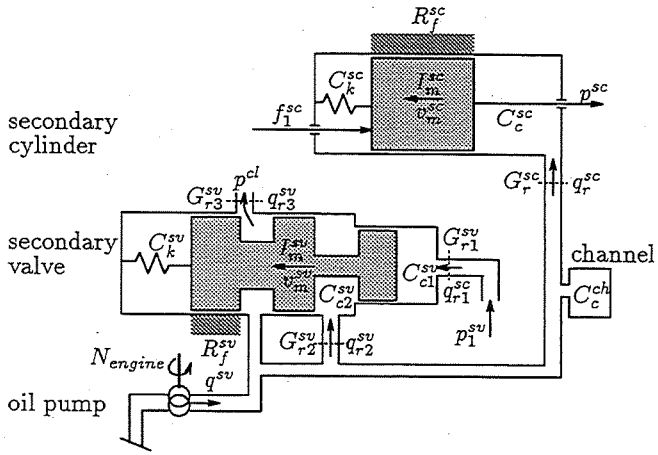


Figure 7.2 The secondary valve hydraulic circuit of CVT.

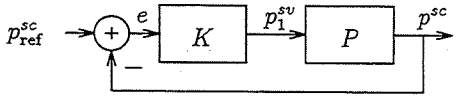


Figure 7.3 The block diagram of the control system.

called classical 1-DOF (one degree-of-freedom) control system as shown in Fig. 7.3, where p_{ref}^{sc} is the desired secondary cylinder pressure, e is the tracking error, K is the controller to be designed, P is the model of the process. Our control problem is the so-called tracking problem: find a stabilizing controller K such that the weighted H_∞-norm from p_{ref}^{sc} to e is minimized (see section 7.2.4 for details).

To model the process P, all components (resistors, capacitors and inductors) are modeled by input/output function blocks based on hydraulic and mechanical laws. The global process is modeled with block diagrams by connecting all these blocks via the bond graph method [47]. Since for

control analysis and synthesis only LTI nominal models are considered, thus those nonlinear component models are consequently replaced with LTI models. In the process of doing so, one is confronted with the following uncertainties:

- *The uncertainty of the temperature:* The hydraulic resistance is a function of the oil density (ρ) and viscosity (ν). Both oil density and viscosity change with the temperature, so does the resistance. As the temperature often changes from time to time, it may be quite different at the starting point and at the normal working point. However, the resistance change caused by the temperature is certainly bounded, for the temperature change is always bounded.

- *The uncertainty of the specified design data:* All of the components (resistors, capacitors and inductors) are modeled based on specified design data. These specified design data however are not always so accurate. For example, the compliance of the spring can be different from the specified design because of the possible variance in the production procedure. Again, this uncertainty is bounded and can be provided by the manufacture factory. The uncertainty of the constant capacitance of C_c^{sc} belongs to this class of uncertainties, as the volume of it is unknown but bounded.

- *Nonlinearities:* As we see from Fig. 7.2, the hydraulic capacitors (C_{c2}^{sv} and C_{c1}^{sv}) are time varying as the volumes of the corresponding cylinders change with the piston's position so the capacitances are variable; further more, all the hydraulic resistors (G_{r1}^{sv}, G_{r2}^{sv}, G_{r3}^{sv} and G_r^{sc}) and the slipping friction resistor (R_f^{sv}) are nonlinear. To design a controller with methods for LTI systems, we use some LTI models for these component functions. The model errors are bounded for the time varying components (here the capacitors) because the working ranges for these components are actually limited. For example, the volumes of the cylinders are limited, so are the corresponding hydraulic capacitances. For the nonlinear components (here the resistors), the model errors can be unbounded or too large theoretically. In this case, we make some assumptions on the bounds and check the results by simulation afterwards.

7.2 Uncertainty modeling

This section will focus on uncertainty modeling for the hydraulic servo system. But first we briefly summarize system modeling techniques used to build up the original NLTV system model in MATRIX$_x$ Super Block format.

7.2.1 System modeling

The system here consists of several mechanical, hydraulic subparts and even electrical subparts if we consider the actuators. For such a complex system, the bond graph technique [47] is used for system modeling. When constructing a bond graph model, one can split up two different aspects: Creating a bond graph gives form to the modeling phase to obtain a true representation of the physical dynamic system; analysis of the computational causality of the bond graph will result in a computable form of the model. In constructing a bond graph , the main concern will be the calculation of the average volume of flow rates through orifices and pressures in the hydraulic subsystem, and the macro fluid dynamic interaction with the solid mechanical elements(valves, cylinders,...). Under these approximations an exact bond graph representation is not necessary, and the familiar lumped parameter model is used because the hydraulic subsystem is in fact a servo mechanism for which the the volume flow rates are rather low and the high order dynamics of the hydraulic inertia can be neglected.

We used the software package MATRIX$_x$ [55] for modeling and simulation. But the input data for MATRIX$_x$ must consist of (causal) block diagrams or alike-structure, not a bond graph. The conversion of a bond graph to a block diagram is thus necessary. We are not going to show how these block diagrams were constructed, but refer to [56] for more details. The complete system model is in Appendix B.

The system model built in this way is nonlinear with some uncertainties (temperature change, design data variation and so on). We will use this model for simulation and to verify time responses of the control designs.

However, the system model in Appendix B may not used directly for robust control analysis and design since uncertainties and nonlinearities

are inside the model. Thus uncertainty modeling procedures introduced in Chapter 4 and Chapter 5 will be used here for separation and normalization of the uncertainties.

7.2.2 Practical procedure of uncertainty modeling

There are basically two steps for uncertainty modeling: uncertainty separation and normalization using the methods introduced in Chapter 4 and Chapter 5. For the separation algorithm in Chapter 3 a system is supposed to be described in a transfer function with uncertainty parameters. While in our case here, the system has been represented by an interconnection of transfer block diagrams in MATRIX$_x$ data format. One could, of course, write down the corresponding transfer function from an interconnection of transfer block diagrams, then use the algorithm in Chapter 4. It is however much easier to do it directly with the transfer block diagrams in MATRIX$_x$. For the normalization step, since we can easily construct a block diagram for an LFT, we can also do the normalization step with the transfer block diagrams. Thus whole uncertainty modeling can be done with the transfer block diagrams. The procedure is described as follows:

1. Find an LFT coefficient transfer matrix L_i for each uncertainty block Δ_i with the method described in Chapter 5 for uncertainty normalization.

2. Replace each uncertainty block diagram with coefficient transfer matrix L_i and define a new input/output pair which is connected to the normalized uncertainty, as shown in Fig. 7.4, where the dotted box is not really connected.

3. Get the system matrices (A,B,C,D) from the final interconnections (now it is an LTI system). Multidimension size reduction then can be done by the algorithm in Chapter 4.

Remarks: 1) This procedure is not only for repeated scalar uncertainties, but full block uncertainties as well. 2) To get system matrices, we only need to run command 'lin' in MATRIX$_x$.

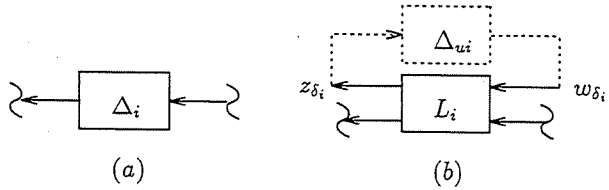


Figure 7.4 Replace an uncertainty block (Δ_i) with an LFT and define a new input (w_{δ_i}) and a new output (z_{δ_i}).

7.2.3 Modeling the uncertainties of the hydraulic circuit of CVT

We assume that the design data are correct, so the uncertainties are from the nonlinearities and the temperature changes. The capacitance of C_c^{sc} is constant but its value is unknown. The capacitances of C_{c1}^{sv} and C_{c2}^{sv} are variable. The capacitances of C_c^{ch} and C_k^{sv} are known and constant. All the hydraulic resistors are nonlinear. The sliding resistance of R_f^{sv} is rather small and assumed to be linear as a function of velocity. So uncertainty modeling is needed for all variable or unknown capacitors and hydraulic resistors. The uncertainty modeling procedures for all the capacitors C_c^{sc}, C_{c1}^{sc} and C_{c2}^{sc} are similar, and the uncertainty modeling procedures are also similar for the resistors G_r^{sc}, G_{r1}^{sv} and G_{r2}^{sv}. So in what follows, we will only show the modeling procedures for the capacitor C_c^{sc}, the resistors G_r^{sc} and G_{r3}^{sv}.

Uncertainty modeling for C_c^{sc}

We redraw the block diagram of C_c^{sc} in Fig. 7.5(a), where the dashed box represents the inverse of the capacitance. The uncertainty modeling procedure is described in the following steps, which are illustrated in Fig. 7.5

1. Calculate the maximum and minimum values of the inverse of the capacitance in the dashed box. This is easy as we know that the

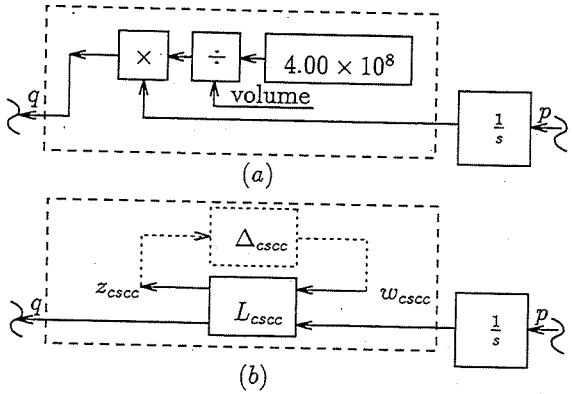


Figure 7.5 Uncertainty modeling of C_c^{sc}. (*a*) The original model. (*b*) Replace with an LFT (L_{cscc}) and define a new input (w_{cscc}) and a new output (z_{cscc}).

maximum and minimum volumes of the cylinder are 400 ml and 158 ml respectively and the bulk modulus of the oil is 4×10^3 (bar) from the design specification, so

$$(\frac{1}{C})_{\max} = \frac{4 \times 10^3}{158} (\text{bar/ml}^3),$$

$$(\frac{1}{C})_{\min} = \frac{4 \times 10^3}{400} (\text{bar/ml}^3).$$

That is $\frac{1}{C}$ in a conic sector $\begin{bmatrix} \frac{1}{C_{max}} & -1 \\ \frac{1}{C_{min}} & -1 \end{bmatrix}$, from (5.18) in section 5.5, the coefficient matrix (not a transfer matrix since the uncertainty is not a dynamic system) of the LFT for normalizing is:

$$L_{cscc} = \begin{bmatrix} 0 & 1 \\ \frac{(\frac{1}{C})_{\max} - (\frac{1}{C})_{\min}}{2} & \frac{(\frac{1}{C})_{\max} + (\frac{1}{C})_{\min}}{2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 17.7 & 7.66 \end{bmatrix}.$$

2. Replace with the LFT (L_{csc}) and define a new input (w_{csc}) and a new output (z_{csc}).

Uncertainty modeling for G_r^{sc}

We use the admittance g of a resistor which is defined

$$g = \frac{q}{p} \text{ so } q = gp$$

where p is the pressure and q is the flow. The relation between input pressure p and output flow q for resistor G_r^{sc} is described by [56]

$$q = \min \left(0.611; \sqrt{\frac{0.611^3 D_h}{20\nu(T)} \sqrt{\frac{2p}{\rho(T)}}} \right) a_r \sqrt{\frac{2p}{\rho(T)}}, \tag{7.1}$$

where D_h is a constant depending on the geometry of the channel, a_r is the cross section area of the channel and $\nu(T)$ and $\rho(T)$ are oil viscosity and density respectively. The uncertainty modeling procedure for G_r^{sc} is exactly the same as that for C_c^{sc}, except that the first step is not so trivial. The output flow q is not only a function of the oil density and viscosity, which themselves are functions of the temperature T, but it is also a complicated function of the input p. We already knew that the the pressure p on G_r^{sc} is in the range $|p| < 40$ bar, the temperature is in the range $-20°C < T < 130°C$ in practice and oil density $\rho(T)$ and viscosity $\nu(T)$ are monotonically decreasing functions of the temperature T as shown in Fig. 7.6. With (7.1) we easily find:

$$g_{max} = \left.\frac{q}{p}\right|_{p=0(\text{bar}),\ T=130°C},$$

$$g_{min} = \left.\frac{q}{p}\right|_{p=40(\text{bar}),\ T=-20°C}.$$

When $p \to 0$, $g_{max} \to \infty$, thus g is passive with index g_{min}. So there is no problem for uncertainty modeling by using (5.17) in section 5.5 ($b^{-1} = 0, a = g_{min}$ and $d = 1$, *e.g.*). However, in practice, when pressure p is low, flow q will be proportional to the pressure p and g will be constant. Therefore in calculating g_{min}, we assume that the minimum

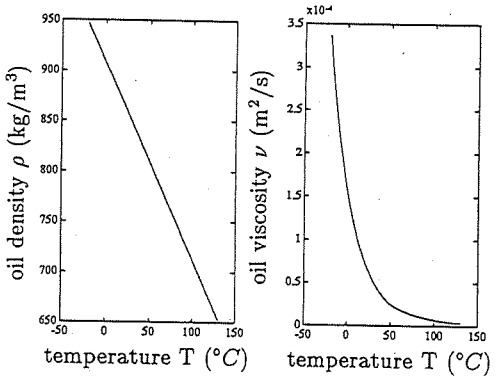


Figure 7.6 Oil density $\rho(T)$ and viscosity $\nu(T)$.

pressure p is 2.5 bar, which is equivalent to assume that g is constant when $p < 2.5$. D_h is 0.01 from the specification, then we have:

$$g_{max} = \frac{q}{p}\Big|_{p=2.5(\text{bar}),\ T=130°C} \simeq 500(\text{ml}^3 \cdot \text{s/bar}),$$

$$g_{min} = \frac{q}{p}\Big|_{p=40(\text{bar}),\ T=-20°C} \simeq 100(\text{ml}^3 \cdot \text{s/bar}).$$

The remaining steps of uncertainty modeling are the same as those for C_c^{sc}, and we obtain

$$L_{gscr} = \begin{bmatrix} 0 & 1 \\ \frac{g_{max}-g_{min}}{2} & \frac{g_{max}+g_{min}}{2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 200 & 300 \end{bmatrix}.$$

The block diagram is shown in Fig. 7.7. Note that g is also incremental gain bounded with the same maximum and minimum values.

Uncertainty modeling of G_{r3}^{sc}

The pressure and flow relation for resistor G_{r3}^{sc} is similar to (7.1), but with

$$a_r = a_r(x);\ D_h = \sqrt{\frac{4}{\pi} a_r(x)}, \tag{7.2}$$

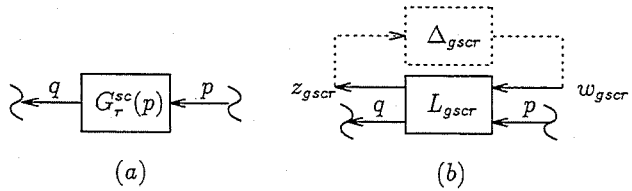


Figure 7.7 Uncertainty modeling of G_r^{sc}. (*a*) Original model (*b*) Replace with an LFT (L_{gscr}) and define a new input (w_{gscr}) and a new output (z_{gscr}).

where x is the piston position and the function $a_r(x)$ is nonlinear as shown in Fig. 7.8. Now we write q as:

$$q = (\lambda \frac{q}{x})x + ((1-\lambda)\frac{q}{p})p, \tag{7.3}$$

where $0 < \lambda < 1$. Let

$$g_x = \lambda \frac{q}{x},\ \ g_p = (1-\lambda)\frac{q}{p}. \tag{7.4}$$

Then (7.3) becomes:

$$q = g_p p + g_x x. \tag{7.5}$$

Next we try to find the ranges for g_x and g_p. Simulations show that

$$0.01\ (\text{m}) < x < 0.02\ (\text{m}),\ 2\ (\text{bar}) < p < 32\ (\text{bar}). \tag{7.6}$$

We choose $\lambda = 13/14$[1]. With (7.1), (7.2),(7.4), (7.6), and the opening function in Fig. 7.8, recalling that $\nu(T)$ and $\rho(T)$ are monotonic to T and

[1]The reason for this choice is as follows: Obviously we can choose any value in [0 1] for λ.. Different λ values will lead to different g_x and g_p. But for robust analysis and control design, we need to choose the value such that the results are the least conservative. Actually we choose λ such that a μ synthesis based control design, satisfying the robustness and performance specification, is feasible. (Experience has shown that μ synthesis is not possible for all values of λ.)

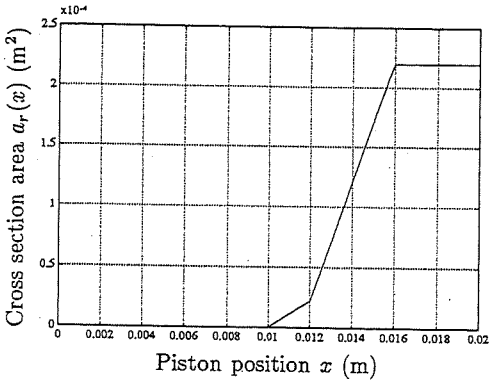


Figure 7.8 The opening function $a_r(x)$ of G_{r3}^{sc} as a function of the piston position x.

choosing x=0.0162 (m) at which $a_r(x)/x$ reaches its maximum, we can easily obtain

$$\begin{aligned} g_{xmax} &= \left.\frac{13q}{14x}\right|_{p=32(\text{bar}),\ T=130(^\circ C),\ x=0.0162(\text{m})} \\ &= 7.5\times 10^5(\text{ml}^3/\text{s/m}), \\ g_{xmin} &= \left.\frac{13q}{14x}\right|_{p=2(\text{bar}),\ T=-20(^\circ C),\ x=0.01(\text{m})} = 0, \\ g_{pmax} &= \left.\frac{q}{14p}\right|_{p=2(\text{bar}),\ T=130(^\circ C),\ x=0.02(\text{m})} = 120(\text{ml}^3/\text{s/bar}), \\ g_{pmin} &= \left.\frac{q}{14p}\right|_{p=32(\text{bar}),\ T=-20(^\circ C),\ x=0.01(\text{m})} = 0. \end{aligned}$$

However, this uncertainty modeling is very conservative. Indeed, there is no controller at all which stabilizes the system in the "worst case": $g_{xmin}=0$ and $g_{pmin}=0$ when the relation of input x and p and output q of G_{r3}^{sv} is described by (7.5) linearly. This is easy to see by inspection from the layout of the secondary hydraulic circuit in Fig. 7.2: the "worst case" of $g_{xmin}=0$ and $g_{pmin}=0$ means that there is no flow

passing through G_{r3}^{sv} at any time. As q^{sv} is constant, the pressure in the secondary cylinder will go to infinity whatever the input pressure p_1^{sv} is. The problem here is that we are not only dealing with uncertainties from changing temperature, but also the errors from nonlinearities, which are dependent on the system variables. To eliminate the conservativeness, g_{xmin} and g_{pmin} have to be increased, here we choose:

$$\begin{aligned} g_{xmin} &= \frac{g_{xmax}}{3} = 2.5\times 10^5(\text{ml}^3/\text{s/m}), \\ g_{pmin} &= \frac{g_{pmax}}{3} = 40(\text{ml}^3/\text{s/bar}). \end{aligned}$$

These assumptions mean that the secondary valve is never completely closed, and $x > 0.013$ (m). Although in steady state, these assumptions are true as the oil flow q^{sv} is assumed to be constant, dynamically, they do not always hold. Thus we need to check the results by closed loop simulation afterwards[2]. The remaining steps of the uncertainty modeling are the same as for C_c^{sc}, but with two LFT involved:

$$L_{gsvr3_x} = \begin{bmatrix} 0 & 1 \\ \frac{g_{x\max}-g_{x\min}}{2} & \frac{g_{x\max}+g_{x\min}}{2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 2.5\times 10^5 & 5\times 10^5 \end{bmatrix},$$

$$L_{gsvr3_p} = \begin{bmatrix} 0 & 1 \\ \frac{g_{p\max}-g_{p\min}}{2} & \frac{g_{p\max}+g_{p\min}}{2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 40 & 80 \end{bmatrix}.$$

Combining these two LFTs and (7.5), we eventually find one LFT for the uncertainty modeling:

$$L_{gsvr3} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 40 & 2.5\times 10^5 & 80 & 5\times 10^5 \end{bmatrix}.$$

Fig. 7.9 shows the block diagram.

[2] Here we choose g_{xmin} and g_{pmin} again in the control design phase as we did for λ, see footnote 1. However, unlike λ, g_{xmin} and g_{pmin} are no longer the true value (0,0). So robustness can not be sufficiently guaranteed in the control design phase. We can only check the results by simulation afterwards. We recognize that this is a disadvantage of using analysis and synthesis methods for LTI systems with bounded uncertainties to nonlinear uncertainty systems by taking nonlinearities as uncertainties when the nonlinearities are dominant

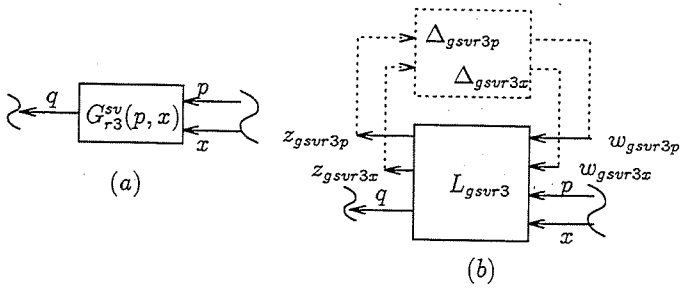


Figure 7.9 Uncertainty modeling of G_{r3}^{sv}. (*a*) Original model (*b*) Replace with an LFT (L_{gsvr3}) and define two new inputs (w_{gsvr3x},w_{gsvr3p}) and two new outputs (z_{gsvr3x}, z_{gsvr3p}).

The final interconnection

As all possible components with uncertainties and/or nonlinearities are modeled by an LTI system with some new inputs and outputs for the uncertainties, we can easily get the final interconnection scheme shown in Fig. 7.10 in MATRIX$_x$ Super Block format. Note that the inputs and outputs with number 1 to 8 are the outputs and inputs pairs for the uncertainties in the components described above respectively, see Table. 7.1. Only the last input and the last output are the system input and output. Each (real) uncertainty is connected to a corresponding input/output pair. The system transfer function $M(s)$ of the secondary hydraulic circuit now can be easily obtained from the interconnection in SuperBlock form. The following is the state space realization of $M(s)$,

$$M(s) = \left[\begin{array}{c|c} A_m & B_m \\ \hline C_m & D_m \end{array}\right],$$

where

$A_m =$

Table 7.1 Components and corresponding uncertainty input/output pairs.

Components	Label	
	input	output
G_{r1}^{sv}	w_{gsvr1}	z_{gsvr1}
G_{r2}^{sv}	w_{gsvr2}	z_{gsvr2}
G_{r3}^{sv}	w_{gsvr3_x}	z_{gsvr3_x}
	w_{gsvr3_p}	z_{gsvr3_p}
C_{c1}^{sv}	w_{csvc1}	z_{csvc1}
C_{c2}^{sv}	w_{csvc2}	z_{csvc2}
G_{r}^{sc}	w_{gscr}	z_{gscr}
C_{c}^{sc}	w_{csvc}	z_{csvc}

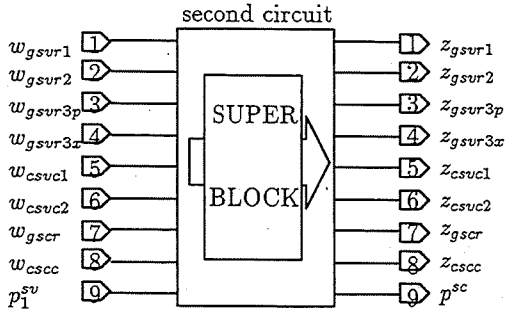


Figure 7.10 Final interconnection in Super Block form.

$$\begin{bmatrix} -1336.90 & -2100.00 & 0.00 & 7973.28 & 8983.35 & 0.00 \\ 53.48 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & -80000.00 & -25605.85 & 0.00 & 17966.70 & 300.00 \\ -7486.63 & 0.00 & 0.00 & -17085.60 & 0.00 & 0.00 \\ -802.14 & 0.00 & 200.57 & 0.00 & -17966.70 & 0.00 \\ 0.00 & 0.00 & 352357.86 & 0.00 & 0.00 & -5270.40 \end{bmatrix},$$

$B_m =$

$$\begin{bmatrix} 0.00 & 0.00 & 0.00 & 14.00 & 1.50 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & -1.00 & -1.00 & 0.00 & 3.00 & -1.00 & 0.00 & -1.00 & 0.00 \\ 1.00 & 0.00 & 0.00 & -30.00 & 0.00 & 0.00 & 0.00 & 0.00 & 30.00 \\ 0.00 & 1.00 & 0.00 & 0.00 & -3.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 17.57 & 1.00 & 0.00 & 0.00 \end{bmatrix},$$

$$C_m = \begin{bmatrix} 0.00 & 0.00 & 0.00 & -5695.20 & 0.00 & 0.00 \\ 0.00 & 0.00 & 133.71 & 0.00 & -11977.80 & 0.00 \\ 0.00 & 0.00 & 2674.24 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 60.90 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 706.40 & 0.00 \\ 0.00 & 0.00 & 13371.20 & 0.00 & 0.00 & -200.00 \\ 0.00 & 0.00 & 153594.97 & 0.00 & 0.00 & -2297.40 \\ 0.00 & 40000.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix},$$

$$D_m = \begin{bmatrix} 0.0 & 0.0 & 0.0 & -10.0 & 0.0 & 0.0 & 0.0 & 0.0 & 10.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & -2.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 7.66 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix}.$$

For this realization, the corresponding states represent some physical quantities shown in Table 7.2. As there are only scalar uncertainties, the

Table 7.2 Physical descriptions of the states.

State No.	Name	In component	Description
1	v_m^{sv}	I_m^{sv}	velocity
2	x	I_m^{sv}	position
3	p_c^{ch}	C_c^{ch}	pressure
4	p_{c1}^{sv}	C_{c1}^{sv}	pressure
5	p_{c2}^{sv}	C_{c2}^{sv}	pressure
6	p^{sc}	C_c^{sc}	pressure

block structure matrix is

$$bs = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}.$$

The frequency response for the nominal plant $(M(s))_{p^{sc}p_1^{sv}}$, which is the transfer function from p_1^{sv} to p^{sc}, is shown in Fig. 7.11.

7.2.4 Performance specification and open loop interconnection

The secondary hydraulic circuit of the CVT is designed as a servo system: the output, the secondary pressure p^{sc}, is to track a reference or command signal w_p (it is the same as p_{ref}^{sc} in Fig. 7.3). So the control problem for the secondary circuit is a tracking problem. Hence one can translate the performance specification as the requirement that the H_∞ norm from the reference input w_p to the tracking error e between w_p and p^{sc} is less than some specified value. Here we use the frequency weighted H_∞ norm with a frequency weighting function W_p. The resulting open loop interconnection of the secondary circuit system together with the weighting function is shown in Fig. 7.12 where G is the open loop transfer function. The

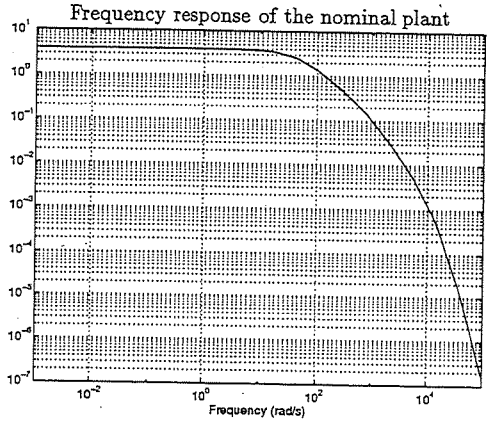


Figure 7.11 The frequency response for the nominal plant.

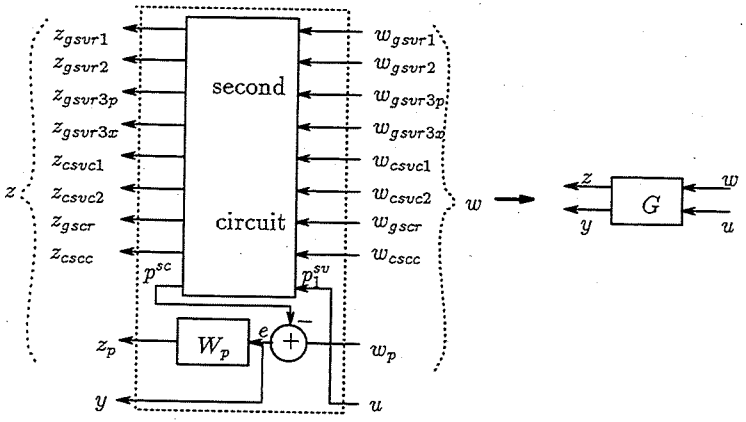


Figure 7.12 Open loop interconnection structure.

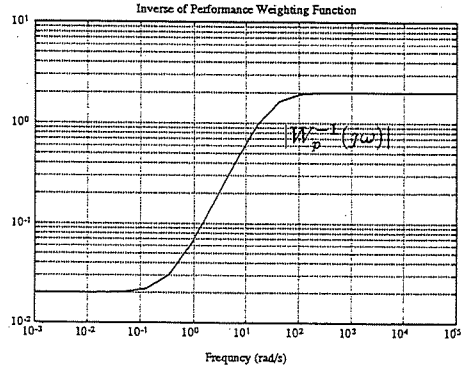


Figure 7.13 Frequency response of W_p^{-1}.

performance specification is:

$$\gamma_2 \left(\mathcal{T}(\Delta, G_c)\right) \leq 1, \tag{7.7}$$

where G_c is the closed loop (with a controller feedback from y to u) transfer function. (7.7) shows that the maximum singular value curve of the closed loop transfer function $\mathcal{T}(\Delta, G_c)W_p^{-1}$ from w_p to e is required to lie below the curve of $|W_p^{-1}(j\omega)|$ for any uncertainty $\Delta \in \mathbf{\Delta}$. We choose the weighting function as:

$$W_p(s) = 0.5\frac{s+30}{s+0.3},$$

the inverse of which has the frequency response curve shown in Fig. 7.13. It is easy to check that the ratio of the reference to tracking error in the steady state is less than 0.02. The zero of $W_p(s)$ is chosen such that the inverse of it $1/30 = 0.033$ is less than 0.05 second which is the specified rise time. For separating LTI or memoryless incremental non expansive uncertainties with nonlinear uncertainties, we rearrange input/output pairs as shown in Table 7.3. The last input/output pair is control/measurement pair. The first input/output pair is command/regulated-output(frequency

Table 7.3 Open system input/output pairs.

No.	input	output
1	w_p	z_p
2	w_{gsvr1}	z_{gsvr1}
3	w_{gsvr2}	z_{gsvr2}
4	w_{gscr}	z_{gscr}
5	w_{cscc}	z_{cscc}
6	w_{gsvr3_x}	z_{gsvr3_p}
7	w_{gsvr3_p}	z_{gsvr3_x}
8	w_{csvc1}	z_{csvc1}
9	w_{csvc2}	z_{csvc2}
10	u	y

weighted tracking-error). The remaining 8 inputs and outputs are connected to uncertainties. The open loop transfer function G in then has the following realization:

$$A = \begin{bmatrix} -1336.90 & -2100.00 & 0.00 \\ 53.48 & 0.00 & 0.00 \\ 0.00 & -80000.00 & -25605.85 \\ -7486.63 & 0.00 & 0.00 \\ -802.14 & 0.00 & 200.57 \\ 0.00 & 0.00 & 352357.86 \\ 0.00 & 0.00 & 0.00 \end{bmatrix}$$

$$\begin{bmatrix} 7973.28 & 8983.35 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 17966.70 & 300.00 & 0.00 \\ -17085.60 & 0.00 & 0.00 & 0.00 \\ 0.00 & -17966.70 & 0.00 & 0.00 \\ 0.00 & 0.00 & -5270.40 & 0.00 \\ 0.00 & 0.00 & -0.50 & -0.30 \end{bmatrix},$$

$$B = \begin{bmatrix} B_w & B_u \end{bmatrix} =$$

$$\begin{bmatrix} 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 14.0 & 1.5 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & -1.0 & -1.0 & 0.0 & -1.0 & -1.0 & 0.0 & 3.0 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -30.0 & 0.0 & 30.0 \\ 0.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -3.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 17.6 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.5 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix},$$

$$C = \begin{bmatrix} C_z \\ C_y \end{bmatrix} =$$

$$\begin{bmatrix} 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.5 & 29.7 \\ 0.0 & 0.0 & 0.0 & -5695.2 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 133.7 & 0.0 & -11977.8 & 0.0 & 0.0 \\ 0.0 & 0.0 & 13371.2 & 0.0 & 0.0 & -200.0 & 0.0 \\ 0.0 & 0.0 & 153595.0 & 0.0 & 0.0 & -2297.4 & 0.0 \\ 0.0 & 0.0 & 2674.2 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 40000.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 60.9 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 706.4 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -1.0 & 0.0 \end{bmatrix},$$

$$D = \begin{bmatrix} D_{zw} & D_{zu} \\ D_{yw} & D_{yu} \end{bmatrix} =$$

$$\begin{bmatrix} 0.5 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -10.0 & 0.0 & 10.00 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -2.0 & 0.00 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \\ 0.0 & 0.0 & 0.0 & 7.7 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \\ 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.00 \end{bmatrix}. \quad (7.8)$$

Note that the 7th state (corresponding to last column and row of matrix A, the last row of matrix B and the last column of matrix C) is

the state for weighting function $W_p(s)$; The block structure now is

$$bs = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}.$$

The first row of bs corresponds to the fictitious uncertainty block for the performance specification. The uncertainty structure includes full blocks only with dimension 1×1 for each. The first 3 uncertainties (connected to the 2nd to 4th input/output pairs) corresponding to hydraulic resistors in orifices are SISO, memoryless and incremental non expansive, which can be considered as LTI uncertainties, and the 4th uncertainty (connected to 5th input/output pair), corresponding to an hydraulic capacitor of a cylinder with fixed but unknown volume, is LTI; The last four uncertainties, corresponding to hydraulic capacitors of cylinders with time varying volume and hydraulic resistors for a valve with time varying opening, are NLTV uncertainties. Thus the dimension of $\mathbf{\Delta}_l$ (and $\mathbf{D}_f$) is 5 and the dimension of $\mathbf{\Delta}_n$ (and $\mathbf{D}_c$) is 4.

7.3 Control design and robustness analysis

In this section, we design controllers with various control design methods: classical PID auto-tuning, H_∞ synthesis, D-K iteration, PID parameter search. Robustness analysis of the closed loop system will be done for the performance first, if the performance robustness is not ensured, the stability robustness will be analyzed.

Robust performance will be analyzed by the unscaled, constant D scaled, mixed D scaled and frequency dependent D scaled H_∞-norm. If

$$\|G_c\|_D \triangleq \min_D \|DG_cD^{-1}\|_\infty < 1, \tag{7.9}$$

then the robust performance is satisfied, where the structures of scaling D are listed in Table 7.3. Table 7.3 also summarizes the corresponding

Table 7.4 Different D-scaling methods (u.b.: upper bound, str.: structured).

	unscaled	const. D	mixed D	freq. D
$D \in \mathbf{D}$	I	D_c	$\begin{bmatrix} D_f(j\omega) & \\ & D_c \end{bmatrix}$	$D_f(j\omega)$
method	H_∞-norm	LMI opt.	LMI opt.& μ u.b.	μ u.b.
Δ	any NLTV	str. NLTV	str. NLTV & LTI	str. LTI
$\|G_c\|_D$	largest	$\longrightarrow$		smallest

calculation methods, allowed uncertainty sets and comparison of values of $\|G_c\|_D$. Note that, as shown in Table 7.3 the frequency dependent D-scaling is the weakest condition, that is if for frequency dependent D-scaling (7.9) fails, then for any other D-scalings, (7.9) will fail.

For stability robustness analysis, partition the nominal closed loop system $G_c(s)$ as

$$G_c(s) = \begin{bmatrix} G_{pp}(s) & G_{p\delta}(s) \\ G_{\delta p}(s) & G_{\delta\delta}(s) \end{bmatrix}$$

where subscript "p" is corresponding to the performance input/output and "δ" is corresponding to the uncertainty inputs/outputs. Thus the system $G_c(s)$ is robust stable if the the following condition is satisfied:

$$\|G_{\delta\delta}(s)\|_D < 1$$

Note that D here is the mixed scaling and does not include the artificial block which corresponds to the performance specification.

We used the MATLAB software package for all design and analysis.

7.3.1 Classical PID auto-tuning

Auto-tuning PID control

A PID controller has a transfer function of the form:

$$K_{pid}(s) = K_p + \frac{K_i}{s} + \frac{K_d s}{T_d s + 1},$$

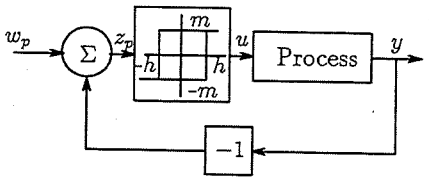


Figure 7.14 An auto-tuner with a hysteresis relay.

where K_p, K_i and K_d are parameters to be tuned. The PID tuning used here is based on the specification of gain and phase margins. To do so, the limit cycle method (LCM) is used with a hysteresis relay controller to generate oscillations (see Fig. 7.14), so that one point in the Nyquist curve can be determined [2]. When one point on the Nyquist curve is known, it is straightforward to find the parameters of a PID regulator which give the desired amplitude and phase margins. Fig. 7.15 shows the gain and phase margin specification where curve a denotes the Nyquist plot of $G(j\omega)$ and curve b that of $(K_p+j\omega K_d+K_i/j\omega)G(j\omega)$ respectively. We choose the gain and phase margins so that the Nyquist plot intersects the circle with a radius 0.5 at an angle 225°[3]. To do so, we should move the point $G(j\omega_n)$ (A in Fig. 7.15), ω_n is the oscillation frequency) to the point $0.5\underline{/225°}$ (B in Fig. 7.15). That means

$$(K_p + j\omega_n K_d + \frac{K_i}{j\omega_n})G(j\omega_n) = 0.5\underline{/225°}. \tag{7.10}$$

$G(j\omega_n)$ can be rewritten in terms of gain A_g and phase ϕ_g as $G(j\omega_n) = A_g\underline{/\phi_g + 180°}$. By solving (7.10) for K_p, K_d and K_i, we obtain:

$$K_p = \frac{0.5\cos(45° - \phi_g)}{A_g},$$

[3]During the tuning phase, we let $T_d = 0$, while it is set to 10^{-5} second for practical implementation, thus for control analysis later on, we let $T_d = 10^{-5}$

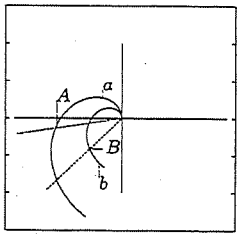


Figure 7.15 Amplitude and phase margin specification.

$$K_d = K_p\frac{\tan(45° - \phi_g) + \sqrt{\frac{4}{p} + \tan^2(45° - \phi_g)}}{2\omega_n},$$

$$K_i = \frac{K_p^2}{pK_d}.$$

Here p is set to 4, which is implied in the Ziegler and Nichols tuning rule [3]. So this method is assumed that the controlled system is LTI system, however it turns out that it is also acceptable for some nonlinear systems in practice [91], typically the tuned control parameters are often good initial values for further refining.

PID auto-tuning for the hydraulic servo system

The PID parameters were tuned on the original model (SISO nonlinear model), with the method described in section 7.3.1. The hysteresis width h and the amplitude m of the relay tuner were set to 10 bar and 2.5 bar respectively. The limit cycle experiment was done with this relay tuner at the point where the input servo pressure $p_1^{sv} = 2.5$ bar and the temperature 70°C. Then the PID parameters obtained from the experimental data and the algorithm as shown above were :

$$\begin{aligned} K_p &= -7.7143e-02, \\ K_i &= -5.7359e+00, \\ K_d &= -2.5976e-04. \end{aligned} \tag{7.11}$$

The corresponding PID controller is denoted as K_{pid}. The frequency response of K_{pid} is shown in Fig. 7.16.

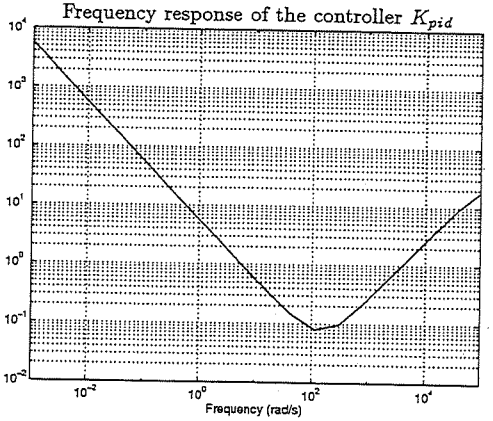


Figure 7.16 The frequency response for the controller K_{pid}.

Performance robustness

The frequency dependent D-scaling for the closed loop system is actually analyzed by the complex μ calculation at certain frequency points. The μ plot of the closed loop system is shown in Fig. 7.17. The maximum lower bound of μ is 1.4643, so $\|G_c\|_D > 1$. These results show that this PID controller settings can not guarantee the performance robustness since there exists no D such that $\|G_c\|_D$ is smaller than one. This result is not surprising since during the tuning phase, we set PID controller such that the Nyquist plot passes one point $0.5/225°$ which is not directly related to our performance specification.

Stability robustness

Now we analyze the robust stability of the closed loop system. By the mixed D-scaling method, $\|G_{\delta\delta}\|_D = 1.021$, the singular value plot of the scaled $G_{\delta\delta}(j\omega)$ is shown in Fig. 7.18. Since $\|G_{\delta\delta}\|_D > 1$, the closed loop system might be not stable for some uncertainty thus the stability robustness is not guaranteed. However it does not necessarily mean that

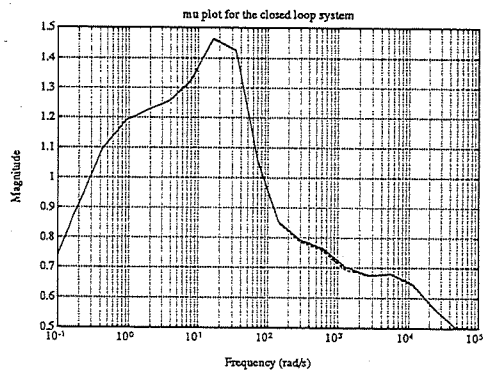


Figure 7.17 μ plot of the closed loop system G_c with PID controller K_{pid} with $K_p = -7.7143e-02, K_i = -5.7359e+00, K_d = -2.5976e-04$. The solid curve is for the upper bound and the dashed curve is for the lower bound.

the closed loop system G_c is always unstable. Actually, later on we shall show by time simulations that G_c is stable for the original nonlinear system with a fixed temperature.

7.3.2 H_∞ control

For H_∞ control design, the following conditions have to be satisfied [28]: (1) (A, B_w) and (A, B_u) are stabilizable; (2) (C_z, A) and(C_y, A) are detectable; (3) D_{zu} has full column rank; (4) D_{yw} has full row rank. From the open loop system matrices A, B, C and D, these conditions can be easily checked: All eigenvalues of A are in the left half plane. The matrix pairs (A, B_w) and(A, B_u) are stabilizable, and (C_z, A) and (C_y, A) are detectable. D_{zu}, $[\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,]^T$, and D_{yw}, $[\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,]$, have full rank. An H_∞ controller, denoted as K_∞ is designed by the algorithm in [28]:

$$K_\infty = \left[\begin{array}{c|c} A_k & B_k \\ \hline C_k & D_k \end{array}\right],$$

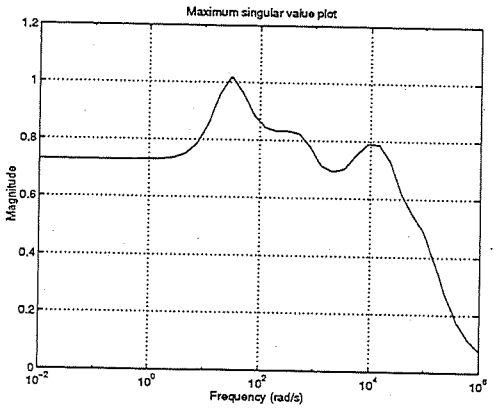


Figure 7.18 The maximum singular value plot of $G_{\delta\delta}$ with the PID controller K_{pid} with $K_p = -7.7143e-02, K_i = -5.7359e+00, K_d = -2.5976e-04$.

where

$$A_k = \begin{bmatrix} -1.1432e+03 & 1.8068e+04 & 6.4020e+02 \\ 5.2956e+01 & -5.3121e+01 & -1.2731e+01 \\ 2.7810e+00 & -7.9723e+04 & -2.3587e+04 \\ -1.7992e+04 & -1.0939e+06 & -6.6180e+02 \\ -7.9775e+02 & 4.5658e+02 & -1.5132e+03 \\ 2.0336e+01 & 1.3167e+03 & 3.5992e+05 \\ 4.6968e-18 & 1.6152e-15 & 2.3254e-15 \end{bmatrix}$$

$$\begin{bmatrix} 8.2151e+03 & 8.7044e+03 & 7.5962e+00 & 1.4792e+00 \\ -6.4892e-01 & 7.0538e-01 & 1.4423e-01 & -3.9677e-03 \\ 3.4736e+00 & 1.8000e+04 & 2.7005e+02 & 2.1238e-02 \\ -1.3119e+04 & 1.5409e+04 & -9.2129e+02 & -8.0225e+01 \\ 5.4860e+00 & -1.7994e+04 & 2.6018e+01 & 3.3543e-02 \\ 2.5400e+01 & -2.5141e+02 & -5.3816e+03 & 1.5530e-01 \\ 5.8664e-18 & -4.3304e-18 & -1.1063e-16 & -3.0000e-01 \end{bmatrix},$$

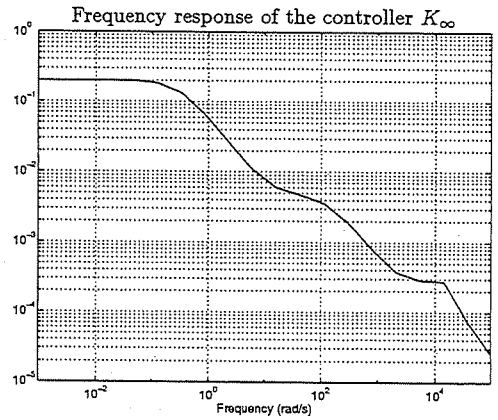


Figure 7.19 The frequency response for the controller K_∞.

$$B_k = \begin{bmatrix} -3.2465e-01 \\ 4.7571e-02 \\ -8.5214e-01 \\ -9.2757e-02 \\ 1.3643e-01 \\ -5.8988e+01 \\ 3.3857e+01 \end{bmatrix}, C_k^T = \begin{bmatrix} -5.0184e+00 \\ -5.2254e+02 \\ -1.7246e+00 \\ 2.1435e+00 \\ 7.3467e+00 \\ -4.1905e-01 \\ -3.8325e-02 \end{bmatrix}, D_k = 0.$$

It is of order 7, which is the same as the order of the plant G. It is already known in the design process that the H_∞-norm of the closed loop system is 37.5. The frequency response of K_∞ is shown in Fig. 7.19.

Performance robustness

The μ plot is shown in Fig. 7.20. The maximum lower bound of μ is 27.848, that is $\|G_c\|_D > 1$. Thus the closed loop robust performance may not be guaranteed with such a controller. Actually it is far from being satisfied since the peak μ value is too large.

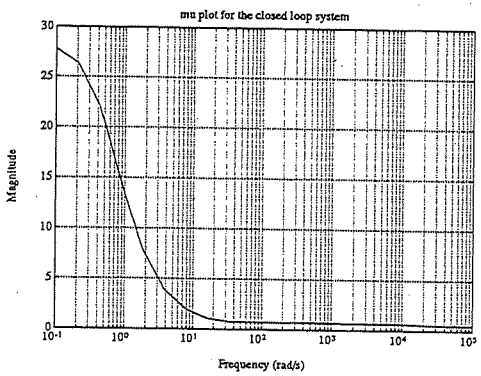


Figure 7.20 μ plot of the closed loop system with the H_∞ controller K_∞.The solid curve is for the upper bound and the dashed curve is for the lower bound.

Stability robustness

However, by the mixed D-scaling, we find that $\|G_{\delta\delta}\|_D = 0.7476$, that is the robust stability is actually guaranteed. Fig. 7.21 show the maximum singular value plot for the scaled $G_{\delta\delta}$.

7.3.3 μ control synthesis: standard D-K iteration

A second order transfer function matrix (with each entry being a second order transfer function) fit was used in each iteration for each block. In general, the higher the order of fitting, the more accurate it will be. However, for high order functions, there may be numerical problems and, typically, it results in high order controllers. Even with second order fitting, the resulting controller will be of order 39. (The scaled open loop system is DGD^{-1}, D is diag(d_1 d_2 d_3 d_4 d_5 d_6 d_7 d_8 d_9 1) with $d_i, i = 1 \cdots 8$ being second order transfer functions and d_9 equal to 1, G is of order 7, thus the scaled open loop system DGD^{-1} is of order 39 which is equal to the order of the controller). After 12 D-K iterations, a controller

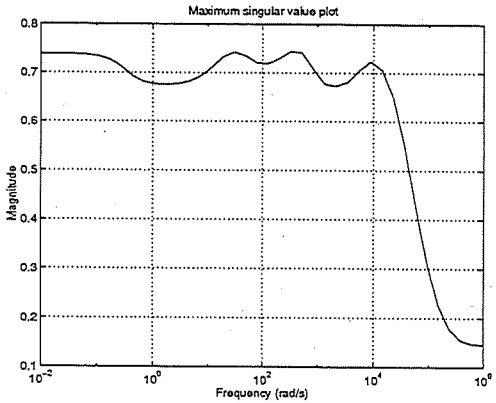


Figure 7.21 The maximum singular value plot of $G_{\delta\delta}$ with the H_∞ controller K_∞.

is found with the maximum of μ of the closed loop system less than 1. The μ plot is shown in Fig. 7.22. The maximum upper bound of μ is 0.96618, that is $\|G_c\|_D < 1$, so both stability and performance robustness can guaranteed for the closed loop system with such a controller when the uncertainties are LTI. However, with such a high order, the controller can not be implemented in practice. A controller order reduction is definitely needed. Here we use a balanced realization model reduction method [57]. This will result in an iterative procedure. First, reduce the controller order to first order. If the μ test is satisfied, then stop, otherwise reduce the controller to second order and so on until the μ test is satisfied. By such an order reduction procedure, a sixth order controller, denoted as K_μ is obtained (The controllers of lower order obtained by this way were not stabilizing). The μ plot of the closed loop system with the order reduced controller is shown in Fig. 7.23. The maximum upper bound of μ is 0.979. Controller K_μ has the following state space realization:

$$K_\mu = \left[\begin{array}{c|c} A_k & B_k \\ \hline C_k & D_k \end{array}\right],$$

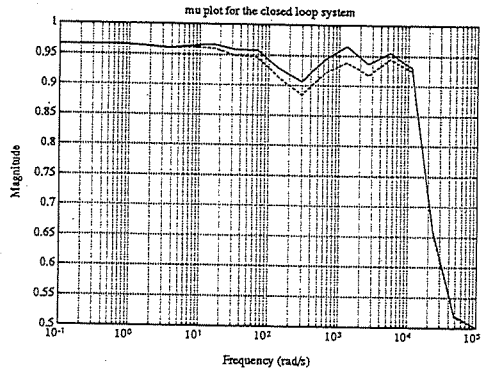


Figure 7.22 μ plot of the closed loop system with the controller from D-K iteration. The solid curve is for the upper bound and the dashed curve is for the lower bound.

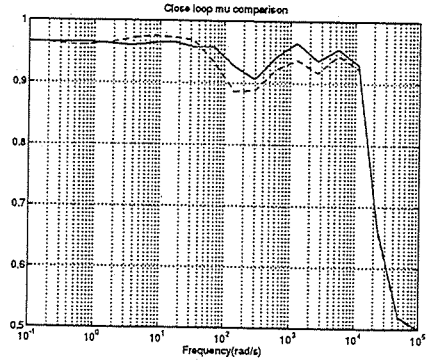


Figure 7.23 Comparison of μ upper bound plots of the closed loop system G_c with the 39th order controller (solid line) from D-K iteration and its reduced (6th) order controller K_μ with a balanced reduction (dotted line).

where

$$A_k = \begin{bmatrix} -1.1370e+02 & 8.8901e+04 & -1.3099e+03 \\ -8.8901e+04 & -5.9718e+05 & 6.3717e+04 \\ 1.3099e+03 & 6.3717e+04 & -1.0174e+04 \\ -1.7399e+01 & -9.7887e+02 & 1.7180e+02 \\ 5.5339e+02 & 4.9845e+04 & -1.5626e+04 \\ -4.7652e+01 & -3.7521e+03 & 9.3092e+02 \end{bmatrix}$$

$$\begin{bmatrix} 1.7399e+01 & 5.5339e+02 & 4.7652e+01 \\ -9.7887e+02 & -4.9845e+04 & -3.7521e+03 \\ 1.7180e+02 & 1.5626e+04 & 9.3092e+02 \\ -2.9898e+00 & -3.7061e+02 & -1.8840e+01 \\ 3.7061e+02 & -1.3354e+04 & -2.3508e+03 \\ -1.8840e+01 & 2.3508e+03 & -1.1301e+03 \end{bmatrix},$$

$$B_k = \begin{bmatrix} 1.4412e+02 \\ 9.5217e+03 \\ -6.4463e+02 \\ 9.2864e+00 \\ -3.7048e+02 \\ 3.0070e+01 \end{bmatrix}, C_k^T = \begin{bmatrix} 1.4412e+02 \\ -9.5217e+03 \\ 6.4463e+02 \\ -9.2864e+00 \\ -3.7048e+02 \\ -3.0070e+01 \end{bmatrix}, D_k = 0.$$

The frequency response of K_μ is shown in Fig. 7.24.

However, the design problem is not completely solved yet since we used here standard frequency dependent D-scaling, the condition of $\|G_c\|_D < 1$ can not ensure stability/performance robustness for nonlinear uncertainties. So stability and performance robustness has yet to be analyzed as before, but with mixed D-scaling now.

Performance robustness

Now we use the mixed D-scaling. The algorithm in section 6.2 for mixed D-scaling gives:

$$\|G_c\|_D = 0.9896,$$

where D is of order 2, 4, 4, 4 and 4 for the first 5 blocks respectively corresponding to the dependent scaling part, while the last four blocks of D is just constant parameters corresponding to the constant scaling part. Since fortunately $\|G_c\|_D < 1$ with the mixed D, both the stability robustness and the performance robustness are ensured.

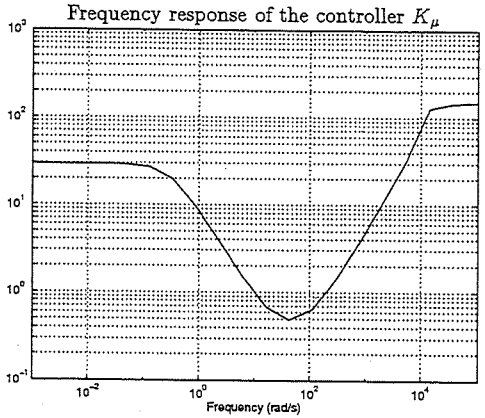


Figure 7.24 The frequency response for the controller K_μ.

7.3.4 μ control synthesis: D-K iteration with mixed D-scaling

The robust control design via D-K iteration with mixed D-scaling has been already discussed in Chapter 6 as an example. In doing so, D is of order 2, 3, 3, 3 and 2, and thus the controller is of order $7+2\times(2+3+3+3+2) = 33$, which is certainly too high for a practical implementation. The same order reduction procedure as used in the standard D-K iteration above but with mixed D-scaling is implemented to reduce the order of controller. The final controller, denoted by K_{mdk}, is of order 7. The controller K_{mdk} has the following realization:

$$K_\mu = \left[\begin{array}{c|c} A_k & B_k \\ \hline C_k & D_k \end{array}\right],$$

where

$$A_k = \begin{bmatrix} -9.7442e+05 & -1.8823e+05 & 9.7431e+03 \\ 0 & -2.1127e+04 & 7.0556e+03 \\ 0 & 0 & -5.1675e+03 \\ 0 & 0 & 1.2613e+04 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 9.3758e+04 & -3.1967e+04 & -4.4969e+04 & -8.2754e+02 \\ -1.9712e+03 & 2.2836e+03 & -9.7954e+02 & -1.3721e+02 \\ -1.2091e+04 & -1.1246e+03 & -5.9358e+02 & 3.2496e+01 \\ -5.4411e+03 & 2.3831e+03 & 2.4182e+03 & 1.5306e+01 \\ 0 & -3.0721e+03 & 1.1677e+02 & -5.6035e+00 \\ 0 & -5.2442e+03 & -2.1941e+03 & -1.5069e+01 \\ 0 & 0 & 0 & -3.0318e-01 \end{bmatrix},$$

$$B_k = \begin{bmatrix} 9.8999e+03 \\ 6.9803e+02 \\ -3.1442e+02 \\ -8.6807e+02 \\ 1.0364e+02 \\ 1.7689e+02 \\ 3.0724e+00 \end{bmatrix}, C_k^T = \begin{bmatrix} -9.8842e+03 \\ -1.0780e+03 \\ 5.7924e+02 \\ 3.8000e+02 \\ -1.2160e+02 \\ -1.8857e+02 \\ -3.8125e+00 \end{bmatrix},$$

$$D_k = -5.9749e-02.$$

The frequency response of K_{mdk} is shown in Fig. 7.25.

The singular value plot of the scaled closed loop system with controller K_{mdk} is shown in Fig. 7.26, and

$$\|G_c\|_D = 0.9802.$$

Since $\|G_c\|_D < 1$ has been already satisfied with the mixed D-scaling, both stability robustness and the performance robustness are guaranteed now.

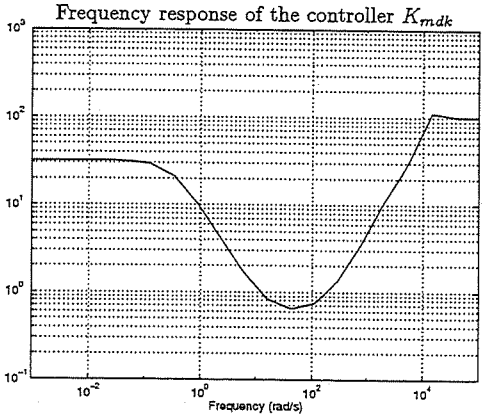


Figure 7.25 The frequency response for the controller K_{mdk}.

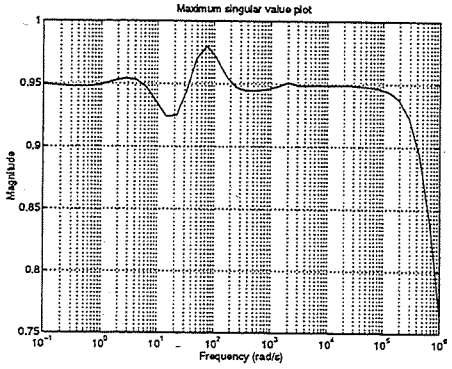


Figure 7.26 Maximum singular value plot of the scaled closed loop system with controller K_{mdk} designed via the mixed D-K iteration followed by the order reduction.

7.3.5 PID control design with parameter space search

PID controllers are frequently used in industry because of their simple structure and operator's intuitive understanding of them. The natural question is whether PID control parameters (K_p, K_i and K_d) can be tuned so that stability and performance robustness can be satisfied. The problem can be posed as follows:

$$\min_{K_p,K_i,K_d\in\mathbf{R},\ K_{pid}\text{ stabilizing}} \|\mathcal{T}(G, K_{pid}(K_p, K_i, K_d))\|_D.$$

This is not a convex minimization problem, so there is no attractive way to solve it. Here we search locally in the parameter space to find suitable parameters as follows. First, (1): take the initial point in the parameter space with the PID parameter values tuned based on the gain and phase margin method discussed in section 7.3.1. Then (2): grid the parameter space around the initial point in, for example, $5 \times 5 \times 5$ points, (3): calculate $\|G_c\|_D$. (For the sake of simplicity, only frequency D-scaling will be used. The stability and performance robustness will be analyzed later on). (4): If $\|G_c\|_D < 1$, then stop. Otherwise (5) regrid the parameter space with smaller intervals between points around the point with smallest $\|G_c\|_D$ so far, and then go to step (3). In this way we don't need a search over all parameter space, but again, there is no guarantee to find the minimum. However, this method works in our application.

From section 7.3.1, with the PID controller K_{pid}, $\|G_c\|_D$ reaches value 1.4643, which is not so far from 1. To search for better PID parameters with the search method which might reduce $\|G_c\|_D$, we put a grid in the 3-parameter space (K_p, K_i, K_d) with the point of the auto-tuned parameters in (7.11) in the center of the grid. We used $5\times5\times5$ points with a logspace interval 0.5. The minimum of the peak values of μ upper bound on these points is 1.0826 at the point:

$$\begin{aligned} K_p &= -7.7143e-01, \\ K_i &= -1.8139e+01, \\ K_d &= -2.5976e-03. \end{aligned} \tag{7.12}$$

Now we do the same thing with a new grid with a center point of the values given by (7.12) and a smaller interval 0.12. The minimum of the

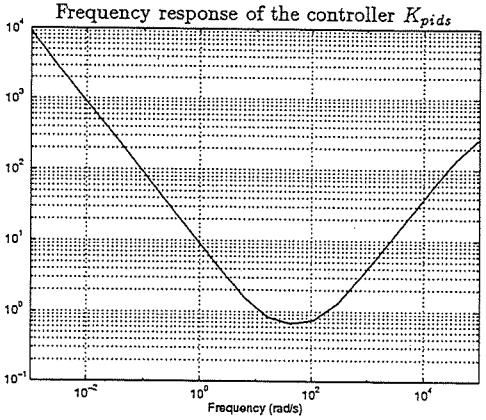


Figure 7.27 The frequency response for the controller K_{pids}.

peak values of μ on these new points is 0.9751 at the point:

$$K_p = -6.6808e-01,$$
$$K_i = -9.0693e+00,$$
$$K_d = -3.8964e-03.$$

The PID controller with these parameters is denoted by K_{pids} and the frequency response of K_{pids} is shown in Fig. 7.27.

Since in searching PID parameters, only the frequency D-scaling has been used, instead of the mixed D-scaling, for fast calculation, we need to further analyze the stability and performance robustness.

Performance robustness

With the mixed D-scaling, the algorithm in Chapter 6 gives

$$\|G_c\|_D = 0.9956,$$

where D is of order 2, 3, 3, 3 and 3 for the first 5 blocks. The maximum singular value plot of the scaled closed loop system DG_cD^{-1} is shown

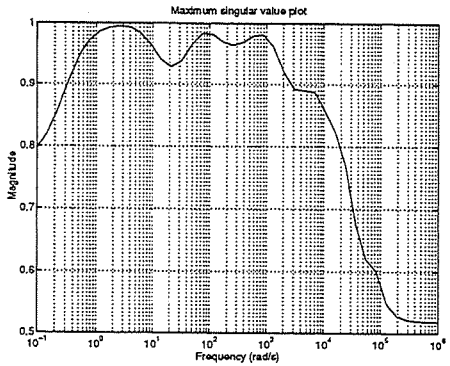


Figure 7.28 Maximum singular value plot of the scaled closed loop system DG_cD^{-1} with the PID controller K_{pids} with $K_d = -3.8964e-3$, $K_p = -6.6808e-01$, $K_i = -9.0693e+00$.

in Fig. 7.28. Obviously, the performance robustness is satisfied since $\|G_c\|_D < 1$ when the mixed D-scaling is used.

Stability robustness

However $\|G_c\|_D$ is almost equal to 1. If the stability robustness is also reached marginally like this, it might cause instability in practice. Thus we now check the stability robustness. The scaled H_∞ norm of $G_{\delta\delta}$ with the mixed D-scaling is $\|G_{\delta\delta}\|_D = 0.8165$ which is certainly less than 1. So the robust stability is indeed ensured. Fig. 7.29 shows the maximum singular value plot of the scaled $G_{\delta\delta}$.

7.4 Time and frequency responses

In this section we give some results in time and frequency response simulations.

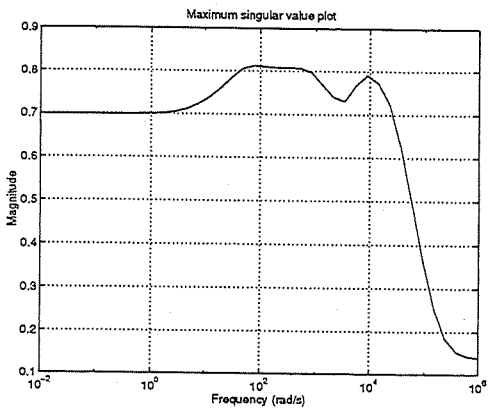


Figure 7.29 Maximum singular value plot of the scaled $G_{\delta\delta}$ with the PID controller K_{pids} with $K_d = -3.8964e-3$, $K_p = -6.6808e-01$, $K_i = -9.0693e+00$.

7.4.1 Time responses

A direct method to verify control designs is to simulate the time responses of closed loop systems. Simulations on the original nonlinear hydraulic system are made for different controllers. The pump flow is set to $9.57 \times 10^5 (\mathrm{ml}^3/\mathrm{s})$, corresponding to engine speed at 3000 rpm. The secondary cylinder volume is 280 ml. The clutch pressure is 8 bar, and finally the temperature is at $70°C$. The input is a step signal from 40 bar to 20 bar. Fig. 7.30 shows the simulation results of the closed loop systems with different controllers. It can be seen that the two best time responses are those with the controllers K_μ from the standard D-K iteration and K_{mdk} from the mixed D-K iteration (both of them are followed by the order reduction). The rise times for both responses are less then 0.033 second that is the time constant for the inverse of the weighting function. The response with the PID controller K_{pids} is also fairy good and certainly acceptable in practice considering the simple control structure. The response with the PID controller K_{pid} is slower. The response

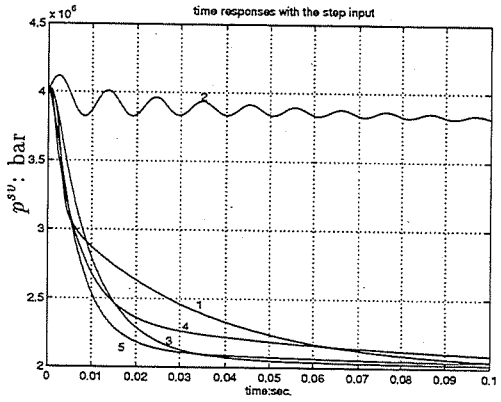


Figure 7.30 Time responses of the closed loop system with K_{pid} (curve 1), K_∞ (curve 2), K_μ (curve 3) and K_{pids} (curve 4) and K_{mdk} (curve 5).

with the H_∞ controller is completely unacceptable.

7.4.2 Frequency responses

Frequency response analysis is basically for LTI systems. We can only analyze for the nominal closed loop system: The closed loop system with all uncertainties being deleted.

The frequency responses for the nominal closed loop systems with different controllers are shown in Fig. 7.31. Nominal performance is not satisfied for the H_∞ controller K_∞ as the frequency response of the closed loop system with K_∞ is not below the specified curve $(W_p^{-1}(j\omega))$. The frequency responses with other controllers are all below the specified curve. Typically the frequency responses with K_μ and K_{mdk} are nicely shaped below the specified curve.

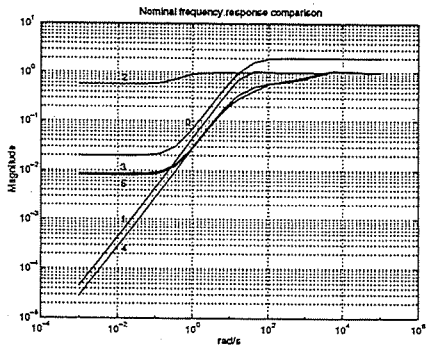


Figure 7.31 Frequency responses of the closed loop systems with K_{pid} (curve 1), K_∞ (curve 2), K_μ (curve 3) and K_{pids} (curve 4), K_{mdk} (curve 5) and the inverse of the weighting function (curve 0).

7.5 Conclusions

In this chapter, we applied robust control analysis and design methods developed in the previous chapters for control design and analysis for the hydraulic servo system of a CVT.

Firstly, we used the uncertainty modeling techniques developed in Chapter 4 and 5 to separate out uncertainties with nominal plant and to normalize all uncertainty block to be non-expansive.

Secondly several methods were used for design and analysis of robust control systems. All designs are now summarized in Table 7.5

The results show that the classic gain and phase margin test and the corresponding PID control design can not guarantee the robust performance, while H_∞ control design fails to get an useful controller. Both the standard D-K iteration and the D-K iteration with the mixed D-scaling successfully give controllers which ensure both the stability and performance robustness, and in doing so, an order reduction procedure has been introduced. By locally searching in the PID parameter space, a PID controller was found such that closed-loop robustness is guaranteed.

Table 7.5 Summary of different control designs.

Controller	Order	Rob.Stab.	Rob. Perf.	Time resp.	Freq. resp.
K_{pid}	2	No	No	Bad	Good
K_∞	7	Yes	No	Bad	Bad
K_μ	6	Yes	Yes	Good	Good
K_{mdk}	5	Yes	Yes	Good	Good
K_{pids}	2	Yes	Yes	Good	Good

Actually it is the best controller for our practical application because of its simple structure.

Finally both the time responses and the frequency responses are calculated. The time responses are simulated on the original nonlinear closed loop systems, while the frequency responses are calculated on the nominal closed loop systems where all uncertainties are omitted. Both time responses and frequency responses confirm the results from robustness analysis.

One remaining question is about the model errors from the nonlinearity. In the case of G_{r3}^{sv}, the parameters g_x and g_p are time varying and bounded, but their ranges are very large. All assumptions made on these ranges are rather tricky. Verifications for the control designs based on these assumptions have been made by simulation only. Another problem is that in modeling and control design, we omitted all the dynamics of electrical hydraulic actuator (PWM valve). Measurements show that in low temperature ($< 0°C$), the dynamics of PWM valve can not be neglected. Modeling these dynamics and control design for the hydraulic system with the actuator included is certainly necessary for lower temperature.

Chapter 8

Conclusions and open questions

In this dissertation, we have treated robustness analysis, uncertainty modeling and controller design for systems with structured uncertainties.

For the **robustness analysis**, we have focused on the μ calculations. The complex μ and the real parametric μ have been treated in two separate chapters, Chapter 2 and Chapter 3. The μ problem, which is cast as an optimization problem, can be tackled either by solving the related *optimality conditions* in an iterative way or by applying *general optimization algorithms* using function evaluation and gradient information. We have used both methods: the continuous power algorithm for the complex μ and the power iteration algorithm and its continuous time version for the STLS are based on the *optimality conditions*, while the steepest ascent and conjugate gradient algorithms for the complex μ are based on the *general optimization* method.

One open question for the μ calculation is the mixed or the real μ calculation. Until now there exists no algorithm with guaranteed convergent properties to solve the problem. One possible further research work is to generalize our gradient based algorithms for the complex μ to the mixed cases by using penalty function on imaginary part of the eigenvalue of the spectra radius, the research work is already under investigation.

For the **structured uncertainty modeling**, we have focused on the parametric uncertainty separation and the structured uncertainty normalization. This work is practically oriented and fills the gap between

system modeling and robust control theory. The parametric uncertainty separation has been tackled by the multidimensional realization method in Chapter 4. A corresponding algorithm has been presented to solve the problem systematically. The structured uncertainty normalization problem has been solved based on the sector transformation and loop transformation techniques in Chapter 5. A more practical and block oriented method has also been discussed in Chapter 7.

One open question for the parametric uncertainty separation based on the multidimensional realization method is that there is no guarantee for minimal realization, though we have proposed two size reduction methods which can only find a local minimum.

For the **controller synthesis**, in Chapter 6, a D-K iteration procedure with mixed frequency and constant D-scalings has been proposed in Chapter 6. This procedure can be used to synthesize robust control for systems with both LTI and NLTV uncertainties. An industrially motivated example shows that the result with this D-K iteration is not so conservative as with the D-K iteration with constant D-scalings.

An open question for mixed D-scalings is that the global minimum is not guaranteed. The problem here is that the D-scaling problem is convex only over the frequency responses of D, that is,

$$\min_{D \in \mathbf{D}} \|D(j\omega)G_c(j\omega)D(j\omega)^{-1}\|_\infty$$

is a convex problem for a certain ω. It is however not a convex problem over the state space matrices of the scaling D except when D is constant (constant D-scalings). Thus we can not combine both frequency dependent scaling and constant scaling in a same framework. This problem can be extended further to the whole D-K iteration. That is to find an optimal mixed D-scaling and an optimal controller such that the H_∞ norm of the scaled closed loop system is minimized globally.

For the **application**, we have applied the robust control analysis and design methods developed in the previous chapters for control design and analysis for an hydraulic servo system of a CVT in Chapter 7.

First, we have used the uncertainty modeling techniques developed in Chapter 4 and 5 to separate out uncertainties from the nominal plant and to normalize all uncertainty blocks to be non-expansive.

Second, several methods have been used for design and analysis of robust control systems. Five controllers have been designed: the classical

PID auto-tuning controller, the H_∞ controller, the controller from the D-K iteration with frequency dependent scaling, the controller from D-K iteration with the mixed D-scalings and the robust PID controller with parameter search. For both robust stability robust performance analysis, we used the mixed D scaling method developed in Chapter 6.

Finally all the results by the different design methods have been compared both in the frequency domain and in the time domain.

One practical related open problem is that taking all nonlinearities as uncertainties will certainly loose information concerning these nonlinearities, especially when the nonlinearity relations from physical laws are exactly known, and consequently the results will conservative. It might help to use the LFT gain scheduling method proposed by Packard in [62]. The method is actually that by augmenting uncertainties, through the input/output measurements of the uncertainties, into a controller the optimization can be done over a larger variable set. However this method does not yet exploit the information of the nonlinearities directly, and besides, measurements of the uncertainties might not be available in practice. Another possible way worth to investigating is to combine the feedback linearization method [44] and the robust control method together. Some nonlinearities can be canceled completely or partially by a feedback nonlinear controller designed first, then the robust control would be applied to the compensated plant with the remaining uncertainties and nonlinearities which are not canceled such that the conservativeness can be reduced eventually. Of course the problem can be very complicated since the feedback linearization itself is rather difficult problem comparing to a linear control problem, especially from the practical point of view.

Appendix A

Convex and LMI optimization

A general setup for an optimization problem can be expressed as

$$\min_{x \in \mathbf{S}} f(x),$$

where $\mathbf{S}$ is a set, $f(x) : \mathbf{S} \to \mathbb{R}$ is an object function. When both the function $f(x)$ and the constraint set $\mathbf{S}$ are convex, the optimization problem is called a convex optimization problem, otherwise, it is a non-convex optimization problem. For a convex optimization problem, any local minimum, if it exists, is a global minimum. While for a non-convex optimization problem, a local minimum might not be a global minimum. Similar to the watershed between linearity and nonlinearity in system theory, the great watershed in optimization theory is between convexity and non-convexity: the former is fundamentally more tractable than the later. There are already some efficient algorithms to solve convex optimization problems [59, 82], but there is no any efficient algorithm to solve non-convex optimization problems, some of them are even NP complete, for which one can only get local minima in general. Gradient based optimization is one of the methods to tackle non-convex problems, which we have already discussed in Chapter 2.

Many problems in control system analysis and synthesis, especially for robust control, can eventually be cast as optimization problems. In [11], Boyd and Barratt show that quite some specifications in control

are convex. More recently, Boyd *et al*, in [13], show that a variety of problems arising in system and control theory can be reduced to a few standard convex or quasi-convex optimization problems involving linear matrix inequalities (LMI). In [77, 37], H_∞ control problems are cast as convex or quasi-convex optimization involving LMI either, based on the Bounded Real Lemma. Here we give a brief overview on convex and LMI optimization.

A set $\mathbf{S}_c$ is convex if for all $\lambda \in [0, 1]$,

$$x_1 \in \mathbf{S}_c, x_2 \in \mathbf{S}_c \to \lambda x_1 + (1 - \lambda) x_2 \in \mathbf{S}_c.$$

That is: the line segment between any two points lies in $\mathbf{S}_c$. The epigraph of a function is defined as:

$$\mathbf{epi}\ f \triangleq \{(x, t) \mid f(x) \leq t\}.$$

A function $f(x) : \mathbf{S} \to \mathbb{R}$ is convex if its epigraph $\mathbf{epi}\, f$ is a convex set, or equivalently if the function $f(x)$ is convex if and only if

$$f(\lambda x + (1 - \lambda) y) \leq \lambda f(x) + (1 - \lambda) f(y)$$

for all $\lambda \in [0, 1]$ with $x, y \in \mathbf{S}$ and $f(x) > -\infty$ and $f(y) > -\infty$. A main property of convex functions is that any local minimum is a global minimum. A function is quasi-convex if its sub-level sets

$$\mathbf{S}_\alpha \triangleq \{x | f(x) \leq \alpha\}$$

are convex sets. Obviously if $f(x)$ is convex then it is quasi-convex. Again, for quasi-convex functions, any local minimum is a global minimum.

The optimization problem:

$$\min_{x \in \mathbf{S}_c} f(x) \tag{A.1}$$

with $f(x)$ a convex function and $\mathbf{S}_c$ a convex set is called a convex optimization problem.

Among the convex optimization problems, a specific problem is the Linear Matrix Inequality (LMI) optimization problem [13]. A standard LMI optimization problem (or Positive Definite Programming, PDP) is

defined as a convex optimization problem as in (A.1) with $f(x) = c^T x$ and S_c determined by a linear matrix inequality (LMI):

$$F_0 + x_1 F_1 + \cdots + x_m F_m \geq 0, \tag{A.2}$$

where $x \in \mathbb{R}^m$ is a variable vector, both $c \in \mathbb{R}^m$ (a given fixed vector) and $F_i = F_i^T \in \mathbb{R}^{n \times n}$ (given fixed matrices) contain the problem data. This LMI optimization problem is also called an eigenvalue optimization problem since it is equivalent to the problem:

$$\min_{\substack{\lambda I - A(x) > 0 \\ B(x) > 0}} \lambda, \tag{A.3}$$

where $A(x)$ and $B(x)$ are symmetric matrices that depend affinely on the optimization variable x. The extension to the LMI optimization problem, or eigenvalue optimization problem is the so called generalized eigenvalue optimization problem:

$$\min_{\substack{\lambda B(x) - A(x) > 0 \\ B(x) > 0 \\ C(x) > 0}} \lambda, \tag{A.4}$$

where $A(x), B(x)$ and $C(x)$ are symmetric matrices that are affine functions of x. The optimization problem is also equivalent to:

$$\min_{\substack{B(x) > 0 \\ C(x) > 0}} \lambda_{\max}(A(x), B(x)),$$

where $\lambda_{\max}(X, Y)$ denotes the largest generalized eigenvalue of the pencil $\lambda Y - X$. This general eigenvalue problem is however a quasi-convex optimization problem since the objective function is quasi-convex though the constrains are convex.

There are a number of general algorithms for the solutions of LMI optimization problems. Among them are the Ellipsoid algorithms and the interior point algorithms. The Ellipsoid algorithm is simple and, from a theoretical point of view, efficient. In practice, however, the interior-point algorithms are much more efficient [59, 82, 12].

We are not going to further investigate convex or LMI optimization problems in the rest of this dissertation. Instead, once a control problem

is cast as a convex or LMI optimization problem, we consider the control problem as solved since there do exists efficient algorithms to solve it. Actually there are already MATLAB toolboxes available to solve convex or LMI optimization problems arising from robust control (see LMIlab [38]).

Appendix B

Super Block interconnection for the CVT hydraulic servo system

In this appendix, we show the models of the CVT hydraulic servo system in the MATRIX$_x$ Super Block format. They includes the secondary circuit, secondary valve, secondary cylinder Supper Block interconnections. The Super Block interconnection for simulation setup is also included here. But first we give the mathematic models for the hydraulic and mechanical elements represented by Super Blocks.

Input/output relation for an hydraulic resistor

$$q = \min(0.611;\ \sqrt{\frac{0.611^3 D_h}{20\nu(T)}\sqrt{\frac{2p}{\rho(T)}}})a_r\sqrt{\frac{2p}{\rho(T)}}, \tag{B.1}$$

where

input p: the oil pressure across the resistor,
output q: the oil flow through the resistor,
D_h: the characteristic length
(depends on the shape of the opening),
a_r: the cross section area of the channel,
$\rho(T)$: the oil density as a function of temperature T,
$\nu(T)$: the oil viscosity as a function of temperature T.

This input/output relation is used for G_{r1}^{sv}, G_{r2}^{sv}, G_{r3}^{sv} and G_r^{sc}.

Input/output relation for an hydraulic capacitor

$$p = \frac{1}{C}\int q dt + p_0, \tag{B.2}$$

where

input q: the oil flow into the capacitor,
output p: the oil pressure in the capacitor,
$C = \frac{V}{B}$: the compliance,
V: the oil volume which can be variable,
B: the bulk modulus of the oil (4.10^8Pa),
p_0: the initial value of the oil pressure.

This input/output relation is used for C_{c1}^{sv}, C_{c2}^{sv} and C_c^{sc}.

Input/output relation for a translational object

$$v = \frac{1}{I}\int F dt + v_0, \tag{B.3}$$

where

input F: the forces acting on the object,
output v: the velocity of the object,
$I = m$: the inertia,
m: the mass of the object.

This input/output relation is used for I_m^{sv}.

Input/output relation for a viscous friction resistor

$$f = rv, \tag{B.4}$$

where

input v: the difference between velocities of the friction surfaces,
output f: friction force,
r: the friction coefficient.

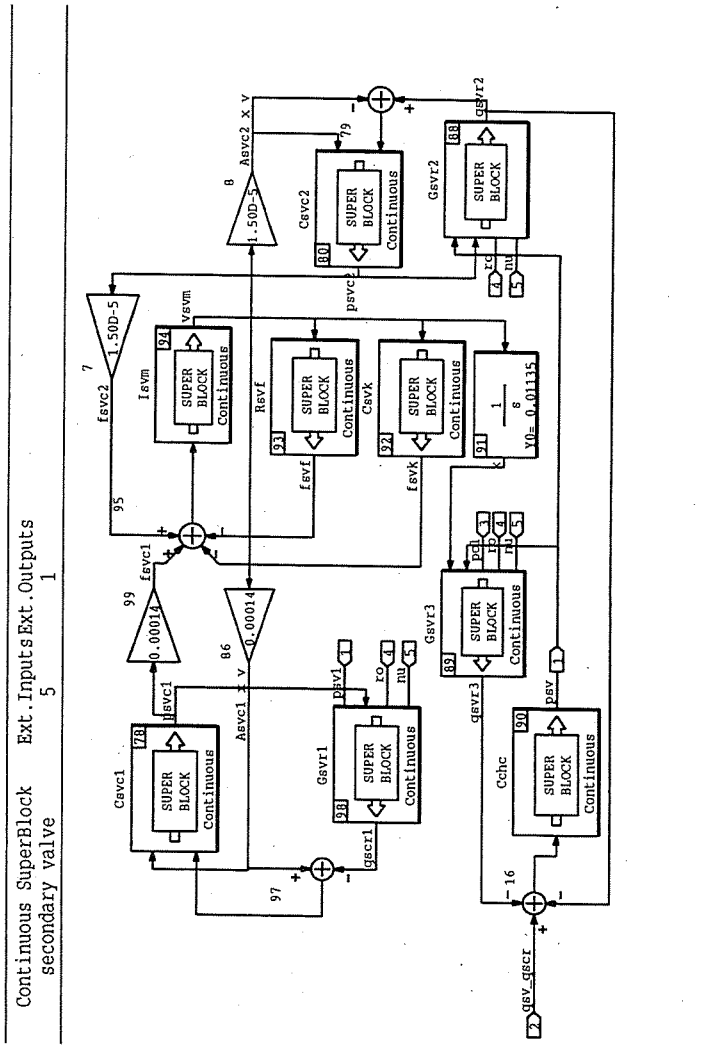
This input/output relation is used for R_f^{sv}.

Input/output relation for a spring

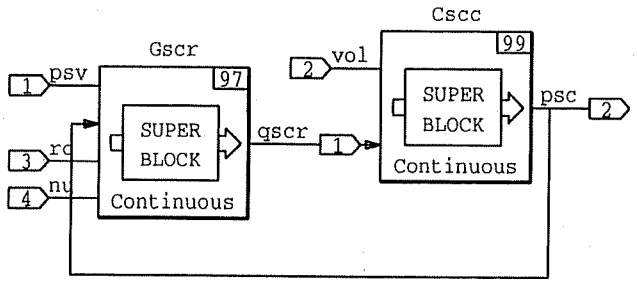
$$f = \frac{1}{C}\int v dt + f_0, \tag{B.5}$$

input v: the relative velocity on the sprint,
output f: the spring force,
$C = \frac{1}{k}$: the compliance,
k: the spring constant.

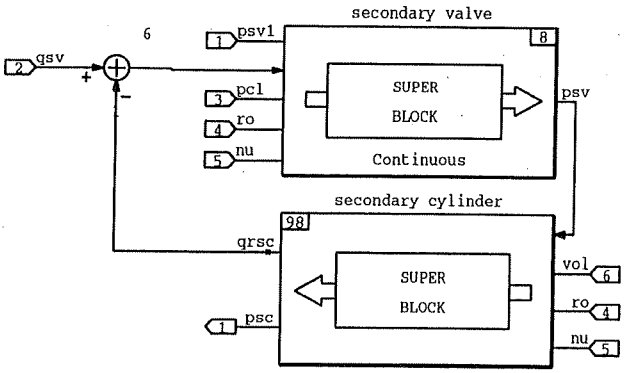
This input/output relation is used for C_k^{sv}.



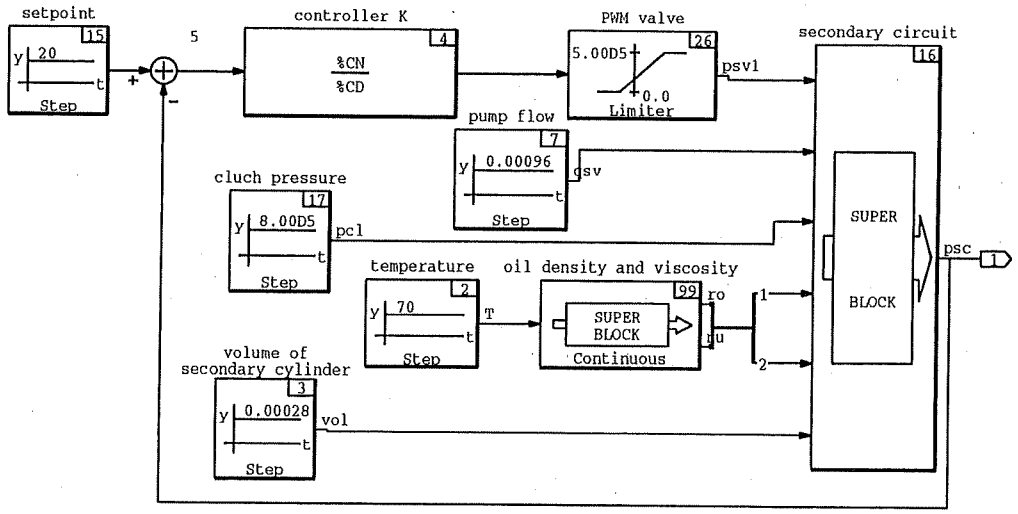
Continuous SuperBlock	Ext.Inputs	Ext.Outputs
secondary cylinder	4	2



Continuous SuperBlock	Ext.Inputs	Ext.Outputs
secondary circuit	6	1



Continuous SuperBlock	Ext.Inputs	Ext.Outputs
CVT Super Block for simulation	0	1



Bibliography

[1] P. Apkarian, J.P Chretien, P. Gahinet and J.M. Biannic, "μ synthesis by D-K iterations with constant scaling", *Proc. 1993 Amer.Contr.*, San Francisco, pp. 3192-3196, June 1993.

[2] K.J. Åström, etc, "Automatic tuning of simple regulators with specifications on phase and amplitude margins", *Automatica.* Vol 30, pp. 645-651, 1984.

[3] K.J. Åström, Bjöm Wittenmark, *Adaptive control*, Addison-Wesley, 1989.

[4] G.J. Balas, J.C. Doyle, K. Glover, A. Packard and R. Smith, *μ-Analysis and Synthesis Toolbox*, The Math Works Inc., 1993.

[5] B. Bamieh and M. Dahleh, "On robust stability with structured time-invariant perturbations", *System and Control Letter*, vol. 21, 1993, pp. 103-108.

[6] V. Balakrishnan, S. Boyd, and S. Balemi, "Branch and bound algorithm for computing the minimum stability degree of parameter dependent linear systems," *Int. J. of Robust and Nonlinear Control*, vol. 1, pp. 295-317, Oct.-Dec. 1991.

[7] B.R. Barmish, "Invariance of the strict Hurwitz property for polynomials with perturbed coefficients", *IEEE Trans. on Automatic Control*, AC-29, 935-936, 1984.

[8] B.R. Barmish and H.I. Kang, "A survey of extreme point results for robustness of control systems", *Automatica*, (Special issue on robust control), Vol. 29, No.1, pp. 13-35, 1993.

[9] N. K. Bose, "Problems and progress in multidimensional system theory." *Proceedings of the IEEE*, Vol.65, No. 6, 1977, pp.824-840.

[10] Stephen Boyd, Qinping Yang, "Structured and simultaneous Lyapunov functions for system stability problem", *Int. J. Contrl*, vol. 49, No. 6, pp. 2215-2240.

[11] S. Boyd and S. Barratt, *Linear Control Design: Limits of Performance*, Prentice-Hall, Englewood Cliffs, 1991.

[12] S. Boyd and L. EL Ghaoui,"Method of centers for minimizing generalized eigenvalues", *Linear Algebra and Applications, special issue on Numerical Linear Algebra Methods in Control, Signals and Systems*, vol. 188, pp.63-111, July, 1993.

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

[14] R. D.Braatz, P. M. Young, J. C. Doyle and M. Morari, "Computational complexity of μ calculation", *Proc. ACC93*, San Francisco, pp. 1682-1683, 1993.

[15] Moody T. Chu, "On the continuous realization of iterative processes", *SIAM Review*, Vol. 30, No. 3, pp. 375-387, 1998.

[16] J.B. Conway, *Functions of one complex variable*, Springer-Verlag, N.Y., 1978.

[17] J. David, *Algorithms for Analysis and Design of Robust Controllers*, Ph.D. dissertation, Katholieke Universiteit Leuven, Belgium, 1994.

[18] M. Dahleh and J.C. Doyle, "Overview of robust stability and performance methods of systems with mixed perturbations",

Proc. 30th Conf. Decision Contr., Tucson, Arizone, Dec. 1992, pp. 3158-3162.

[19] M.A. Dahleh and M.H. Khammash, "Controller design for plants with structured uncertainty", *Automatica*, vol. 29, no. 1, pp. 37-56, 1993.

[20] J. Demmel, "The componentwise distance to the nearest singular matrix," *SIAM Journal on Matrix Analysis and Applications*, vol. 13, pp. 10-19, 1992.

[21] B. De Moor, "Structured total least squares and L_2 approximation problems", Special Issue of *Linear Algebra and its Applications* on Numerical Linear Algebra in Systems & Control, (eds : Van Dooren, Ammar, Nichols, Mehrmann), Vol. 188-189, pp. 163-207, July 1993.

[22] Bart. De Moor, "Riemanian SVD", in preparation, 1994.

[23] C.A. Desoer and M. Vidyasagar, *Feedback Systems: Input-Output Properties*, Academic Press, New York, 1975.

[24] J.C. Doyle, "Guaranteed margins for LQG regulators", *IEEE Trans. Auto. Control*, vol. AC-23, No. 4, pp. 45-56.

[25] J.C. Doyle and G. Stein, "Multivariable feedback design: Concepts for a classical/modern synthesis," *IEEE Trans. Auto. Contr.*, vol. AC-26, pp. 4-16, 1981.

[26] J.C. Doyle, "Analysis of feedback systems with structured uncertainties", *IEE Proc.* vol. 129, Pt.D, No.6, pp. 242-250, Nov., 1982.

[27] J.C. Doyle, "Synthesis of robust controllers and filters", IEEE Proc. 22nd Conf. Decision and Control, pp. 109-114, Dec. 1983.

[28] J.C. Doyle, K. Glover, P.P.Khargonekar, and E.A. Francis, "State space solutions to standard H_2 and H_∞ control problems", *IEEE Transactions on Automatic control*, Vol. AC-34, No.8, August 1989.

[29] J.C. Doyle, A. Packard and K. Zhou, "Review of LFTs, LMIs and μ", *Proc. 30th Conf. Decision Contr.*, Brighton, England, Dec. 1991, pp. 1227-1232.

[30] L. El Ghaoui, *Robustness of linear systems to parameter variations*, (Ph.D. dissertation, Stanford Univ.), 1990.

[31] L. El Ghaoui, "Fast computation of the largest stability radius for a two-parameter linear system", *IEEE Trans. Automat. Contr.*, vol. 37, No.7, pp. 1033-1037, 1992.

[32] L. EI Ghaoui, P. Gahinet, "Rank minimization under LMI constraints: a framework for output feedback problems", *Proc. 1993 European Control Conf.*, Groningen, pp. 1176-1179, 1993.

[33] M.K.H. Fan, and A. Tits, "Characterization and efficient computation of the structured singular value", *IEEE, Automat. Contr.*, Vol. 31, No. 8, pp. 734-743, 1986.

[34] M.K.H.Fan, Andre L. Tits and John C. Doyle, "Robustness in the presence of mixed parametric uncertainty and unmodelled dynamics", *IEEE Trans. Automat. Contr.*, vol. AC-36, No. 1, pp. 25-38, Jan. 1991.

[35] Fletcher R., *Practical Methods of Optimization*, Wiley & Sons, Chichester, 2nd ed. (repr.), 1993.

[36] B.A. Francis, *A Course in* H_∞ *Control*, Number 88 in Lecture Notes in Control and Information Sciences, Springer-Verlag, Berlin, 1987.

[37] Pascal Gahinet, "A convex parameterization of H_∞ suboptimal controllers", *31st IEEE Conference on Decision and control*, Tucson, pp. 937-942,1992.

[38] P. Gahinet and A. Nemirovskii, *LMIlab A Package for Manipulating and Solving LMI's*, 1993.

[39] G.H. Golub and C.F. Loan, *Matrix Computation*, Johns Hopkins University Press, Baltimore, MD, 1989.

[40] M. Green and D.J.N. Limebeer, *Linear Robust Control*, Prentice Hall, 1995.

[41] R. Hakvoort, *System Identification for Robust Control*, Ph.D. Dissertation, Delft University of Technology, The Netherlands, Nov. 1994.

[42] S. Helgason, *Differential Geometry, Lie Groups and Symmetric Spaces*. Academic Press, New York, 1978.

[43] Uwe Helmke, John B. Moore, *Optimization and Dynamical Systems*, New York: Springer-Verlag 1993.

[44] A. Isidori, *Nonlinear Control Systems: An introduction*, Spinger-Verlag, second edition, 1989.

[45] T. McKelvey and A. Huseyin, "An efficient frequency domain state-space identification algorithm", *proc. 33nd Conf. Decision and Control*, Lake Buena Vista, FL, Dec. 1994, pp. 3359-3364.

[46] T. Kailath, *Linear Systems*, (Prentice-Hall), 1980.

[47] D.C. Karnopp and R.C. Rosenberg, *System dynamics: a unified approach*. John Wiley & Sons, New York, USA, 1975.

[48] V.L. Kharitonov, "Asymptotic stability of an equilibrium position of a family of systems of linear differential equations", *Differential Equations*, vol. 14, pp. 1483-1485, 1979.

[49] T. Kato, *A Short Introduction to Perturbation Theory for Linear Operators*, Springer-Verlag, 1982.

[50] S.Y. Kung, B. Lévy, M. Morf, T. Kailath, 1977, " New results in 2-D systems theory, part II: 2-D state-space models - realization and the notations of controllability, observability, and minimality", *Proceedings of the IEEE*, Vol. 65, No.6, pp. 945-861, 1977.

[51] Paul Lambrechts, Jan Terlouw, Samir Bennani and Maarten Steinbuch,"Parametric uncertainty modeling using LFTs", *Delft University Press*, vol.5, pp. 1-10, Dec. 1992.

[52] John N. Little and Loren Shure, *Signal Processing Toolbox*, The Math Works Inc., 1992.

[53] L. Ljung, *System Identification: Theory for the User*, Prentice Hall, Englewood Cliffs, N.J., 1987.

[54] The MathWorks Inc., *MATLAB Reference Guide*, Natick, Massachusetts, 1992.

[55] *MATRIX$_X$ SYSTEM BUILD user's guide*, Integrated System Inc. 1990.

[56] W.Minten and P. Vanvuchelen, *Modeling and Simulation of An Electronically Controlled Variable Transmission*, Departement Electrotechniek, K.U. Leuven, Belgium, 1991, Master's Thesis (Bart De Moor, advisor), in Dutch.

[57] B.C. Moore, "Principal components analysis in linear systems: controllability, observability and model reduction", *IEEE Transaction on Automatic Control*, Vol. AC-26, pp 17-31, 1981.

[58] T. Nanda, "Differential equations and the QR algorithm", *SIAM J. Numer. Anal.*, Vol. 22, No. 2, pp. 310-321, 1985.

[59] Yu. Nesterov and A. Nemirovsky, *Interior Point Polynomial Methods in Convex Programming: Theory and Application*, SIAM, 1993.

[60] A. V. Oppenheim and R. W. Schaffer, Digital Signal Processing, Prentice-Hall, New Jersey, 1975.

[61] Andy Pachard, Michael K.H.Fan and John Doyle, "A power method for the structured singular value", *Proc. 27th Conf. Decision Contr.*, Austin, Texas, Dec. 1988, pp. 2132-2137.

[62] A.K. Packard and J.C. Doyle, "The complex structured singular value", *Automatica*, Special issue on robust control, Vol. 29, No.1, pp. 71-110, 1993.

[63] I.R. Peterson, B.D.O. Anderson and E.E. Jockheere, "A first principle solution to the non-singular H_∞ control problem", Int. J. of robust and nonlinear control, vol 1, 1991, pp. 171-185.

[64] L. Qiu and E.J. Davison, "A simple procedure for the exact stability robustness computation of polynomials with affine coefficient perturbations,", *System & control letters*, vol. 13, pp. 413-420, 1989.

[65] R.M. Redheffer, "Inequalities for a matrix Riccati equation," *Journal of mathematics and Mechanics*, vol. 8, no. 3, 1959.

[66] M.G. Safonov and M. Athans, "Gain and phase margins for multiloop LQG regulators", *IEEE Trans. on Automat. Contr.*, Vol. 22, No. 4, pp. 173-179, 1977.

[67] M. G. Safonov, "Tight bounds on the response of multivariable systems with component uncertainty", *Allert Conference on Communication, Control and Computing*, pp. 451-460, 1978

[68] M. G. Safonov, *Stability and Robustness of Multivariable Feedback Systems*, Cambridge, MA: MIT Press, 1980.

[69] M.G. Safonov and M. Athens, " A multiloop generalization of the circle criterion for stability margin analysis", *IEEE Trans. Automatic Control*, vol. AC-26, no. 22, pp. 415-422, Apr. 1981.

[70] M.G. Safonov, E.A. Jonckheer, M. Verma and D.J.N. Limebeer, "Synthesis of positive real multivariable feedback systems", *Int. J. Control*, vol. 4, No. 3, 817-842, 1987.

[71] M.G. Safonov, "Propagation of conic model uncertainty in hierarchical systems", *IEEE Trans. on circuits and systems*, vol, cas-30, no. 6, pp. 388-396, June, 1983.

[72] M.G. Safonov, "Stability of interconnected systems having slope bounded nonlinearities", *6th Int. Conf. on Analysis and optimization of systems*, Nice, France, pp. 275-287, 1984.

[73] I.W. Sandberg, "On the L_2-boundedness of solutions of nonlinear functional equations", *Bell Sys. Tech. J.*, vol. 43, 1964, pp. 1581-1599.

[74] I.W. Sandberg, "A frequency-domain condition for stability of feedback systems containing a single time-varying nonlinear element", *Bell Sys. Tech. J.*, vol. 43, 1964, pp. 1601-1608.

[75] I.W. Sandberg, "Some results on the theory of physical systems governed by nonlinear functional equations", *Bell Sys. Tech. J.*, vol. 44, 1965, pp. 871-898.

[76] D.H. Sattinger and O.L. Weaver, "Lie Groups and Algebras with Applications to Physics, Geometry and Mechanics", Applied Mathematical Sciences, Vol.61, Springer-Verlag, New York, 1993.

[77] C. Scherer, *The Riccati Inequality and State-Space H_∞-optimal Control*, Ph.D Dissertation, Universitat Wurzburg, Germany, 1990.

[78] J.S. Shamma, "Robustness analysis for time-varying systems", *Proc. of the 31st Conf. on Decision and Control*, Tucson, Artzons, Dec. 1992, pp. 3163-3168.

[79] S.T. Smith, *Geometric Optimization Methods for Adaptive Filtering*, Ph.D. thesis, Harvard University, Cambridge, Massachusetts, May 1993.

[80] M. Steinbuch, J.C. Terlouw, O.H. Bosgra and S.G. Smit, "Uncertainty modeling and structured singular-value computation applied to an electromechanical system", *IEE Proceedings-D*, **139**(3), pp. 301-307, 1992.

[81] A. Tesi and A. Vicino, "Robust stability of state-space models with structured uncertainties", *IEEE Trans. Automat. Contr.*, Vol. 35, No. 2, pp. 191-195,1990.

[82] L. Vandenberghe and S. Boyd, "Primal-dual potential reduction method for problems involving matrix inequalities",

Mathematical Programming Series B, special issue on nondifferentiable and large scale optimization, 1993.

[83] P. Vanvuchelen and B. De Moor, "A mechatronics approach to electronic control of continuously variable transmissions : towards the limits of performance of combustion engine cars", Internal Report (ESAT-SISTA/TR 1993-36I), Departement Elektrotechniek, K.U.Leuven, Belgium, July, 1993.

[84] M. Vidyasagar, *Nonlinear Systems Analysis*, second edition, Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[85] B. Wei and D.S. Bernstein, "A benchmark problem for robust control design", *Proc. American Control Conference*, San Diego, May 1990, pp. 961-962.

[86] J.H. Wilkinson, *The algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1965.

[87] Yi Cheng and J. A. Peperstraete, "A microcomputer based self-tuning PID controller intended for distributed control system", *Proceedings of the ISMM International Symposium: Intelligent Distributed Processing*, Ft. Lauderdale, Florida, Dec. 13 - 15, 1989. pp. 107-110.

[88] Yi Cheng and J.A. Peperstraete, "Auto-tuning control systems", *Proceedings of the Ninth IASTED International Symposium: Modeling, Identification and Control*, Innsbruck, Austria, Feb. 10-12, 1992, pp.11-14.

[89] Yi Cheng, Bart De Moor, "Notes on VOLVO PID control algorithm", Internal Report (ESAT-SISTA/TR 92-52I), Departement Elektrotechniek, K.U.Leuven, Belgium, 1992.

[90] Yi Cheng, Bart De Moor, "A scheme for adapting the PWM frequency and PID controller to temperature changes", Internal Report (ESAT-SISTA/TR 1992-54I), Departement Elektrotechniek, K.U.Leuven, Belgium, 1992.

[91] Yi Cheng and J.A. Peperstraete, "An auto-tuned thermostat for an industrial food process control", *JOURNAL A*, Vol. 34, No.2, pp. 27-37, 1993.

[92] Yi Cheng and Bart De Moor, "A multidimensional realization algorithm for parametric uncertainty modeling problems", *Proc. 32th Conf. Decision Contr.*, San Antonio, Texas, U.S.A., 1993.

[93] Yi Cheng and Bart De Moor, "Continuous power iteration for computing lower bound of μ", *1994 American Control Conference*, Baltimore, Maryland, June 29 - July 1, 1994, pp. 1416-1417.

[94] Yi Cheng and Bart De Moor, "Robustness analysis and control system design for the hydraulic servo system of a continuously variable transmission", *IEEE Trans. Control Systems Technology*,Vol. 2, No. 3, September, 1994, pp. 183-197.

[95] Yi Cheng and Bart De Moor, "A multidimensional realization algorithm for parametric uncertainty modeling and multiparameter margin problems", *International Journal of Control*, Vol. 60, No. 5, pp. 789-807, 1994.

[96] Yi Cheng, Jeroen Dehaene and Bart De Moor, "Computation of the structured singular value with gradient based optimization algorithms", Internal Report (ESAT-SISTA/TR 1994-65I), Departement Elektrotechniek, K.U.Leuven, Belgium, September, 1994, accepted by *1995 American Control Conference*, also submitted to *IEEE Transaction on Automatic Control.*

[97] Yi Cheng and Bart De Moor, "D-K iteration with mixed D-scaling applied to an industrial hydraulic servo system", Internal Report (ESAT-SISTA/TR 1994-66I), Departement Elektrotechniek, K.U.Leuven, Belgium, September, 1994.

[98] Yi Cheng and Bart De Moor, "Uncertainty normalization with LFTs", Internal Report (ESAT-SISTA/TR 1995-20I), De-

partement Elektrotechniek, K.U.Leuven, Belgium, January, 1995

[99] Peter M. Young and John C. Doyle, "Computation of μ with real and complex uncertainties", *Proc. 29th Conf. Decision Contr.*, Honolulu, Hawaii, Dec. 1990, pp. 1230-1235.

[100] Peter M. Young, M. P. Newlin and John C. Doyle, "Practical computation of the mixed μ problem", *Proc. 1992 Amer. Contr.*, Chicago, Illinois, pp. 2190-2194, June 1992.

[101] G. Zames, "On the input-output stability of time-varying nonlinear feedback systems, part I: conditions derived using concepts of loop gain, conicity, and positivity", *IEEE Trans. Automatic Control*, vol. AC-11, No. 2, April, 1966.

[102] G. Zames, "On the input-output stability of time-varying nonlinear feedback systems, part II: conditions involving circles in the frequency plane and sector nonlinearities", *IEEE Trans. Automatic Control*, vol. AC-11, No. 3, July, 1966.

[103] G. Zames, "Feedback and optimal sensitivity: model reference transformations, multiplicative seminorms, and approximate inverses", *IEEE Trans. Auto. Control*, vol. AC-26, pp. 301-320.