CDS

TECHNICAL MEMORANDUM NO. CIT-CDS 93-010
June 17, 1993

"Robust Loopshaping for Process Control"

Dissertation by Richard Dean Braatz

Control and Dynamical Systems
California Institute of Technology
Pasadena, CA 91125

ROBUST LOOPSHAPING FOR PROCESS CONTROL

Dissertation by

Richard Dean Braatz

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Division of Chemistry & Chemical Engineering

Pasadena, California

1993

(Defended 18 May 1993)

Theorem: Nothing is truly complicated. The mark of true intelligence is the ability to see the simplicity of all real things.

My only hope is that this thesis makes a difference.

Acknowledgments

I am deeply indebted to my Ph.D. advisor Manfred Morari for his support and guidance, and for allowing me considerable freedom in conducting this research while simultaneously exacting his very high standards.

I am grateful to other people who have given me advice at various times in the last nine years: Octave Levenspiel and Keith Levien at Oregon State University while I was an undergraduate; Sigurd Skogestad at Norges Tekniske Høgskole in Trondheim, Norway during my stay there last summer, and Karl Åström at Lunds Tekniska Högskola in Lund, Sweden during his stay earlier this year at Caltech as a Fairchild Scholar.

Thanks to the people who I have worked with in chemical, mechanical, and electrical engineering over the last few years: Morten Hovd, Dan Laughlin, Jay Lee, Andy Packard, Matthew Tyler, and Peter Young. I thoroughly enjoyed interacting with Ferdinand Pranckh and Luigi Sartor at Avery/Dennison Research Center in Pasadena in developing the adhesive coating controller for Avery/Dennison Company which is described in this thesis. I am happy for the friends I have made with students in Manfred Morari, John Doyle, and Sigurd Skogestad's groups, and around the world: Frank Allgöwer, Venkataramanan (Ragu) Balakrishnan, Carolyn Beck, Nikolaus Bekiaris, Hector Budman, Bobby Bodenheimer, Peter Campo, Frank Doyle, Håkan Hjalmarsson, Tyler Holcomb, Iftikhar Huq, Elling Jacobsen, Mayuresh Kothare, Frank Laganier, Lionel Laroche, Petter Lundström, Knut Mathisen, George Meski, John Morris, Al Moser, Matt Newlin, Simone Oliveira, Ricardo Peña, Carl Rhodes, Cris Radu, Doug Raven, Tony Skjellum, Mario Sznaier, Athanasios Tsirukis, Cris Webb, Zhi (Alex) Zheng. Outside the sphere of control research, I will remember my compatriots on the Graduate Review Board and the Chemical Engineering Softball Team, and my roommate and cycling partner Phil Lovalenti.

I thank John Doyle, George Gavalas, and Richard Murray their roles on my thesis committee. I thank Karl Åström, John Doyle, Carl Nett, and Manfred for providing a wide range of role models in the control area.

Great teaching made my graduate courses a joy. I thank Octave Levenspiel for teaching me kinetics, Rudy Marcus for statistical mechanics, Francis Arnold and Jay Bailey for biochemical engineering, Henry Weinberg for thermodynamics, Gary Leal for transport phenomena, Howard Brenner for macro-transport phenomena, John Doyle and Thanasis Sideris for control theory, Steve Wiggins for nonlinear dynamical systems, and Tom Caughey for nonlinear stability analysis.

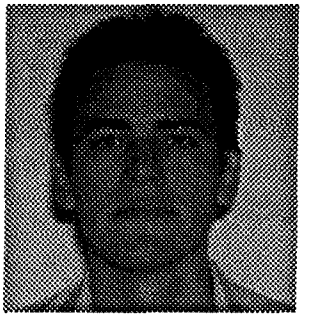
I gratefully acknowledge the various funding agencies which have supported my work: the Norwegian Marshall Fund, the Texaco Foundation, the Amoco Foundation, the American Scandinavian Foundation of Los Angeles, and the Hertz Foundation.

Lastly, I thank my biological parents, Richard Sr. and Loralee Ann, for being constant sources of moral support, and Mitsuko for helping me preserve my sanity during the long hours needed to complete this effort.

One last pre-doctoral toast to my friends:

"Mead does more than Milton can

to explain God's ways to man."



ROBUST LOOPSHAPING FOR PROCESS CONTROL

Richard Dean Braatz

Abstract

Strong trends in chemical engineering and plant operation have made the control of processes increasingly difficult and have driven the process industry's demand for improved control techniques. Improved control leads to savings in resources, smaller downtimes, improved safety, and reduced pollution.

Though the need for improved process control is clear, advanced control methodologies have had only limited acceptance and application in industrial practice. The reason for this gap between control theory and practice is that existing control methodologies do not adequately address *all* of the following control system requirements and problems *associated* with control design:

- The controller must be insensitive to plant/model mismatch, and perform well under unmeasured or poorly modeled disturbances.
- The controlled system must perform well under state or actuator constraints.
- The controlled system must be safe, reliable, and easy to maintain.
- Controllers are commonly required to be decentralized.
- Actuators and sensors must be selected before the controller can be designed.
- Inputs and outputs must be paired before the design of a decentralized controller.

A framework is presented to address these control requirements/problems in a general, unified manner. The approach will be demonstrated on adhesive coating processes and distillation columns.

Contents

List of Figures

List of Tables

Part I

Introduction

Chapter 1

Introduction

Issues in Chemical Process Control Leaders from industry, government, and academia meet every few years [21] to critique and assess the current status and future needs in the field of process control. At each of these meetings, participants reaffirm that advanced process control methodologies have had only limited acceptance and application in industrial practice. The reason for this gap between control theory and practice is that existing control methodologies do not adequately address *all* of the following control system requirements and problems *associated* with control design:

1. The controller must be insensitive to plant/model mismatch, and perform well under unmeasured or poorly modeled disturbances.

2. The controlled system must perform well under state or actuator constraints.

3. The controlled system must be safe, reliable, and easy to maintain.

4. Controllers are commonly required to be decentralized.

5. Actuators and sensors must be selected before the controller can be designed.

6. Inputs and outputs must be paired before the design of a decentralized controller.

Researchers in the 1940s developed methods for single-input single-output plants to design controllers to be insensitive to plant/model mismatch and perform well under unmeasured or poorly modeled disturbances [7]. However, the extension of these results to multivariable systems was found not to be straightforward. "Optimal" control theory (e.g., Linear Quadratic Gaussian control) developed during the 1960s could readily handle multivariable systems. However, it was shown in the 1970s that optimal controllers can be arbitrarily sensitive to plant/model mismatch—a small perturbation in the model can lead to poor performance or even instability when the controller is applied to the real system.

This demonstrated the need to account for model uncertainty in the controller design procedure, i.e., the controller must be *robust.* An effective framework for analyzing *robustness* in multivariable systems was not developed until the 1980s. A new function, μ, was introduced as a nonconservative measure for system robustness. A synthesis method for robust controllers soon followed (referred to as *DK-iteration* or *μ-synthesis*) and was applied to design controllers for a large number of academic case studies such as high purity distillation columns [106], CSTRs [76], and packed bed reactors [116].

Though a framework for robustness analysis and synthesis is available, and the methods have been applied by academicians to various processes, industrial applications have not been forthcoming. This is because the *other* important practical process control considerations (2-6) must also be addressed. This thesis presents a framework to address these control requirements/problems in a general, unified manner.

Thesis Overview The structured singular value framework which is used to analyze the robustness of uncertain systems is summarized. The synthesis technique proposed by Doyle in the early 1980s [30] is applied to a simple mass-spring exam-

ple, both to familiarize the reader with the framework, and to illustrate important practical control considerations which are not addressed by the design technique.

Next the detailed modeling, identification, and control for an industrial scale adhesive coater is presented. Because the dynamics and interactions for this process were particularly simple, many important process control considerations (such as the effect of design on control, and how to design low order controllers and handle actuator constraints and real parametric uncertainty) could easily be addressed which would have been much more difficult to address in general. This motivates the core of this thesis which is the development of a general approach, called *robust loopshaping*, for addressing these and other control considerations described below.

The robust loopshaping framework is shown to be a direct generalization of classical loopshaping which was so successful for single loop design in the 1930s-40s. It is shown how to use the framework to design low order controllers (e.g., PID) which are easier for operators to understand and maintain, and decentralized controllers which are the rule rather than the exception in industrial process control. It is shown how to analyze the reliability of control systems, and how to design controllers which are inherently reliable to equipment faults or failures. These techniques are illustrated on a high purity distillation column.

The robust loopshaping framework is used to develop tools for choosing actuators and sensors to use for control purposes in the presence of model/plant mismatch. In decentralized controller design, these tools are also used for determining the appropriate partitions and pairings of controller inputs and outputs. New results are presented, as well as simplified and unified proofs of existing results. Application to a distillation column illustrates the importance of considering plant/model mismatch in choosing actuators and sensors. A branch-and-bound procedure for control structure selection is described which can greatly reduce the number of candidates under consideration. These tools also provide recommendations on how to modify the *plant*

design to improve the closed loop control.

This leads to the next part of the thesis which explores computational issues associated with both the structured singular value and robust loopshaping frameworks. First we develop a method to reduce conservatism in the analysis of constraints by covering them with a nonlinear *real* parametric uncertainty description. This system with nonlinear uncertainty is converted into a constant-matrix μ problem so that stability and performance can be analyzed using off-the-shelf software. We discuss how these results can also be applied to analyze the stability and performance globally for gain-scheduled systems and locally for general nonlinear systems.

Next we address the computational complexity of the matrix function μ, which is an integral part of robustness analyses. It is shown that *any* algorithm for exactly calculating μ has exponential growth in the size of the problem, which motivates the approach by Doyle and co-workers [29, 37, 119] which is to calculate tight polynomial-time upper and lower bounds instead. The last computational issue addressed is the development of a polynomial-time method for calculating the minimized scaled condition number, which is useful for analyzing robust stability and for choosing and pairing actuators and sensors for control purposes. This is followed by a summary and ideas for future research.

Part II

Robust Control and the Needs of Process Control

Chapter 2

Structured Singular Value Framework

Summary

In practice, a model is only an approximate description of the physical system. Unknown disturbances, uncertainty about actuator and sensor dynamics, and inaccurate values for the parameters of the physical system make it impossible to generate an exact model. The error between the true behavior of the physical process and that predicted by the model can significantly affect the ability of the control system to meet the performance requirements. For the controller to work satisfactorily on the real system, the controller must be designed to be insensitive to this model uncertainty. Controllers that satisfy the specified performance requirements and are insensitive to model uncertainty are said to be *robust.*

The structured singular value (or μ) framework was developed in the early 1980s to nonconservatively analyze robustness. A method of designing robust controllers soon followed (referred to as *DK-iteration*), and was applied to a large number of academic case studies such as high purity distillation columns, CSTRs, and fixed bed reactors. This chapter describes the structured singular value framework, and then provides a simple example showing how to use this framework. Though the structured

singular value framework provides a general approach to addressing many uncertainty and performance specifications, we list some important process control considerations which are not directly addressed. This motivates the search for a broader framework for process control.

2.1 Robust Performance

The goal of any controller design is that the overall system is stable and satisfies some minimum performance requirements. These requirements should be satisfied at least when the controller is applied to the *nominal* plant, that is, we require nominal stability and nominal performance.

In practice the real plant $\hat{P}$ is not equal to the model P. The term *robust* is used to indicate that some property holds for a set Π of possible plants $\hat{P}$ as defined by the uncertainty description. In particular, by *robust stability* we mean that the closed loop system is stable for all $\hat{P} \in \Pi$. By *robust performance* we mean that the performance requirements are satisfied for all $\hat{P} \in \Pi$. Performance is commonly defined in robust control theory using the H_∞-norm of some transfer function of interest.

Definition 2.1 *The closed loop system exhibits nominal performance if*

$$\|\Sigma\|_\infty \equiv \sup_\omega \overline{\sigma}(\Sigma) < 1. \tag{2.1}$$

Definition 2.2 *The closed loop system exhibits robust performance if*

$$\|\hat{\Sigma}\|_\infty \equiv \sup_\omega \overline{\sigma}(\hat{\Sigma}) < 1, \quad \forall \hat{P} \in \Pi. \tag{2.2}$$

For example, for rejection of disturbances at the plant output, Σ would be the weighted sensitivity

$$\begin{aligned} \Sigma &= W_1 S W_2, \quad S = (I + PK)^{-1}, \\ \hat{\Sigma} &= W_1 \hat{S} W_2, \quad \hat{S} = (I + \hat{P}K)^{-1}. \end{aligned} \tag{2.3}$$

In this case, the input weight W_2 is often equal to the disturbance model. The output weight W_1 is used to specify the frequency range over which the sensitivity function should be small and to weigh each output according to its importance. The transfer function of the controller is denoted K.

Doyle [29] derived the *structured singular value*, μ, to test for robustness of uncertain systems. To use μ we must model the uncertainty (the set Π of possible plants $\hat{P}$) as norm bounded perturbations (Δ_i) on the nominal system. Through weights each perturbation is normalized to be of size one

$$\|\Delta_i\|_\infty \leq 1, \tag{2.4}$$

where Δ_i is complex for representing unmodeled dynamics, and real for representing parametric uncertainty. The perturbations, which may occur at different locations in the system, are collected in the block-diagonal matrix Δ_U (the U denotes uncertainty)

$$\Delta_U = \text{diag}\,\{\Delta_i\} \tag{2.5}$$

and the system is arranged to match the left block diagram in Fig. 2.1. The interconnection matrix M in Fig. 2.1 is determined by the nominal model (P), the size and nature of the uncertainty, the performance specifications, and the controller (K).

Without loss of generality we assume that each Δ_i and M is square [64]. The definition of μ is:

Definition 2.3 *Let $M \in \mathcal{C}^{n\times n}$ be a square complex matrix and define the set $\mathbf{\Delta}$ of block-diagonal perturbations by*

$$\begin{aligned}\mathbf{\Delta} \equiv \Big\{ &\text{diag}\left\{\delta_1^r I_{r_1}, \cdots, \delta_k^r I_{r_k}, \delta_{k+1}^c I_{r_{k+1}}, \cdots, \delta_m^c I_{r_m}, \Delta_{m+1}, \cdots, \Delta_l\right\}\Big| \\ &\delta_i^r \in \mathcal{R}, \delta_i^c \in \mathcal{C}, \Delta_i \in \mathcal{C}^{r_i\times r_i}, \sum_{i=1}^{l} r_i = n \Big\}. \end{aligned} \tag{2.6}$$

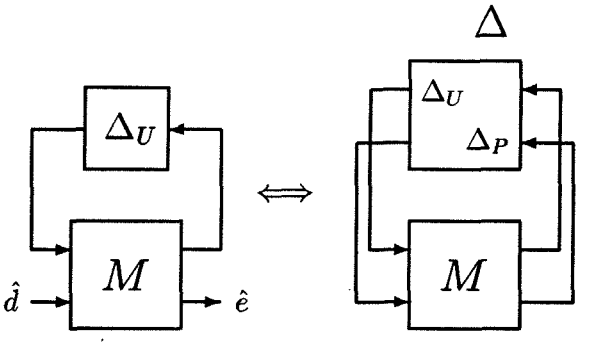


Figure 2.1: Robust performance and the $M - \Delta$ block structure.

Then $\mu_{\Delta}(M)$ (the structured singular value with respect to the uncertainty structure $\mathbf{\Delta}$) is defined as

$$\mu_{\Delta}(M) \equiv \begin{cases} 0 \text{ if there does not exist } \Delta \in \mathbf{\Delta} \text{ such that } \det(I - M\Delta) = 0, \\ \left[\min_{\Delta \in \mathbf{\Delta}} \{\overline{\sigma}(\Delta) \,|\det(I - M\Delta) = 0\}\right]^{-1} \quad \text{otherwise.} \end{cases} \tag{2.7}$$

Partition M in Fig. 2.1 to be compatible with $\Delta = \text{diag}\{\Delta_U, \Delta_P\}$:

$$M = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}. \tag{2.8}$$

The following are tests for robust stability and robust performance [29].

Theorem 2.1 *The closed loop system exhibits robust stability for all $\|\Delta_U\|_{\infty} \leq 1$ if and only if the closed loop system is nominally stable and*

$$\mu_{\Delta_U}(M_{11}(j\omega)) < 1 \qquad \forall \omega. \tag{2.9}$$

Theorem 2.2 *The closed loop system exhibits robust performance for all $\|\Delta_U\|_{\infty} \leq 1$ if and only if the closed loop system is nominally stable and*

$$\mu_{\Delta}(M(j\omega)) < 1 \qquad \forall \omega, \tag{2.10}$$

where $\mathbf{\Delta} = \text{diag}\{\Delta_U, \Delta_P\}$, *and* Δ_P *is a full square matrix with dimension equal to the number of outputs (the subscript* P *denotes performance).*

Multiple performance objectives can be tested similarly using block-diagonal Δ_P.

The value of $\mu_\Delta(M)$ depends on both the elements of the matrix M and the structure of the perturbation matrix Δ. Note that the issue of robust stability is simply a special case of robust performance. Also note that robust performance implies robust stability, i.e., $\sup_\omega \mu_\Delta(M) \geq \sup_\omega \mu_{\Delta_U}(M_{11})$.

It is a key idea that μ is a *general* analysis tool for determining robust performance. Any system with uncertainty adequately modeled as in (2.4) can be put into $M - \Delta_U$ form, and robust stability and robust performance can be tested using (2.9) and (2.10). Standard programs calculate the M and Δ [3], given the transfer functions describing the system components and the location of the uncertainty and performance blocks Δ_i.

Computation of μ The value of μ is commonly calculated through upper and lower bounds. Define three subsets of $\mathcal{C}^{n\times n}$

$$\mathcal{Q} = \{\Delta \in \mathbf{\Delta} : -1 \leq \delta_i^r \leq 1, |\delta_i^c| = 1, \Delta_i^* \Delta_i = I_{r_i}\}, \tag{2.11}$$

$$\mathcal{D} = \left\{\text{diag}\left[D_1, \cdots, D_m, d_{m+1} I_{r_{m+1}}, \cdots, d_l I_{r_l}\right] : 0 < D_i = D_i^* \in \mathcal{C}^{r_i \times r_i}, 0 < d_i \in \mathcal{R}\right\}, \tag{2.12}$$

$$\mathcal{G} = \left\{\text{diag}\left[G_1, \cdots, G_k, O_{r_{k+1}}, \cdots, O_{r_l}\right] : G_i = G_i^* \in \mathcal{C}^{r_i \times r_i}\right\}. \tag{2.13}$$

Then [37]

$$\max_{Q\in\mathcal{Q}} \rho_r(QM) = \mu_\Delta(M) \leq \sqrt{\max\left\{0, \inf_{\substack{D\in\mathcal{D} \\ G\in\mathcal{G}}} \overline{\lambda}\left[\tilde{M}^*\tilde{M} + j\left(G\tilde{M} - \tilde{M}G\right)\right]\right\}}, \tag{2.14}$$

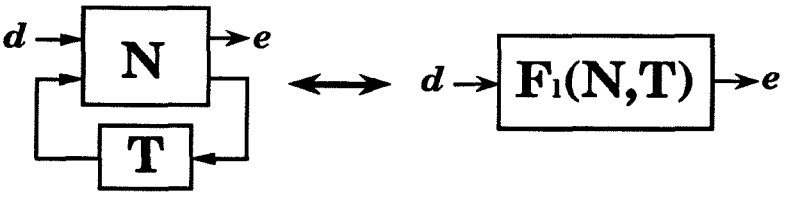


Figure 2.2: Definition of the linear fractional transformation $F_l(N,T)$.

where $\tilde{M} \equiv DMD^{-1}$, $\overline{\lambda}(A)$ is the maximum eigenvalue of A, and $\rho_r(A) \equiv \max\{|\lambda| : \lambda$ is a real eigenvalue of $M\}$.

The leftmost maximization defined in (2.14) is not convex, so an algorithm which attempts to calculate the maximum may converge to a local optimum which would be a lower bound for μ. In contrast, the computation of the upper bound in (2.14) is convex, and so convergence is assured. However, a gap may exist between the upper bound and μ. The upper and lower bounds are almost always within a percent or so for pure complex uncertainty [87]. The gap may be larger when there are real uncertainties. Off-the-shelf software computes the upper and lower bounds for general uncertainty and usually gives a narrow gap [3, 119]. Chapter 8 discusses the pitfalls in attempting to calculate μ exactly.

Linear Fractional Transformations The linear interconnection structure in Fig. 2.2) is called a linear fractional transformation (LFT). The lower LFT denoted $F_l(N,T)$ is defined by

$$F_l(N,T) = N_{11} + N_{12}T(I - N_{22}T)^{-1}N_{21}. \tag{2.15}$$

The LFT $F_l(N,T)$ is well-defined if and only if the inverse of $I - N_{22}T$ exists. A superscript is sometimes used on N, e.g. N^T, to denote that N depends on the choice of T.

The subscript l on F_l is used to denote that the lower loop of N is closed by T.

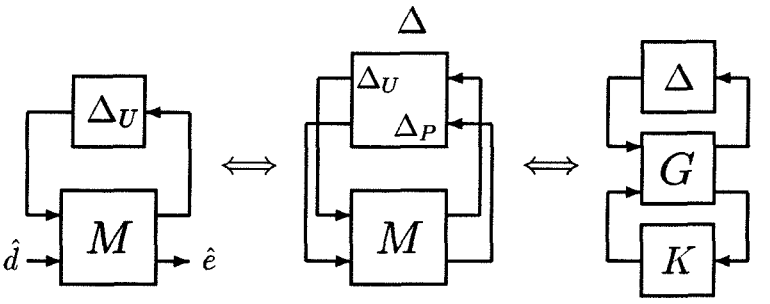


Figure 2.3: General interconnection structures.

When the upper loop is closed, the transfer function between inputs and outputs is the LFT $F_u(N,T) = N_{22} + N_{21}T(I - N_{11}T)^{-1}N_{12}$.

Controller Synthesis With Complex Δ The H_∞-optimal control problem is to find a stabilizing K which minimizes $\sup_\omega \overline{\sigma}\left(F_l(G,K)\right)$ (see Fig. 2.3). The state-space approach for solving the H_∞ control problem is described in [40].

For pure complex uncertainty, the upper bound in (2.14) reduces to $\inf_{D\in\mathcal{D}} \overline{\sigma}\left(DMD^{-1}\right)$. The DK-iteration method (often called μ-synthesis) is an *ad hoc* method which attempts to minimize this tight upper bound of μ for complex uncertainty, i.e., it attempts to solve

$$\inf_K \inf_{D\in\mathcal{D}} \sup_\omega \overline{\sigma}\left(DF_l(G,K)D^{-1}\right). \tag{2.16}$$

The approach in DK-iteration is to alternatively minimize $\sup_\omega \overline{\sigma}\left(DF_l(G,K)D^{-1}\right)$ for either K or D while holding the other constant. For fixed D, the controller synthesis is solved via H_∞-optimization. For fixed K, the quantity is minimized as a convex optimization. The resulting D as a function of frequency is fitted with an invertible stable minimum-phase transfer function and wrapped back into the nominal interconnection structure. This increases the number of states of the scaled G, which leads the next H_∞-synthesis step to give a higher order controller. The iterations stop

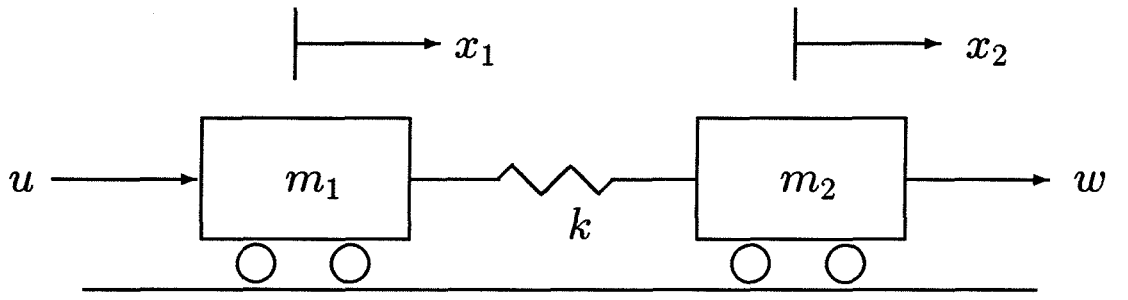


Figure 2.4: Coupled mass-spring system.

after $\sup_{\omega} \overline{\sigma}\left(DF_l(G,K)D^{-1}\right)$ is less than 1 or is no longer diminished. The resulting high-order controller is typically reduced using Hankel model reduction [39]. Though this method is not guaranteed to converge to a global minimum, it has been used to design robust controllers for many mechanical systems, e.g., flexible space structures [2], missile autopilots [92, 55], and rockets [36].

Synthesizing controllers with mixed real and complex perturbations is much more difficult than in the pure-complex case, and no reliable optimization-based synthesis algorithm currently exists.

2.2 Example: A Coupled Mass-Spring System

To illustrate the use of the structured singular value framework, we now apply it to design a robust controller for an undamped pair of coupled masses with a noncolocated sensor and actuator. This simple problem captures many of the features of more complex aircraft and space structure vibration control problems. It is shown how design specifications such as settling time, actuator constraints, insensitivity to measurement noise, and parameter uncertainty are addressed in this framework.

Problem Description Consider the two-mass/spring system in Fig. 2.4, which is a generic model of an uncertain dynamical system with noncolocated sensor and actuator.

The system is represented in state-space form as

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -k/m_1 & k/m_1 & 0 & 0 \\ k/m_2 & -k/m_2 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1/m_1 \\ 0 \end{bmatrix} u + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1/m_2 \end{bmatrix} w \quad (2.17)$$

$$y = x_2 + v \quad (2.18)$$

$$z = x_2 \quad (2.19)$$

where x_1 and x_2 are the positions of body 1 and body 2, x_3 and x_4 are the velocities of body 1 and body 2, u is the control input acting on body 1, y is the sensor measurement, w is the disturbance acting on body 2, v is sensor noise, and z is the output to be controlled. The spring constant is denoted by k, the mass of body 1 by m_1, and the mass of body 2 by m_2.

The coupled spring-mass system is assumed to have negligible damping. The spring constant and masses are assumed to be uncertain. The actuator is located on body 1 while the sensor is located on body 2, i.e., the sensor and actuator are *noncolocated.* This makes the system much harder to control than in the colocated case.

The design specifications are:

(i) Maximize the stability margin with respect to the three uncertain parameters m_1, m_2, k whose nominal values are $m_1 = m_2 = k = 1$.

(ii) For $w(t) =$ unit impulse at $t = 0$, the performance variable z has a settling time of 15 seconds for the nominal system $m_1 = m_2 = k = 1$. The settling time is defined to be the time required for the output to reach and stay within 10% of its peak value.

(iii) The control system can tolerate Gaussian white noise with variance of $9 * 10^{-6}$.

(iv) Because of finite actuator response time, the controller bandwidth must be ≤ 50 rad/s.

(v) The control input $u(t)$ is limited to $|u| \leq 1$.

(vi) The number of controller states should be ≤ 4.

Building the Generalized Block Diagram The uncertain spring constant and the two masses are described by

$$\begin{aligned} k &= k_0 + w_k \delta_k, \\ m_1 &= m_{10} + w_1 \delta_1, \\ m_2 &= m_{20} + w_2 \delta_2, \end{aligned} \tag{2.20}$$

where k_0, m_{10}, and m_{20} are the nominal values and the weights w_k, w_1, and w_2 are used to normalize the uncertainties δ_i so that $|\delta_i| \leq 1$. Simultaneous perturbations in the δ_i are allowed, as long as $|\delta_i| \leq 1$ for each uncertainty i.

Weighted versions of the noise, disturbance, control input, and performance variable are given by

$$\begin{aligned} v &= w_v v', \\ w &= w_w w', \\ u' &= w_u u, \\ z' &= w_z z, \end{aligned} \tag{2.21}$$

where in general the input weights w_v and w_w weigh the frequencies to be rejected and determine the relative important of the noise and disturbance. The performance weight is w_z and w_u is used to limit the magnitude of the control input.

The expressions for k, m_1, and m_2 from (2.20) and w, v, z, and u from (2.22) are

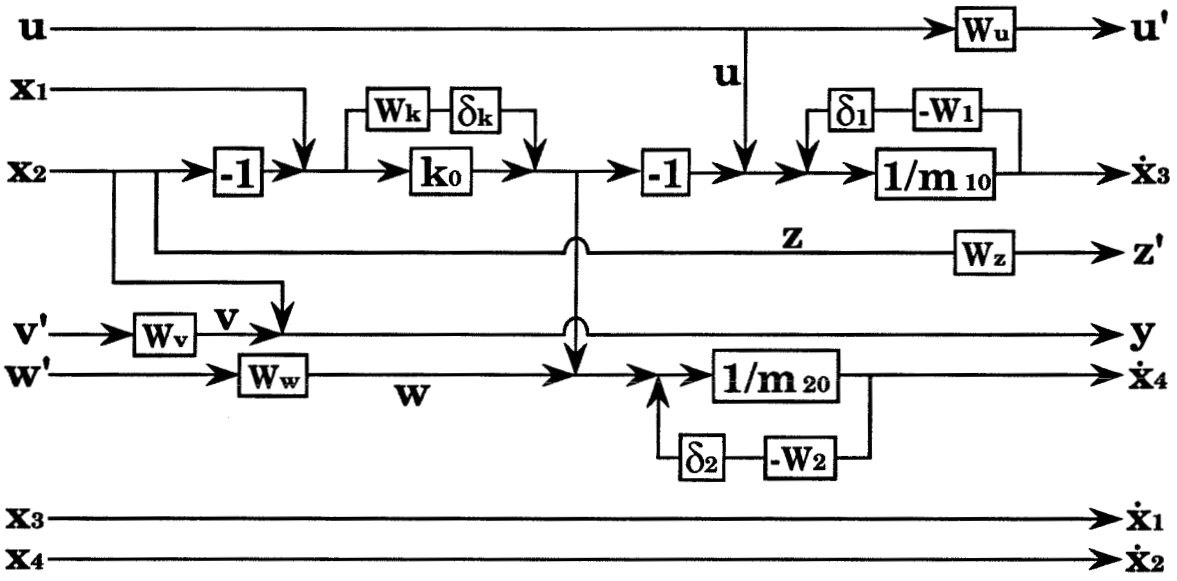


Figure 2.5: Block diagram for coupled mass-spring system.

substituted into the state-space equations (2.17-2.19) and written in block diagram form in Fig. 2.5. The block diagram has x, u, v', w' as inputs and $\dot{x}$, u', z', and y as outputs.

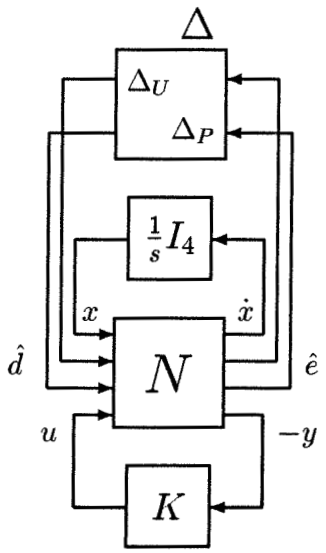


Figure 2.6: Simplified block diagram for coupled mass-spring system.

By inspection, the block diagram in Fig. 2.5 is rearranged to form the block diagram in Fig. 2.6, where

$$N = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{k}{m_1} & \frac{k}{m_1} & 0 & 0 & -\frac{1}{m_1} & \frac{1}{m_1} & 0 & 0 & 0 & \frac{1}{m_1} \\ \frac{k}{m_2} & -\frac{k}{m_2} & 0 & 0 & \frac{1}{m_2} & 0 & \frac{1}{m_2} & 0 & \frac{w_w}{m_2} & 0 \\ w_k & -w_k & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{kw_1}{m_1} & -\frac{kw_1}{m_1} & 0 & 0 & \frac{w_1}{m_1} & -\frac{w_1}{m_1} & 0 & 0 & 0 & -\frac{w_1}{m_1} \\ -\frac{kw_2}{m_2} & \frac{kw_2}{m_2} & 0 & 0 & -\frac{w_2}{m_2} & 0 & -\frac{w_2}{m_2} & 0 & -\frac{w_2 w_w}{m_2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & w_u \\ 0 & w_z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & -w_v & 0 & 0 \end{pmatrix}, \tag{2.22}$$

and the normalized performance variable $\hat{e}$, the normalized disturbance $\hat{d}$, and the uncertainty block Δ_U are given by

$$\hat{e} = \begin{pmatrix} u' \\ z' \end{pmatrix}, \tag{2.23}$$

$$\hat{d} = \begin{pmatrix} v' \\ w' \end{pmatrix}, \tag{2.24}$$

$$\Delta_u = \begin{pmatrix} \delta_k & & \\ & \delta_1 & \\ & & \delta_2 \end{pmatrix}. \tag{2.25}$$

The performance block Δ_P relates the outputs to inputs, K is the controller transfer function, and I_4 is the 4×4 identity matrix. Closing the integrator loop in Fig. 2.6 gives the system interconnection structures in Fig. 2.3.

It can be seen from (2.17-2.19) that the transfer function between the disturbance w and the output z contains a double-integrator. In this case assumptions A1 and A3 in [40] needed to solve the H_∞-control problem are not satisfied. Ways of reformulating the problem to satisfy the assumptions are discussed in [40] and [94]. It is suggested in [94] to either use a bilinear transform to move all open loop poles off

the imaginary axis, or to choose the disturbance and performance weights to cancel the integrators. Three methods are suggested in [40]; the simplest method is to introduce an ϵ perturbation so that assumptions A1 and A3 are satisfied. Choosing this method, we slightly perturbed the poles on the imaginary axis by using $\frac{1}{s+0.00001}$ instead of $\frac{1}{s}$ in Fig. 2.6. However, all results reported here use the true integrator, and no problems were found to result from using the "almost-integrator" instead of the true integrator for the controller synthesis.

The DK-iteration method does not handle real uncertainty directly, so we will treat the real parametric uncertainty in k, m_1, and m_2 as being complex. As such, the DK-iteration method will give a controller whose performance is insensitive to the complex uncertainties. This will also tend to make the controller less sensitive to real uncertainty in k, m_1, and m_2. We will later test the conservatism in allowing the real uncertainty to be complex.

Strategy for Choosing Input and Output Weights The advantage of the structured singular value framework over many other design methods is that it yields directly controllers that are insensitive to model uncertainty. One disadvantage is that performance specifications such as (ii) and (v) can not explicitly be put in terms of the ∞-norm in (2.2). Though there is no explicit relationship between the performance specifications and the ∞-norm, decreasing the ∞-norm of the transfer function between the inputs w' and v' to the outputs z' and u' does improve the speed of response and decrease the peak outputs.

The key to the synthesis technique is the selection of the weights w_w, w_v, w_z, and w_u. The controller synthesis procedure is much faster when lower order weights are used, so constant weights should be used when possible.

The approach to choosing the weights w_w, w_v, w_z, and w_u will be as follows. It can be shown that multiplying w_w and w_v by a scalar transfer function and dividing

w_z and w_u by the same scalar transfer function does not change the ∞-norm in (2.2). Thus without loss of generality we can take $w_w = 1$. Since the noise v is expected to have much smaller effect on the system than the disturbance w, we will choose the noise weight w_v to be small. Noise weights typically are chosen to have larger gain at high frequency, but we expect that the effect of the noise on the system is small enough that choosing a frequency-dependent w_v will not give a controller much better than that when choosing a constant noise weight. If the simulations show sensitivity to measurement noise, then w_v will be increased.

There is no known explicit relationship between the frequency-dependent output weights w_z and w_u and the resulting settling time and peak control input. Some general empirical guidelines for choosing frequency-dependent weights are [55]:

1. choose high gain weight at mid-range to high frequency in order to give small peak values, and

2. choose an even higher gain at low frequency for good tracking.

Guideline 1 implies that w_u should have high gain at high frequencies to keep the peak control input small. High gain for w_u at high frequencies should also cause the controller to avoid high frequency control inputs (specification (iv)).

It can be shown from (2.17-2.19) that an *impulse* disturbance will give no steady-state offset in u and z as long as the controller is internally stable. Thus w_u and w_z need not have a higher gain at low frequency (it is suspected that guideline 2 was intended only to be used to design for tracking of *step* inputs). Since high gain at low frequency is not needed for w_u and w_z, and low-order weights are desirable for controller synthesis, we will use constant w_u and w_z. Increasing w_u and w_z will increase the overall performance of u and z; this will decrease the peak control input and decrease the settling time, respectively. w_u will be chosen large enough so that the peak control input constraint $\max_t |u(t)| \approx 1$ is met. The performance weight w_z

will be chosen large enough so that the settling time specification (ii) is met. If we could not achieve the design specifications using constant weights, then frequency-dependent weighting would be considered.

Defining μ for Robust Performance Since the performance specifications are not explicitly in terms of the ∞-norm, we are not particularly interested in meeting condition (2.10). We are not interested in meeting condition (2.9) for robust stability for complex uncertainties either. We *are* interested in meeting the stability robustness specification (i). In other words, the design is complete when specifications (i-vi) are met, regardless of whether conditions (2.9) or (2.10) are satisfied.

The performance block Δ_P was chosen to be a diagonal matrix with two independent 1×1 blocks for all designs. This decouples the performance specifications that u be small (peak magnitude less than 1) and that z respond quickly to a unit impulse in w. This makes choosing a satisfactory w_u and w_z easier. Also, μ for robust performance defined with this choice of Δ_P is less than μ for robust performance defined using the typical choice of full block Δ_P. Thus, this choice for Δ_P gives a smaller difference between the structured singular value for robust performance and for robust stability, allowing the DK-iteration design method to more directly enhance stability robustness. The specific control design follows.

Robust Controller Design The goal is to maximize the stability with respect to uncertain m_1, m_2, and k with nominal values $m_{10} = m_{20} = k_0 = 1$. Initially we design for 20% complex uncertainty, i.e., $w_1 = w_2 = w_k = 0.2$.

First we choose only constant weights to specify w_w, w_v, w_z, and w_u. Without loss of generality we can choose one of these weights to be 1; we took $w_w = 1$. Since the measurement noise is small in magnitude compared to the size of the disturbance, we initially chose the noise weight to be much smaller than the disturbance weight, $w_v = w_w/100 = 0.01$. Increasing w_z decreases the peak value and settling time for z.

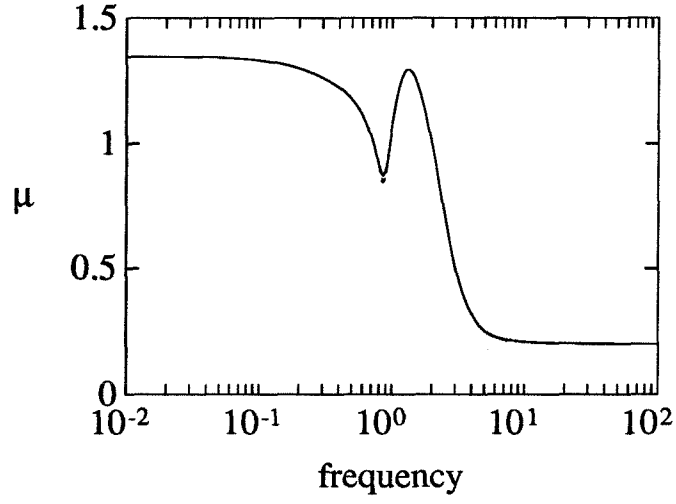


Figure 2.7: μ for robust performance.

Increasing w_u decreases the peak control input.

The DK-iteration design procedure was initially performed for $w_z = w_u = 1$. Simulations with the resulting controller showed that the nominal settling time specification was easily met, but the nominal peak control input was 11 and robust stability was not satisfied. To decrease the peak control input, w_u was increased to 15. Since robust stability was not satisfied and there was excess performance in z, we decided to trade off performance to get increased stability by iterating w_z with w_w and w_v unchanged. After several iterations, the performance weight $w_z = 0.12$ was chosen which meets all the design specifications (i-vi).

The settling time and peak control input were close to their maximum values, so performance could not be traded off for an appreciably larger stability margin (specification (i)).

Each time the DK-iteration method was used, one "D" iteration was needed—further "DK" iterations did not diminish the objective in (2.16). The 24-state controller was reduced to 4 states with negligible loss in stability and performance. The structured singular value for performance with the resulting controller is in Fig. 2.7.

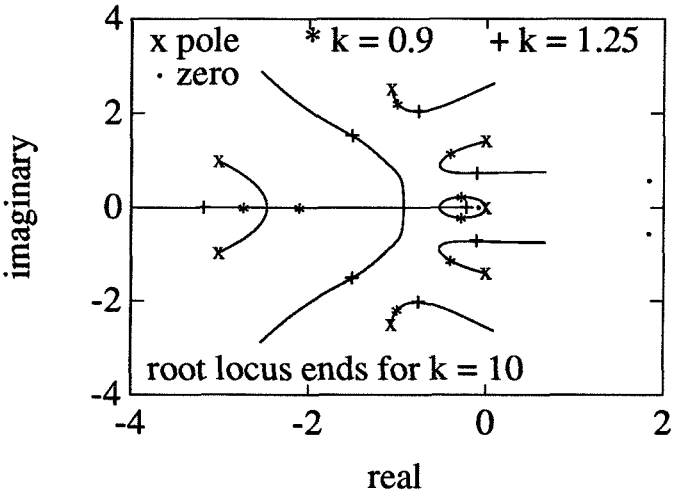


Figure 2.8: Root locus.

The controller after Hankel model reduction is given by:

$$A_c = \begin{bmatrix} -0.5178 & -1.521 & -1.558 & -0.0820 \\ 4.725 & -5.510 & 5.878 & 0.4984 \\ 0 & 0 & -2.039 & 2.432 \\ 0 & 0 & -2.974 & -0.0832 \end{bmatrix}, \tag{2.26}$$

$$B_c = \begin{bmatrix} -0.8847 \\ 2.702 \\ -3.259 \\ -0.8716 \end{bmatrix}, \tag{2.27}$$

$$C_c = \begin{bmatrix} -1.471 & 4.125 & -0.5234 & -0.1214 \end{bmatrix}, \tag{2.28}$$

$$D_c = 0.00372. \tag{2.29}$$

The zeros for the above controller are $\{-3847, -0.1392, 1.780 \pm 0.5621i\}$ and the poles are $\{-3.014 \pm 0.9775i, -1.061 \pm 2.505i\}$. The zero at -3847 is far in the left half plane, and so has a small effect on stability and performance. This zero was dropped to make the final controller strictly proper. See Fig. 2.8 for the root locus.

The controller bandwidth (read from the controller's Bode magnitude plot) is 21

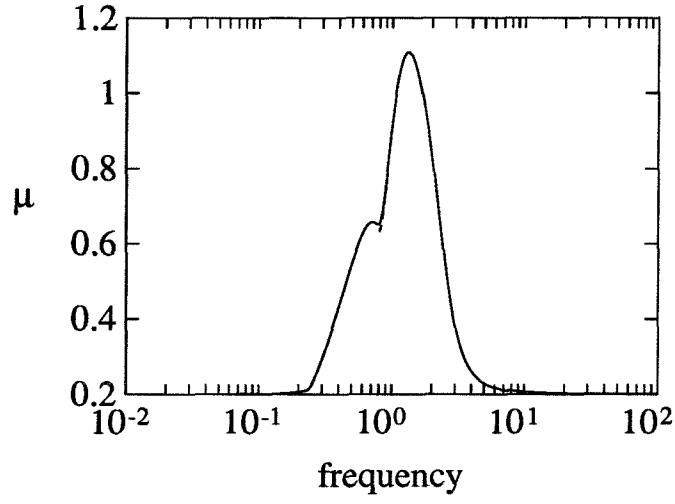


Figure 2.9: μ for complex robust stability.

rad/s. The gain and phase margins are read from the Nyquist plot and found to be 1.43 and 28.2°, respectively. Fig. 2.9 is a plot of the structured singular value for robust stability with complex uncertainty. The peak value on this plot is $\mu^c_{RS} = 1.11$.

The value of μ with pure real parametric uncertainty was calculated (details on calculation are described in [9]) to be $\mu^r_{RS} = 0.67$. This implies that the closed loop system is stable under simultaneous independent real parameter variations up to 30%, i.e., the closed loop system is stable for any values of k, m_1, and m_2 given by

$$k \in [0.7, 1.3], \qquad m_1 \in [0.7, 1.3], \qquad m_2 \in [0.7, 1.3]. \tag{2.30}$$

The conservatism in using complex uncertainty in k, m_1, and m_2 over using real uncertainty is $(\mu^c_{RS} - \mu^r_{RS})/\mu^r_{RS} = (1.11 - 0.67)/0.67 = 67\%$. The lower and upper parameter margins (with $m_1 = m_2 = 1$) for k are 0.55 and 2.5, respectively. The lower and upper parameter margins for m_1 (with $k = m_2 = 1$) are 0 and 3.4, respectively. m_2 has the same parameter margins as for m_1.

The time domain plots for the mass positions x_1 and x_2 and the control input u are given for $m_1 = m_2 = k = 1$ (see Fig. 2.10 and 2.11). The settling times for both

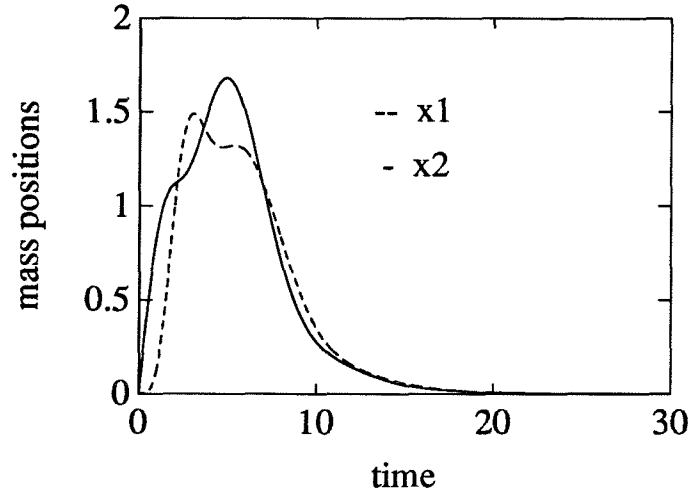


Figure 2.10: Time responses for masses 1 and 2.

x_1 and x_2 are less than 15 seconds. The maximum control input is less than one. All responses are insensitive to measurement noise.

Discussion The gain and phase margins are low for both designs, though the phase margin for Design #2 is very near the $30^o - 60^o$ suggested in most textbooks. The gain and phase margins can be included in the structured singular value framework (though this is cumbersome, see [55]). This was not done here because gm and pm

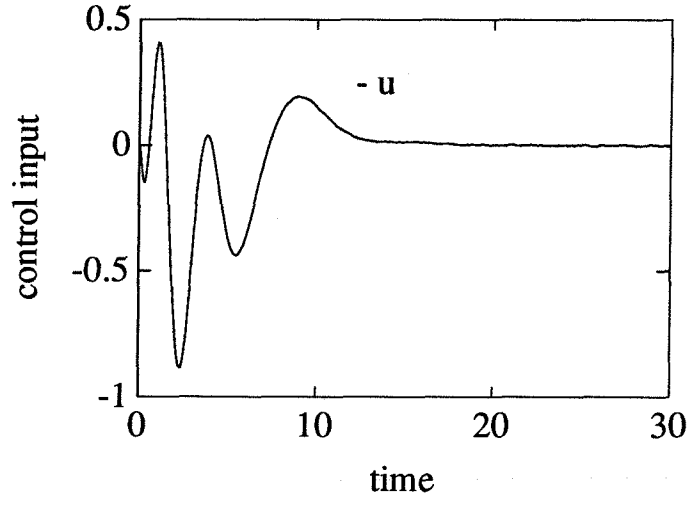


Figure 2.11: Time response for control input.

were not in design specifications. As seen in both designs, gain and phase margins are not necessary for having good parameter margins.

Covering real parameter variations by complex uncertainties was found to be quite conservative. This implies that a controller design procedure that directly takes into account the real nature of k, m_1, and m_2 may give better designs.

This example points out that design specifications must be chosen carefully before the controller is designed. For example, recall that the performance weight w_z was not chosen to have higher gain at low frequency because an impulse disturbance to the mass-spring system gives no steady-state offset. Though the designed controllers will reject impulse disturbances, they give poor rejection of *step* disturbances. If step disturbances are to be expected, then this must be put into the design specifications so that the appropriate weights are chosen in the design procedure. Similar comments can be made concerning sinusoidal disturbances.

Additional designs, including two-degree-of-freedom designs, are described in [9, 10].

2.3 Important Issues in Process Control

The strengths of the structured singular value framework are that it addresses uncertainty and performance specifications in a general, unified manner. The following important practical process control considerations *also* need to be addressed:

- Actuator constraints are of great importance in industrial processes. A process typically operates at or near constraints—otherwise the process would be overdesigned leading to large equipment costs. Including a sufficiently large weight on the control action to avoid actuator constraints for a specific disturbance, as done in the example, will provide a controller which is sluggish for small disturbances, and ineffective for disturbances of larger magnitude.

- Though off-the-shelf software exists for analyzing systems with real parametric uncertainty, the synthesis method (DK-iteration) does not directly address real uncertainty. Covering real uncertainty by complex can be conservative, as shown in the example.

- Many specifications are difficult (or impossible) to address within the DK-iteration design procedure. These include gain and phase margin, fault and failure tolerance, controller order, and multiple but *independent* performance specifications.

- Practical control problems often involve more actuators and sensors than are needed for designing effective, economically viable control systems. An appropriate set of actuators and sensors must be selected from the available candidates. A related problem is the selection between different *plant designs* in terms of the achievable closed loop performance.

- Decentralized controllers are the rule rather than the exception in process industries. DK-iteration cannot be used effectively to design these controllers. Another task in the design of decentralized controllers is that inputs and outputs must be paired before the controller design.

In the next chapter we show how to address some of these process control considerations for a class of adhesive coating processes commonly found in industry. We then develop a framework for general processes for addressing all of these control requirements/problems.

2.4 Conclusions

This chapter summarizes the structured singular value framework, and illustrates the approach on a simple mass-spring system. Though the structured singular value pro-

vides a general approach to addressing many uncertainty and performance specifications, many other important process control considerations are not directly addressed. This motivates the search for a broader framework for process control.

Chapter 3

Identification and Cross-Directional Control of Coating Processes

Summary

Of special industrial interest is the cross-directional control of coating processes, where the cross-direction refers to the direction perpendicular to the substrate movement. The objective of the controller is to maintain a uniform coating under unmeasured process disturbances. Assumptions that are relevant to coating processes found in industry are used to develop a model for control design. This model is used to derive a model predictive controller to maintain flat profiles of coating across the substrate by varying the liquid flows along the cross direction. Actuator constraints, measurement noise, model uncertainty, and the plant condition number are investigated to determine which of these limit the achievable closed loop performance. From knowledge of how these limitations affect the performance we can make some recommendations on how to modify the *plant design* to improve coating uniformity. The theory developed in this chapter is rigorously verified through experiments on a pilot plant. The controller rejects disturbances within two sampling times. The

proposed controller can reduce the variance in coating thickness by as much as 80% compared to what is possible by manual control or simple control schemes. The applicability of the control techniques to the industrial scale coating process motivates the development of an approach for general processes.

3.1 Introduction

Coating refers to the covering of a substrate with a uniform layer of liquid. Coating processes are of great importance to manufacturing, especially in the photographic, magnetic and optical memory, electronic, adhesive, and paper industries [19].

Plant Description Fig. 3.1 is a simplified diagram of a typical plant. The process begins with a feed roller from which substrate is unwound. From there, the substrate passes between a roller and a stainless steel die. The liquid flows through a slot in the die to the substrate. The cavity in the die is designed to distribute a uniform flow of liquid through the slot. A controlled pump supplies a constant flow of liquid through the die.

The term "gap width" refers to the distance across the slot at a given point along the die. The gaps through which liquid flows are adjusted by means of n equally spaced bolts. The bolts are adjusted manually.

After being coated with liquid, the substrate passes through a drier. After the drier, the time-averaged coating thickness at each of the n positions corresponding to the die bolts is measured by a traversing coat-weight sensor. The coated substrate is wound on the product roller.

For further details on die design, die flow phenomena, drying phenomena, coat-weight sensors, and other aspects of coating, see [95, 19, 20, 96] and the literature cited therein.

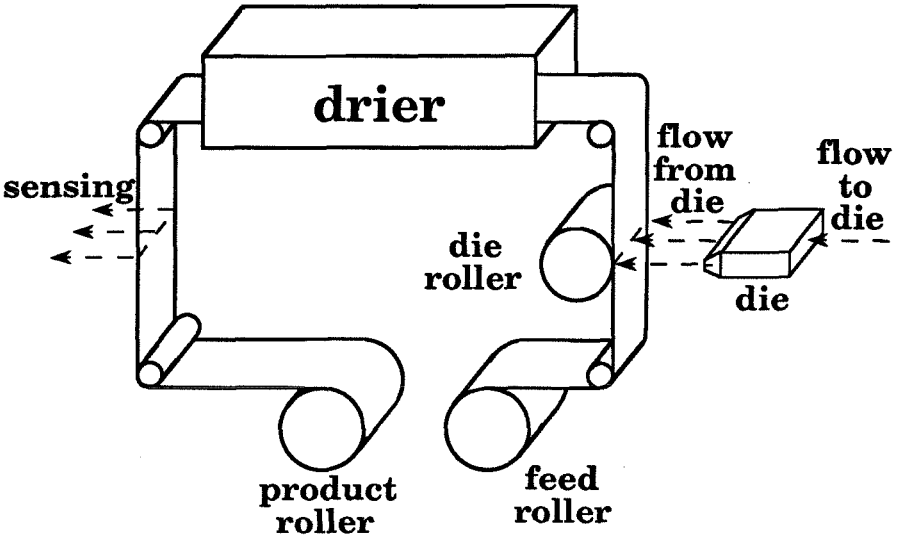


Figure 3.1: Typical coating plant.

Control Objective The cross-directional control problem is aimed at maintaining a uniform profile of liquid across the substrate. Successful control of coating thickness improves product quality and reduces the time needed to bring the plant on-line. Poor control can lead not only to coating thickness nonuniformity but also coating instabilities that leave portions of the substrate uncovered; such substrate must be rejected (for a short summary of coating instabilities, see Sartor, 1990).

We will consider coating processes with a large time delay between a change in gap width and the resulting sensing of the change in coating profile downstream. This time delay could be due to a sensor installed at a fair distance from the die as in the coating plant considered above. Because the controller cannot be expected to reject disturbances faster than this time delay, detailed process dynamics are not considered in the modeling, identification, and control of the cross-directional coating process. Thus the objective of the controller is the elimination of slow disturbances in the coating thickness. The disturbances were of this nature in the Avery/Dennison pilot plant; the control of this plant is studied in this chapter.

Organization Assumptions that are relevant to a subset of coating processes found in industry are used to develop a model for control design. This model is used to derive an unconstrained model predictive controller to maintain flat profiles of liquid across the substrate by varying the gap widths. Several modifications to the unconstrained controller are proposed to prevent physically infeasible actuator movements (gap widths). The simplest yet effective constraint-handling method is chosen.

Actuator constraints, measurement noise, model uncertainty, and the plant condition number are investigated to determine which of these limit the achievable closed loop performance. The theory developed throughout the chapter is applied to a pilot plant liquid coating process at the Avery/Dennison Research Center in Pasadena.

The majority of this chapter was published in *AIChE Journal* [13].

Notation Because of the large amount of matrix manipulations made in this chapter, the notation for this chapter is more specialized than for the rest of the thesis. All scalars are italicized. Matrices are upper case bold. The (i,j) element of the matrix $\mathbf{M}$ is denoted by $M_{i,j}$. Vectors are lower case and bold. The ith element of the vector $\mathbf{x}$ is represented as x_i. The vector $\mathbf{x}(t)$ refers to the value of $\mathbf{x}$ at time t.

3.2 Model Development

Below we make assumptions on the plant that are relevant to a subset of coating processes found in industry. These assumptions are used to develop a dimensional model. This model is transformed to a dimensionless form. The dimensionless model is then rearranged into a form suitable for controller design.

Dimensional Model

Consider a plant with the number of actuators n equal to the number of sensors (or sensor measurement positions). It has been found experimentally (through ex-

amination of pilot plant data) that the plant behaves approximately linearly in the operating region. Let $\breve{\mathbf{u}}$ be the vector of gap widths, $\breve{\mathbf{x}}$ be the vector of coating thicknesses, and $\breve{\mathbf{v}}$ collect any effects on the coating thickness not due to changes in gap width. If the process dynamics are approximated by a pure delay, then the coating thickness at sampling instant t is related to the gap width at the previous sampling instant through

$$\breve{\mathbf{x}}(t) = \mathbf{P}\breve{\mathbf{u}}(t-1) + \breve{\mathbf{v}}(t), \tag{3.1}$$

where $\mathbf{P}$ is a constant $n \times n$ matrix.

Assumption on $\breve{\mathbf{v}}$ The vector $\breve{\mathbf{v}}$ accounts for unmeasured input effects such as measurement noise and disturbances. We assume that $\breve{\mathbf{v}}$ is a non-zero-mean stochastic variable, i.e., $\{\breve{\mathbf{v}}(0), \breve{\mathbf{v}}(1), \ldots, \breve{\mathbf{v}}(h), \ldots\}$ is a sequence of independent random vectors with non-zero mean [68]. We define the steady-state disturbance $\breve{\mathbf{d}}$ as the time-averaged value of $\breve{\mathbf{v}}$, and define $\breve{\mathbf{n}}$ by

$$\breve{\mathbf{n}}(t) = \breve{\mathbf{v}}(t) - \breve{\mathbf{d}}. \tag{3.2}$$

We will assume that $\breve{\mathbf{n}}$ is white noise. It will be referred to as measurement noise.

The unmeasured inputs $\breve{\mathbf{v}}$ are chosen to be stochastic because this describes well the apparently random fluctuations of the process. In practice, equal gap widths do not give a uniform coating because of imperfections in the roller or the die, non-uniformities in the drying process, or poor calibration of the gap widths. These imperfections lead $\breve{\mathbf{v}}$ to have non-zero mean.

Assumptions on P Typically, the total flow of coating through the die is maintained constant through a high gain controller. Because of constant total flow, increasing the flow through one actuator will necessitate decreasing the flow through the others. In the development of the model, we make the following assumptions:

1. The total liquid flow (and therefore the sum of the coating thicknesses) is constant.

2. The responses to all actuators are similar and symmetric about the actuator positions.

3. The only interactions between the actuators are due to the constant flow assumption.

Assumption 2 implies that $\mathbf{P}$ is symmetric. Assumption 3 implies that $\mathbf{P}$ can be separated into two matrices

$$\mathbf{P} = \breve{k}\mathbf{I} - \mathbf{M}, \tag{3.3}$$

where $\breve{k}$ is the gain between the ith gap width and its corresponding coating thickness for an infinitely wide die (i.e., $n \to \infty$). The $n \times n$ identity matrix is denoted by $\mathbf{I}$, $\breve{k}\mathbf{I}$ is the contribution that changing gap widths would have on the coating thicknesses if there were no interactions, and $\mathbf{M}$ represents the effect that increasing one gap width has on decreasing the flow through all the gaps. Assumption 3 also implies that all elements of $\mathbf{M}$ are equal, i.e., $M_{i,j} = m$ for $i, j = 1, 2, \cdots, n$. Then

$$\mathbf{P} = \underbrace{\begin{pmatrix} \breve{k}-m & -m & -m & \cdots & -m \\ -m & \breve{k}-m & -m & \ddots & \vdots \\ -m & \ddots & \ddots & \ddots & -m \\ \vdots & \ddots & \ddots & \breve{k}-m & -m \\ -m & \cdots & -m & -m & \breve{k}-m \end{pmatrix}}_{n \times n}. \tag{3.4}$$

Assumption 1 implies that $\sum_{i=1}^{n} \breve{x}_i$ is constant for all gap widths $\breve{\mathbf{u}}$. Then (ignoring the noise $\breve{\mathbf{n}}$), we have from (3.1) that

$$\sum_{i=1}^{n} \breve{x}_i(t) = \sum_{i=1}^{n} \left(\breve{d}_i + \sum_{j=1}^{n} P_{i,j} \breve{u}_j(t-1) \right) = \sum_{i=1}^{n} \breve{d}_i + \sum_{j=1}^{n} \left(\sum_{i=1}^{n} P_{i,j} \right) \breve{u}_j(t-1) \tag{3.5}$$

must be a constant for all $\breve{u}_j(t-1)$. This implies that

$$\sum_{i=1}^{n} P_{i,j} = 0, \qquad \text{for } j = 1, 2, \cdots, n. \tag{3.6}$$

By substituting the elements of $\mathbf{P}$ from (3.4) into the summation (3.6), we find that m must be related to $\breve{k}$ by

$$m = \frac{\breve{k}}{n}. \tag{3.7}$$

Substituting for m in (3.4) gives the final form for $\mathbf{P}$:

$$\mathbf{P} = \frac{\breve{k}}{n}\mathbf{B}, \tag{3.8}$$

where

$$\mathbf{B} = \underbrace{\begin{pmatrix} n-1 & -1 & -1 & \cdots & -1 \\ -1 & n-1 & -1 & \ddots & \vdots \\ -1 & \ddots & \ddots & \ddots & -1 \\ \vdots & \ddots & \ddots & n-1 & -1 \\ -1 & \cdots & -1 & -1 & n-1 \end{pmatrix}}_{n \times n}. \tag{3.9}$$

The single model parameter $\breve{k}$ does not depend on the number of actuators n.

Dimensionless Model

The model is transformed to a dimensionless form for two reasons. First, using a dimensionless model will allow the control parameters to vary little between different plants. Second, the controller is designed to produce a coating of uniform thickness and will *not* be able to change the mean coating thickness. A flow controller which maintains constant flow to the coating die is used to adjust the mean coating thickness. Therefore the non-dimensional variable $\mathbf{x}$ is chosen to represent coating thickness as a deviation from the mean.

Define $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} \breve{x}_i$ and $\bar{u}$ as the nominal gap width. The nominal gap width should be chosen well within the stable coating region. Define the following dimensionless variables:

$$x_i = \frac{\breve{x}_i - \bar{x}}{\bar{x}}, \quad u_i = \frac{\breve{u}_i - \bar{u}}{\bar{u}}, \quad d_i = \frac{\breve{d}_i - \bar{x}}{\bar{x}}, \quad n_i = \frac{\breve{n}_i}{\bar{x}}, \quad k = \frac{\breve{k}\bar{u}}{n\bar{x}}. \tag{3.10}$$

Solve the above expressions for $\breve{x}_i$, $\breve{u}_i$, $\breve{d}_i$, $\breve{n}_i$, and $\breve{k}$, substitute into (3.1), and rearrange to give the dimensionless model:

$$\mathbf{x}(t) = k\mathbf{B}\mathbf{u}(t-1) + \mathbf{d} + \mathbf{n}(t). \tag{3.11}$$

Model for Control Design

The matrix $\mathbf{B}$ in (3.9) is singular. This is because the coating thicknesses $\mathbf{x}$ are not uniquely determined by the gap widths $\mathbf{u}$. Any increment in gap width added to all the gap widths u_i does not change the coating thicknesses. However, to keep a stable film, the dimensionless gap widths $\mathbf{u}$ must not stray too far from the preferred position of $\mathbf{0}$. We augment the model with the additional equation $\sum_{i=1}^{n} u_i = 0$ to both keep $\mathbf{u}$ from straying and to give a unique mapping of the coating thicknesses to the gap widths. This is done as follows:

- Add a component to $\mathbf{x}$, $\mathbf{d}$, and $\mathbf{n}$, and set this component to zero, i.e., $x_{n+1} = n_{n+1} = d_{n+1} = 0$.
- Add a row of ones to the plant matrix $k\mathbf{B}$ to give the new $(n+1) \times n$ plant matrix $\mathbf{C} = \begin{bmatrix} k\mathbf{B} \\ 1 \cdots 1 \end{bmatrix}$.

This leads to the augmented model

$$\mathbf{x}(t) = \mathbf{C}\mathbf{u}(t-1) + \mathbf{d} + \mathbf{n}(t). \tag{3.12}$$

Since the mean value of $\mathbf{u}$ is a *free* independent variable (it does not change coating thicknesses), a controller design based on the above model which seeks to minimize $\mathbf{x}$ will automatically adjust its control action so that the mean value of $\mathbf{u}$ will be exactly zero. Also, the singularity of the original gain matrix $\mathbf{B}$ is removed; $\mathbf{C}$ has full column rank.

To derive the model predictive controller in the next section, it is convenient to express the model in terms of the changes in the inputs rather than the inputs themselves. For this purpose, we subtract Equation (3.12) for $t-1$ from that at t to arrive at

$$\mathbf{x}(t) = \mathbf{x}(t-1) + \mathbf{C}\Delta\mathbf{u}(t-1) + \Delta\mathbf{n}(t), \tag{3.13}$$

where

$$\Delta\mathbf{u}(t-1) = \mathbf{u}(t-1) - \mathbf{u}(t-2). \tag{3.14}$$

The controller calculates the inputs to the plant based on the measured variables. The model for control design is:

$$\mathbf{x}(t) = \mathbf{x}(t-1) + \mathbf{C}\Delta\mathbf{u}(t-1). \tag{3.15}$$

3.3 Estimation and Prediction

Recall that our objective for using a model is to predict the effect of changes in gap widths on the coating thicknesses. This will allow us to find the "best" adjustments in gap widths to reject disturbances.

State Estimation - Filter

The state estimator is most conveniently expressed in the following two-step form [74, 73]:

Model Prediction:

$$\mathbf{x}(t|t-1) = \mathbf{x}(t-1|t-1) + \mathbf{C}\Delta\mathbf{u}(t-1). \tag{3.16}$$

Correction Based on Measurements:

$$\mathbf{x}(t|t) = \mathbf{x}(t|t-1) + \gamma\left[\hat{\mathbf{x}}(t) - \mathbf{x}(t|t-1)\right], \qquad \gamma \in (0,1]. \tag{3.17}$$

The estimate of $\mathbf{x}(\cdot)$ based on measurements up to time $t-1$ is denoted by $\mathbf{x}(\cdot|t-1)$. The measurement of $\mathbf{x}$ at time t is denoted by $\hat{\mathbf{x}}(t)$. The filter parameter $\gamma \in (0,1]$ is used to filter noise and to obtain robustness to model uncertainty. The larger the measurement noise and model uncertainty, the smaller γ should be chosen.

By substituting (3.16) into (3.17) we obtain the *state estimator*

$$\mathbf{x}(t|t) = (1-\gamma)\left[\mathbf{x}(t-1|t-1) + \mathbf{C}\Delta\mathbf{u}(t-1)\right] + \gamma\hat{\mathbf{x}}(t), \tag{3.18}$$

which allows one to compute the current state estimate $\mathbf{x}(t|t)$ based on the previous estimate $\mathbf{x}(t-1|t-1)$, the previous input move $\Delta\mathbf{u}(t-1)$, and the current measurement $\hat{\mathbf{x}}(t)$. The state estimator is initialized with $\mathbf{x}(0|0) = \hat{\mathbf{x}}(0)$.

The state estimator (3.18) suggests that $\mathbf{x}(t|t)$ is a filtered version of $\hat{\mathbf{x}}$. Indeed, in a noise-free system with the manipulated variables constant, we have

$$\mathbf{x}(t|t) = (1-\gamma)\mathbf{x}(t-1|t-1) + \gamma\hat{\mathbf{x}}(t), \tag{3.19}$$

which shows that the state estimate $\mathbf{x}(t|t)$ is $\hat{\mathbf{x}}$ passed through a first-order filter. If the output $\hat{\mathbf{x}}$ suddenly changes to a constant value then the state estimate $\mathbf{x}(t|t)$

approaches the true value $\hat{\mathbf{x}}$ with the filter time constant:

$$\tau = \frac{t_s}{-\log(1-\gamma)}, \tag{3.20}$$

where t_s is the time between sampling instances [74, 73].

Prediction

The control algorithm prescribes the gap widths $\mathbf{u}$ which reject disturbances in $\mathbf{x}$. In order for the control algorithm to determine the "best" current gap widths there has to be a means for *predicting* the effect of the gap widths on the future coating thicknesses $\mathbf{x}$. The predictor is given by writing (3.16) for the next time step $t+1$:

$$\mathbf{x}(t+1|t) = \mathbf{x}(t|t) + \mathbf{C}\Delta\mathbf{u}(t). \tag{3.21}$$

3.4 Control

We begin by stating the unconstrained control objective. We derive the unconstrained controller that minimizes this objective. Then we discuss three methods of modifying this controller to handle actuator constraints, in our case constraints in adjacent gap widths.

Unconstrained Control Algorithm

Performance Criterion The performance criterion is to minimize the quadratic objective

$$z = \|\mathbf{x}(t+1|t)\|^2, \tag{3.22}$$

where $\|\cdot\|$ represents the Euclidean norm, $\|x\|^2 = \sum_{i=1}^{n} x_i^2$.

Unconstrained Control Problem We express the control problem as an optimization by combining the objective (3.22) with the predictor (3.21):

$$\min_{\Delta \mathbf{u}(t)} \|\mathbf{x}(t+1|t)\|^2, \qquad \text{where } \mathbf{x}(t+1|t) = \mathbf{x}(t|t) + \mathbf{C}\Delta\mathbf{u}(t). \tag{3.23}$$

The least-squares solution to the unconstrained control problem is

$$\Delta\mathbf{u}(t) = -(\mathbf{C}^T\mathbf{C})^{-1}\mathbf{C}^T\mathbf{x}(t|t). \tag{3.24}$$

Methods for Handling Actuator Constraints

Excessive stresses in the die constrain adjacent actuator positions. We will consider two ways of specifying these constraints. First, the specification could be that the difference between adjacent actuator positions is limited, i.e.,

$$|\delta u_i| = |u_{i+1} - u_i| \leq |\delta u|_{max}, \qquad \text{for } i = 1, \ldots, n-1. \tag{3.25}$$

An additional specification could be that the difference between adjacent actuator positions must be even less when large adjacent gap differences are made in opposite directions. This constraint can be written as

$$|\delta^2 u_i| = |u_{i+2} - 2u_{i+1} + u_i| \leq |\delta^2 u|_{max}, \qquad \text{for } i = 1, \ldots, n-2. \tag{3.26}$$

For those plants where $|\delta^2 u|_{max} \geq 2|\delta u|_{max}$, the first constraint (3.25) implies the second constraint (3.26), so for these plants the second constraint need not be considered.

Constraint-handling will be needed when the disturbances are sufficiently large and have sharp *spatial* variations across the substrate. When the disturbances are uniform across the substrate, then the control action calculated from the uncon-

strained control algorithm will be uniform, and constraint-handling is not needed.

Actuator constraints can be handled in three ways: by including additional terms in the objective function, by adding the constraints explicitly to the control algorithm, or by scaling the control actions to be "feasible," i.e., to satisfy the constraints. Below we describe each method of handling actuator constraints. We will choose the simplest yet effective constraint-handling method for our control problem.

Additional Terms in the Objective Function

Additional terms weighting $|u_{i+1} - u_i|$ and $|u_{i+2} - 2u_{i+1} + u_i|$ could be added to the objective function (3.22), i.e.,

$$z = \|\mathbf{x}(t+1|t)\|^2 + \beta_1 \sum_{i=1}^{n-1} |u_{i+1} - u_i|^2 + \beta_2 \sum_{i=1}^{n-2} |u_{i+2} - 2u_{i+1} + u_i|^2. \tag{3.27}$$

The disadvantage of this approach is that the added weighted terms *always* affect the control action. The weights for these terms must be large enough to keep the control action feasible for disturbances which contain sharp spatial variations, but large weights on the control action will substantially slow the control action when the disturbances are uniform across the substrate and the extra terms are not needed.

Explicitly Adding Constraints to the Control Algorithm

The constraints could be added explicitly to the control algorithm. Then the constrained control problem will be the unconstrained control problem (3.23) plus the additional constraints (3.25) and (3.26):

$$\min_{\Delta\mathbf{u}(t)} \|\mathbf{x}(t+1|t)\|^2, \tag{3.28}$$

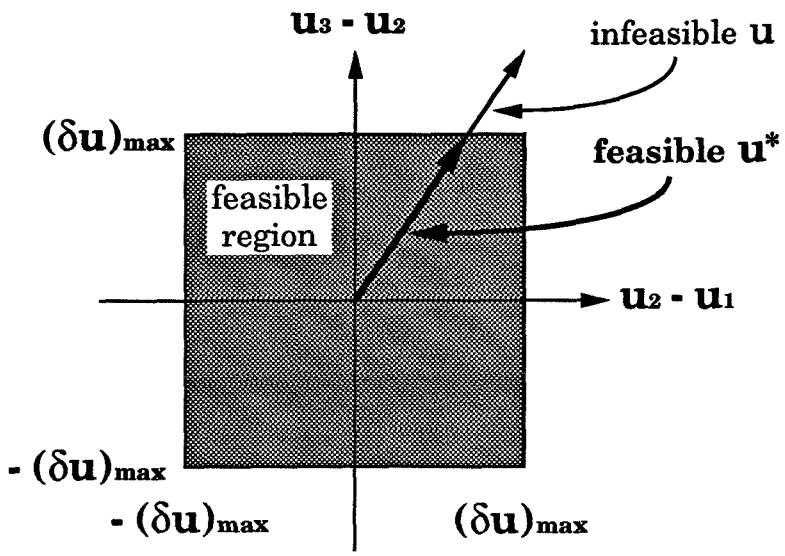


Figure 3.2: Projection of an infeasible control action to the feasible space.

$$\begin{aligned} \text{such that} \quad & \mathbf{x}(t+1|t) = \mathbf{x}(t|t) + \mathbf{C}\Delta\mathbf{u}(t) \\ & |\delta u_i| = |u_{i+1} - u_i| \leq |\delta u|_{max}, && \text{for } i = 1, \ldots, n-1. \\ & |\delta^2 u_i| = |u_{i+2} - 2u_{i+1} + u_i| \leq |\delta^2 u|_{max}, && \text{for } i = 1, \ldots, n-2. \end{aligned} \tag{3.29}$$

This is a quadratic programming problem that must be solved at each time step for the optimal actuator movements $\Delta\mathbf{u}(t)$. This approach is not as simple to implement and analyze as the third constraint-handling method discussed next.

Scaling Control Actions

Constraints can be handled by projecting any infeasible $\mathbf{u}$ given by the unconstrained control law (3.24) to the feasible space. Fig. 3.2 illustrates this idea for the first constraint (3.25) for $n = 3$. All feasible control actions $\mathbf{u}$ are given by the shaded region. When the unconstrained control law (3.24) suggests an infeasible control action, a feasible control action is found by projecting $\mathbf{u}$ to the feasible space. Many projections could be used, but the projection shown (which involves simple scaling of the control action) maintains the direction of the control action, which can be

important for multivariable systems [16].

Now consider satisfying the first constraint (3.25) for general n. This is done by scaling the control action $\mathbf{u}$ calculated from the unconstrained control law (3.24):

$$\mathbf{u}^*(t) = \begin{cases} \mathbf{u}(t) & \max_i |\delta u_i(t)| \leq |\delta u|_{max} \\ \frac{|\delta u|_{max}}{\max_i |\delta u_i(t)|}\mathbf{u}(t) & \max_i |\delta u_i(t)| > |\delta u|_{max}. \end{cases} \tag{3.30}$$

In addition, the control action from the above equation can be scaled to satisfy the second constraint (3.26):

$$\mathbf{u}^\dagger(t) = \begin{cases} \mathbf{u}^*(t) & \max_i |\delta^2 u_i^*(t)| \leq |\delta^2 u|_{max} \\ \frac{|\delta^2 u|_{max}}{\max_i |\delta^2 u_i^*(t)|}\mathbf{u}^*(t) & \max_i |\delta^2 u_i^*(t)| > |\delta^2 u|_{max}. \end{cases} \tag{3.31}$$

Thus $\mathbf{u}^\dagger$ satisfies both constraints (3.25) and (3.26).

This constraint-handling method is easy to implement and performs exactly as the unconstrained algorithm when constraint handling is not needed. It is shown in [12] that, provided the assumptions in Section 2 hold, the scaling method performs nearly as well as explicitly adding the constraints to the control algorithm.

Constrained Control Algorithm In summary, the constrained control algorithm is:

- Calculate the estimated state through (3.18).
- Calculate the unconstrained control move from (3.24).
- Scale the unconstrained control move using (3.30) and (3.31) to obtain the constrained control move which is implemented. The state estimator for the next step (3.18) will use the *constrained* implemented move from the previous step.

3.5 Limits of Performance

We would like to know how well the controller can be expected to reject disturbances in coating thicknesses. This leads us to study the various factors that *limit* the achievable closed loop performance. Knowledge of how these limitations affect the performance can show us how to modify the plant to improve the uniformity of the coating process. Also, because identification of model parameters is time-consuming and costly, we study how accurate the identification must be to achieve a given level of performance. We would also like to compare the performance of our control algorithm to the best closed loop performance achievable by any control algorithm. This allows us to convince ourselves that we have indeed designed the best possible controller.

We begin by making the assumptions necessary to achieve perfect one-step rejection of disturbances. This provides a standard to which the various limitations on the closed loop performance can be compared.

Perfect Control We are interested in the ability of the controller to reject slow disturbances. Let us study the rejection of a steady-state disturbance and let the control algorithm start at $t = 0$. For simplicity of presentation, let the disturbance $\mathbf{d}$ have zero-mean and the initial gap widths $\mathbf{u}(-1) = 0$. If we make the following three assumptions:

1. no actuator constraints,

2. no measurement noise, and

3. our model is exactly equal to our plant,

then it can be shown that the control algorithm with $\gamma = 1$ perfectly rejects the steady-state disturbance in one step.

We will drop the assumptions of no actuator constraints, no measurement noise, and no model uncertainty in turn and show how each of these prevent the controller

from rejecting the steady-state disturbance in one step. We will also investigate if the plant condition number limits performance.

Constraints on Actuator Movements

The constraints on the actuator positions will degrade performance only when the control move from the unconstrained algorithm must be scaled to keep the gap widths feasible. It can be shown that in this case the coating thicknesses at the next time $\mathbf{x}(1)$ do not equal zero. We will also show below that the coating thicknesses $\mathbf{x}$ may never reach zero.

Assume no measurement noise, $\gamma = 1$, that the model is perfect, and for simplicity of presentation that $\mathbf{d}$ has zero mean and the initial gap widths $\mathbf{u}(-1) = 0$. Then the measured coating thicknesses at $t = 0$ is $\hat{\mathbf{x}}(0) = \mathbf{x}(0) = \mathbf{d}$. The control move for the first step from (3.24) is

$$\mathbf{u}(0) = -(\mathbf{C}^T\mathbf{C})^{-1}\mathbf{C}^T\mathbf{d}. \tag{3.32}$$

If the control move from the unconstrained algorithm must be scaled to keep the gap widths feasible, the constrained control move is

$$\mathbf{u}^\dagger(0) = -\lambda(\mathbf{C}^T\mathbf{C})^{-1}\mathbf{C}^T\mathbf{d}, \tag{3.33}$$

where $0 < \lambda < 1$. If the operator implements the control move $\mathbf{u}^\dagger(0)$ exactly and there is no measurement noise, then applying the control move to the plant (3.12) gives that (after some matrix manipulation)

$$\mathbf{x}(1) = (1 - \lambda)\mathbf{d}. \tag{3.34}$$

We see that the effect of the disturbance has been diminished by a factor of $1 - \lambda$.

It can be shown that under the given assumptions, the control move will not change, and the coating thicknesses will continue to be $\mathbf{x}(t) = \mathbf{x}(1) = (1 - \lambda)\mathbf{d}$.

The constraints on gap widths prevent the steady-state disturbance from being completely rejected. This is true regardless of the control algorithm used.

Plant Modifications to Improve Performance The gap widths are constrained to prevent high stresses in the die. A die can be designed to have weaker constraints on its die gap widths by either placing the bolts further apart, by making the die lip thinner, or by making the die out of a more flexible metal. Putting the die bolts too far apart leads to strips of uncontrolled coating thickness between the die bolts. Machining a die to tight tolerances becomes increasingly difficult as the die metal becomes thinner or more flexible.

Measurement Noise

Measurement noise always limits performance. A noise filter is used to diminish the effects of noise. Because increased noise filtering also slows the controller response time, there is a tradeoff between improved coating uniformity and slower response times. We now define a measure of coating uniformity and study this tradeoff in more detail.

Consider the closed loop system with a perfect model without disturbances, only measurement noise. For a stabilizing controller, the expected value for the estimated state $x(t|t)$ is zero. The estimated state will not exactly equal zero because the controller will treat the measurement noise as a disturbance and will try to reject it. Thus the estimated state will have some variance depending on the size of the noise. The variance of the estimated state $x(t|t)$ is an appropriate measure of the uniformity of the coating. For simplicity of presentation, assume a perfect model and that the noise at each gap position is equal—dropping these assumptions only slightly affects

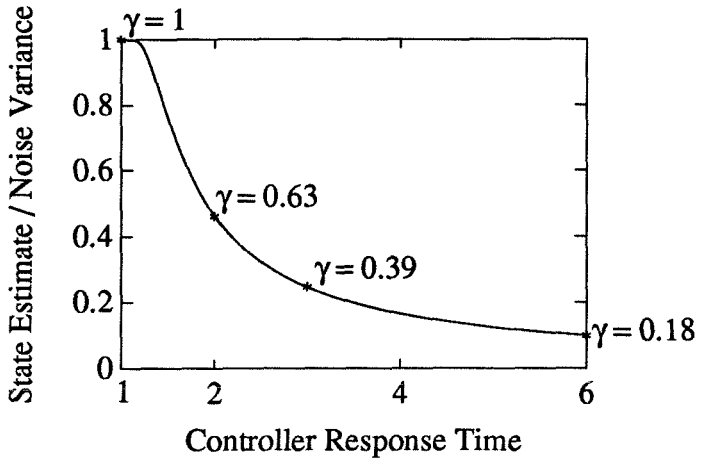


Figure 3.3: Relationship between coating uniformity and controller response time.

the following. Then it can be shown that

$$\text{Variance}(x_i) = \frac{\gamma}{2-\gamma}\text{Variance}(n_i) \qquad \text{for } i = 1, \ldots, n. \tag{3.35}$$

A measure of the controller's speed of response is the filter time constant plus 1, i.e., $\tau + 1$ (The '1' accounts for the delay through the plant).

Both Variance(x_i) and τ [through (3.20)] are functions of the noise filter parameter γ. Fig. 3.3 compares the controller response time versus the ratio of the variance of the state estimate to the measurement noise variance for different values of γ. A small amount of filtering ($\gamma \rightarrow 1$) corresponds to fast response times, but poor coating uniformity. A large amount of filtering corresponds to good coating uniformity, but with slow response times.

Plant Modifications to Improve Performance Ways to decrease the sensor noise should be investigated. The cables to the sensor should be shielded adequately to keep the sensor noise as small as possible. The effect of air currents can be diminished by decreasing the distance between the sensor and the coated substrate. The vibration of the substrate and the sensor should be minimized. Of course, an accurate sensor

reading requires a stable film.

Model Uncertainty

Model uncertainty refers to the mismatch between the model and the plant. The error between the true behavior of the physical process and that predicted by the model can significantly affect the ability of the control system to perform adequately. Controllers that are insensitive to model uncertainty are said to be *robust.* Below we quantify the effect of uncertainty. More specifically, we show that the control algorithm proposed in this article is robust to gain uncertainty. Also, we will analyze the robustness as a function of the filter parameter γ to determine the effect of the noise filter on robustness.

Uncertainty in Gain Matrix The closed loop stability can be analyzed from the state-space equation for the closed loop system. A system will be considered stable when the effect of small disturbances remains small. A system is considered unstable when the effect of small disturbances grows until the constraints (3.25) and (3.26) are reached. The effect of disturbances will never grow unbounded because the constraints (3.25), (3.26), and $\sum_{i=1}^{n} u_i = 0$ hold, which bound the magnitude of the control action.

Let the measurement be described in terms of the real plant:

$$\hat{\mathbf{x}}(t) = \mathbf{C_r}\mathbf{u}(t-1) + \mathbf{v_r}(t). \tag{3.36}$$

No assumptions are made on the unmeasured inputs $\mathbf{v_r}$.

Define $\mathbf{\Gamma}$ by

$$\mathbf{\Gamma} = -(\mathbf{C}^T\mathbf{C})^{-1}\mathbf{C}^T. \tag{3.37}$$

Then the control law (3.24) is given by

$$\mathbf{u}(t) = \mathbf{u}(t-1) + \mathbf{\Gamma x}(t|t). \tag{3.38}$$

Substitute $\hat{\mathbf{x}}(t)$ and $\mathbf{u}(t-2)$ from (3.36) and (3.38) into (3.18) and rearrange to give

$$\mathbf{x}(t|t) = (1-\gamma)(\mathbf{I} + \mathbf{C\Gamma})\mathbf{x}(t-1|t-1) + \gamma \mathbf{C_r u}(t-1) + \gamma \mathbf{v_r}(t). \tag{3.39}$$

Substitute $\mathbf{x}(t|t)$ from (3.39) into (3.38) to give

$$\mathbf{u}(t) = (1-\gamma)\mathbf{\Gamma}(\mathbf{I} + \mathbf{C\Gamma})\mathbf{x}(t-1|t-1) + (\mathbf{I} + \gamma \mathbf{\Gamma C_r})\mathbf{u}(t-1) + \gamma \mathbf{\Gamma v_r}(t). \tag{3.40}$$

Let $\mathbf{u}(t)$ be a state, then (3.39) and (3.40) give the state-space equation that defines the closed loop system,

$$\begin{bmatrix} \mathbf{x}(t|t) \\ \mathbf{u}(t) \end{bmatrix} = \begin{bmatrix} (1-\gamma)(\mathbf{I} + \mathbf{C\Gamma}) & \gamma \mathbf{C_r} \\ (1-\gamma)\mathbf{\Gamma}(\mathbf{I} + \mathbf{C\Gamma}) & \mathbf{I} + \gamma \mathbf{\Gamma C_r} \end{bmatrix} \begin{bmatrix} \mathbf{x}(t-1|t-1) \\ \mathbf{u}(t-1) \end{bmatrix} + \begin{bmatrix} \gamma I \\ \gamma \mathbf{\Gamma} \end{bmatrix} \mathbf{v_r}(t). \tag{3.41}$$

For a discrete time system, we have closed loop stability if and only if the eigenvalues of

$$\mathbf{A} = \begin{bmatrix} (1-\gamma)(\mathbf{I} + \mathbf{C\Gamma}) & \gamma \mathbf{C_r} \\ (1-\gamma)\mathbf{\Gamma}(\mathbf{I} + \mathbf{C\Gamma}) & \mathbf{I} + \gamma \mathbf{\Gamma C_r} \end{bmatrix} \tag{3.42}$$

are inside the unit circle. More specifically, the effect of disturbances will decay to zero if the spectral radius of $\mathbf{A}$ is less than one, and the effect of small disturbances will grow until the constraints are met when the spectral radius of $\mathbf{A}$ is greater than one [1].

Uncertainty in Gain This section considers uncertainty in the gain; interaction uncertainty for the Avery/Dennison pilot plant will be considered in Section 3.6. The

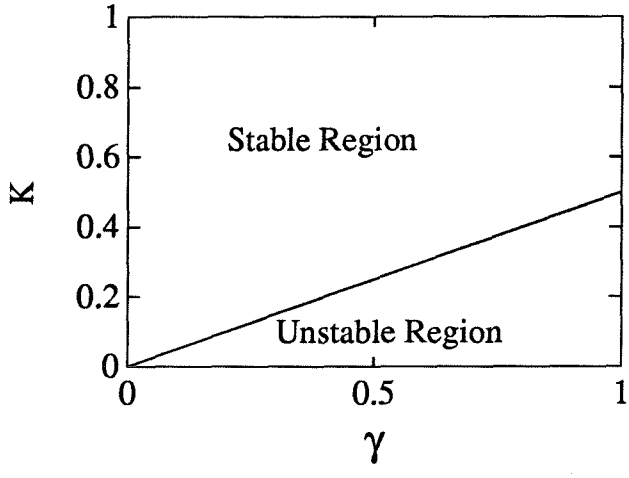


Figure 3.4: Closed loop stability as a function of γ and $K = k/k_r$, no interaction uncertainty.

real plant gain will be denoted as k_r and the augmented real plant is

$$\mathbf{C_r} = \begin{bmatrix} k_r\mathbf{B} \\ 1 \cdots 1 \end{bmatrix}. \tag{3.43}$$

Recall that k is the gain and $\mathbf{C}$ is the gain matrix for the model.

By calculating the eigenvalues of $\mathbf{A}$ in (3.42) we determine which values of the ratio $K = k/k_r$ give a stable closed loop system for each value of filter parameter γ (see Fig. 3.4). If the gain of the real plant is not underestimated by more than a factor of two ($K > 1/2$), then the closed loop system is stable. For increased filtering (smaller γ), the model gain k need not be as accurate. In other words, increased filtering adds robustness to gain uncertainty. It can be shown that the stability boundary in Fig. 3.4 is the straight line given by $k = \gamma k_r/2$.

The plant gain need not be known accurately for the closed loop system to be stable. Uncertainty in the plant gain will lead only to slower rejection of disturbances. Since we need approximate only a plant gain to design the controller, detailed identification runs are unnecessary for controller design. Any reasonable estimate will do. This makes it easier to apply the control algorithm to new cross-directional systems

when $\breve{k}$ does not change much between systems.

The Plant Condition Number

It is well-known that high condition number plants (called *ill-conditioned*) can be difficult to control [72, 102, 106]. By the condition number we mean

$$\kappa(\mathbf{C}) \equiv \frac{\overline{\sigma}(\mathbf{C})}{\underline{\sigma}(\mathbf{C})}, \tag{3.44}$$

where $\overline{\sigma}$ and $\underline{\sigma}$ denote the maximum and minimum singular values of the plant

$$\overline{\sigma}(\mathbf{C}) = \max_{\mathbf{u}\neq 0} \frac{\|\mathbf{Cu}\|_2}{\|\mathbf{u}\|_2}, \qquad \underline{\sigma}(\mathbf{C}) = \min_{\mathbf{u}\neq 0} \frac{\|\mathbf{Cu}\|_2}{\|\mathbf{u}\|_2}. \tag{3.45}$$

A plant with a high condition number is characterized by strong directionality because inputs in directions corresponding to high plant gains are strongly amplified by the plant, while inputs in directions corresponding to low plant gains are not. Thus, ill-conditioned plants may be sensitive to actuator uncertainty [106].

Recall from Section 3.2 that $\mathbf{C} = \begin{bmatrix} k\mathbf{B} \\ 1\cdots 1 \end{bmatrix}$. The last row of $\mathbf{C}$ was augmented to the plant matrix $k\mathbf{B}$ to keep $\mathbf{u}$ from straying from zero. The elements of the last row of $\mathbf{C}$ need not be 1's—the last row can be any constant multiplied by a row of 1's. Because the controllability of the process is not dependent on what scalar is used in the last row of $\mathbf{C}$, a true measure of the controllability of the process must be independent of this scalar. A "true" measure of the controllability of the plant can be defined as

$$\kappa^*(\mathbf{C}) \equiv \inf_s \kappa\left(\begin{bmatrix} k\mathbf{B} \\ s\cdots s \end{bmatrix}\right). \tag{3.46}$$

It can be proven using the theory of circulant matrices [23, 52] that $\kappa^*(\mathbf{C}) = 1$ for all n (the s that minimizes the condition number in (3.46) is $s = \sqrt{n}$). This means that ill-conditioning is not a serious problem for cross-directional processes of

die width	0.35	-	2.5 m
die bolt spacing	30	-	60 mm
coating thickness	10	-	60 μm
coating weight	15	-	50 g/m^2
substrate speed	0.5	-	6 m/s

Table 3.1: Typical ranges of physical parameters for adhesive coaters.

the type studied here.

3.6 Application to Avery/Dennison Pilot Coater

The control algorithm of Section 4 is applied to a pilot plant coater at Avery/Dennison Research Center[1] (see Fig. 3.1). Typical ranges of physical parameters for such coaters are given in Table 3.1.

First the model is identified and the model assumptions are justified based on input-output data. Then the effect of interaction uncertainty on the stability of the closed loop system was investigated using the model fit to the pilot plant data. This was done to ensure that uncertainty in the interactions [i.e., deviations from the structure implied by (3.4)] would not cause the controller to perform poorly. We then demonstrate that the controller can be effectively tuned on-line. We conclude the section with an experimental closed loop test of the controller.

Identification

For the pilot plant, the number of actuators $n = 12$. The plant gain k was fitted by least-squares from fifty input-output data sets. In Fig. 3.5 the predicted coating thicknesses are compared with experimental data for a typical input.

[1] All figures and data are given in terms of dimensionless variables for proprietary reasons.

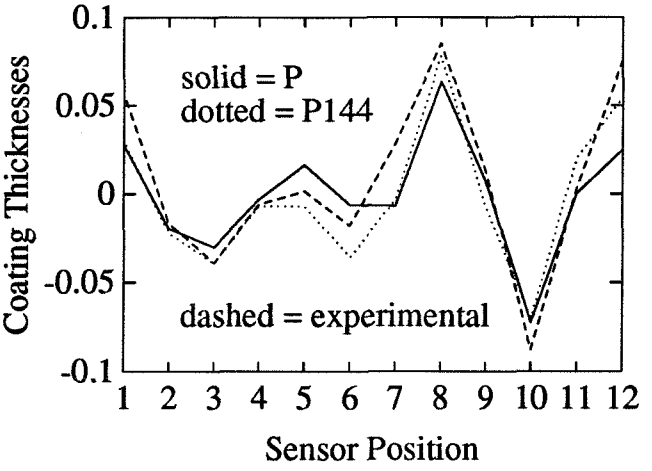


Figure 3.5: Comparison of coating thicknesses predicted by $\mathbf{P}$ and $\mathbf{P}_{144}$ with experimental data.

To test the assumptions used to develop the form of the gain matrix $\mathbf{P}$ in Section 3.2, we fitted the entire 12×12 gain matrix in (3.1) to estimate a total of 144 parameters—we denote this matrix by $\mathbf{P}_{144}$. As shown in Fig. 3.5, this model gives little improvement over the gain matrix $\mathbf{P}$ satisfying the assumptions, so the assumptions on $\mathbf{P}$ are valid.

The die had been designed to give a small interaction between nearest-neighbor positions. Assumption 3 in Section 3.2 would not have been justified if the spacing between the actuators had been much smaller.

Robustness to Interaction Uncertainty

The effect of interaction uncertainty on the stability of the closed loop system was investigated using the model fit to the pilot plant data. This was done to ensure that uncertainty in the interactions would not cause the controller to perform poorly. The same procedure as in Section 3.5 was used, but with $\mathbf{C_r} = \begin{pmatrix} \mathbf{P}_{144} \\ 1 \cdots 1 \end{pmatrix}$ for the real plant and $\mathbf{C} = \begin{pmatrix} k\mathbf{B} \\ 1 \cdots 1 \end{pmatrix}$ for the model. Fig. 3.6 shows the stable region as a function of the normalized model gain $K = k/k_r$ where k_r denotes the best fit gain

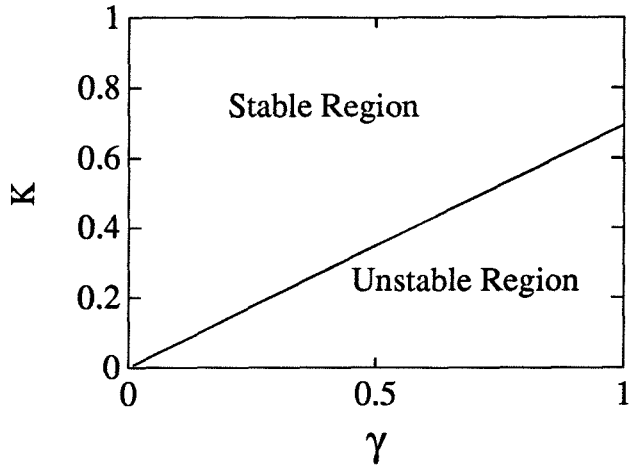


Figure 3.6: Closed loop stability as a function of γ and $K = k/k_r$. Interaction uncertainty was included through the use of $\mathbf{P}_{144}$.

in C. As in Fig. 3.4, the boundary between the stable and unstable regions is a straight line, but the slope in Fig. 3.6 is steeper. Introducing interaction uncertainty decreases the stable region, but an accurate estimate of k is still not required. This will be experimentally verified below.

Experimental Closed Loop Control

The main purpose of the experiments was to verify that detailed identification of k is not required for the resulting controller to give good performance. This is important because gathering detailed input-output data is expensive.

All the die gaps were set equal to their nominal value. Because of imperfections in the die and roller and inaccuracy in the die gap settings, this gives non-uniform coating thicknesses. The goal of the controller is to make the coating thicknesses uniform. This disturbance is small enough that constraint-handling was not needed. Because the number of experiments was limited, we decided to perform all experiments with a fixed γ near one. As discussed in Section 3.5, in plant operation γ would be chosen to trade off the closed loop speed of response with the variance of the coating thicknesses.

There were two major differences between the coater used for the identification experiments and the coater used for the closed loop experiments. First, the measurement noise was smaller for the second coater. Second, the coaters had different dies, so the responses with the two dies are expected to be different. A comparison of the die designs showed that the interactions are negligible for both dies but the steady-state gains k are expected to be substantially different. Because experiments are costly, our strategy was to avoid re-identifying k from open loop experiments but to perform closed loop experiments instead for a few values of k and choose the one that gives good control—effectively determining the optimal k through on-line tuning.

Fig. 3.7 shows the variance of the coating thicknesses for $k = 0.17$, 0.1, and 0.05. Since γ was chosen near 1 and the interactions were negligible, we expect a fast response when the model steady-state gain k is close to the true gain. Because the gain $k = 0.17$ identified for the previous die gave slow response, the controller gain is too small. This implies that the steady-state gain for the model is too large. The response for $k = 0.1$ also gave sluggish response. Therefore we tried a smaller k. For $k = 0.05$, the disturbance was rejected in two sampling times.

If we had perfect control and $\gamma = 1$, the disturbance would be rejected in one sampling time. If the assumptions of perfect control in Section 3.5 were satisfied with $\gamma = 0.95$ then the closed loop time constant would be $\tau + 1 = -1/\log(1 - 0.95) + 1 \approx 4/3 > 1$. Since we do not satisfy all the assumptions of perfect control, we cannot expect the disturbance to be rejected in less than two sampling times, i.e., $k = 0.05$ gives the best achievable performance. We see that k needed to design the controller was determined from only three closed loop experiments.

From Fig. 3.6 we expect that using k much less than 0.05 would give poor performance. This agrees with experiment—the control actions calculated using $k = 0.025$ were excessively large and were not implemented.

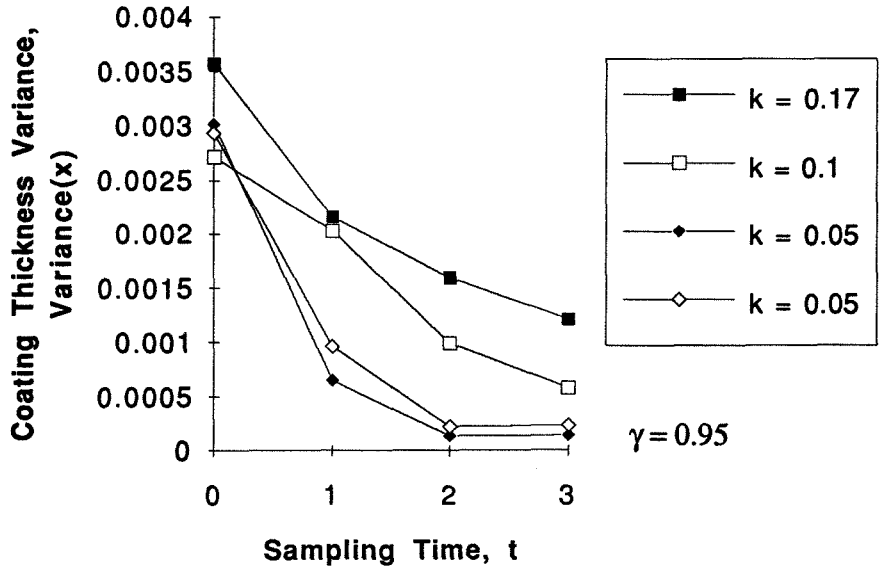


Figure 3.7: Comparison of coating thickness variances (the control actions calculated using $k = 0.025$ were excessively large and were not implemented).

Fig. 3.8 shows the closed loop response for $k = 0.05$. The disturbance was not completely rejected by the controller because of measurement noise and stiction-like effects in the die gaps.

The purpose of the next closed loop experiment was to test the closed loop performance with the controller designed above ($k = 0.05$, $\gamma = 0.95$). Fig. 3.9 shows the closed loop response (the variance of the coating thicknesses) with the designed controller to two types of disturbances. The first disturbance was caused by a roller that had a larger radius for the intermediate sensor positions than for the edge positions—this disturbance was rejected within 2 sampling times as shown in Fig. 3.9. The second disturbance was caused by ramping the roller speed and liquid flow rates (in a constant ratio) to double their values between the fourth and fifth sampling instances. The nominal gap width was kept at a constant value. We see from Fig. 3.9 that changing the roller speed and liquid flow rates in a constant ratio does not substantially affect the variance of the coating thicknesses.

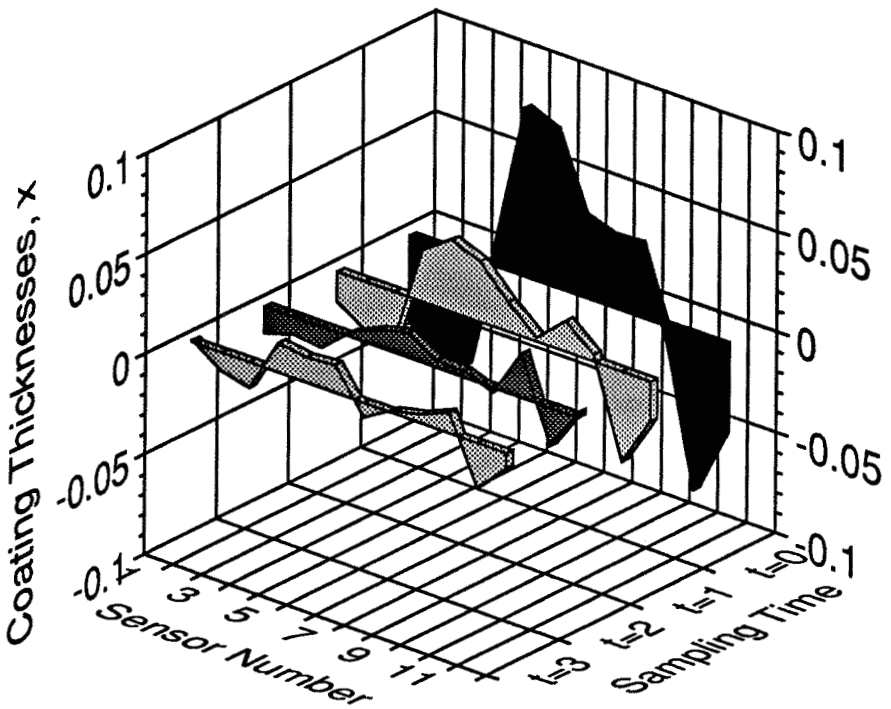


Figure 3.8: Closed loop response for $k = 0.05$.

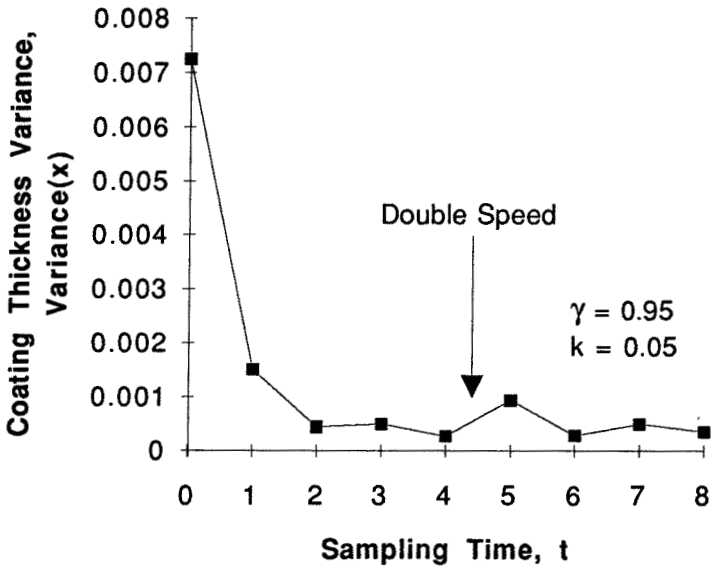


Figure 3.9: Closed loop response for two disturbances.

3.7 Discussion

The model predictive control algorithm rejects slow disturbances in coating thicknesses for a class of industrial coating processes. The control algorithm can be applied to processes other than coating, for example to the control of paper machines [61], as long as the assumptions in Section 2 are valid. The most restrictive assumption regarding the form of the plant matrix **P** is that the only interactions are due to the constant flow assumption. Additional interactions make the analysis and control much more complex. Significant interaction uncertainty makes plots such as Fig. 3.4 and 3.6 more difficult to determine and less useful. The plant condition number can become a serious limitation on closed loop performance. Laughlin et al. [61] give examples of plants with only nearest-neighbor interactions for which the condition numbers are infinity—this implies that the plants are uncontrollable.

On-line tuning becomes difficult when there are interactions—both because the controller depends on multiple model parameters and because the closed loop response can be extremely sensitive to poor estimates of the model parameters. When the plant condition number is large, an inexact estimate of the interactions can give an unstable closed loop system [106].

This chapter shows that there are strong advantages to spacing the actuators far enough apart to keep the interactions minimal. The actuators must be spaced close enough together to prevent strips of uncontrolled coating thickness between the actuators. This is how the Avery/Dennison pilot plant was designed.

3.8 Addressing the Needs of Process Control

In this chapter we addressed plant/model mismatch, the effect of plant design on control, and actuator constraints for a class of coating processes commonly found in industry. Below we relate the approaches used for handling these process control

considerations for these simple coating processes to the general approaches developed in the remainder of this thesis.

Plant/Model Mismatch The approach in this chapter to addressing model uncertainty was very simple—the controller was designed ignoring plant/model mismatch, and then shown to be relatively insensitive to the main model parameter which was the overall process gain. Identification experiments suggested that interaction uncertainty was negligible.

Interaction uncertainties are often not negligible, and process dynamics (and the associated model uncertainty) tend to be much more complicated than for the simple class of coating processes studied here. Also, in general the controller based *only* on the nominal model may perform arbitrarily poorly. For these reasons, Laughlin et al. [61] developed a method to analyze gain, interaction, and dynamic uncertainties for cross-directional processes (such as adhesive coating, paper manufacturing, plastic extrusion, and other processes with similar structure as described in [61]), and to design controllers which are robust to these uncertainties. A different approach by Hovd et al. [53, 54] does not address real parametric uncertainties, but can be used to design *robust optimal* controllers via a modified DK-iteration procedure which has much lower computation requirements than standard DK-iteration described in Chapter 2. Both the approaches of Laughlin et al. [61] and Hovd et al. [53, 54] exploit the structure of cross-directional processes (which is *symmetric circulant*, see citations for details). An approach to design controllers for processes of arbitrary structure, which satisfy a variety of performance specifications including tolerance to faults or failures in actuators and sensors, is developed in Chapters 4 and 5.

Interaction Between Design and Control This chapter included a detailed study of the potential limitations on the achievable performance posed by actuator constraints, measurement noise, model uncertainty, and the plant condition number.

This information provided recommendations on how to modify the *plant design* to improve the closed loop control. In general, information on potential limitations to the achievable performance can be used to choose actuators and sensors for control purposes, and to choose pairing and partitioning of inputs and outputs for the design of decentralized controllers. A unified approach to studying this problem is presented in Chapter 6.

Actuator Constraints Three constraint-handling methods were explored in this chapter: including additional terms in the objective function, explicitly adding constraints to the control algorithm, and scaling the control actions to satisfy the constraints. The first method is taken in the structured singular value approach discussed in the previous chapter—a weight on the control action is increased for a specific disturbance until the control action avoids the actuator constraints. The disadvantage of this approach is that the resulting controller will be sluggish for small disturbances, and ineffective for disturbances of larger magnitude.

The method of explicitly adding constraints to the control algorithm is referred to as *model predictive control* (MPC). A quadratic program must be solved at each sampling instance, and off-the-shelf software is available for performing these calculations [75]. Unfortunately, MPC is computationally too complex for many industrial processes, which is part explains why MPC is typically implemented in a supervisory mode, i.e., *on top* of the regulatory control systems. Two additional disadvantages are that some operational requirements are impossible to express through a single objective function, and the stability and performance analysis with the resulting nonlinear controller is difficult.

The third approach of scaling the control actions to satisfy the constraints, while maintaining the direction of the unconstrained control move, is referred to as *directionality compensation.* For the industrial scale adhesive coater studied in this

chapter, directionality compensation was found to perform nearly as well as model predictive control. A detailed discussion of the importance of applying directionality compensation to otherwise linear controllers, especially when the controller is an inverse-based design, is provided by Campo [16]. Analyzing the stability and performance of systems under directionality compensation is studied in Chapter 7.

The analysis in this chapter shows that many important process control considerations can be addressed, at least for simple processes. The chapters which follow develop a general approach for addressing these practical control considerations.

3.9 Conclusions

A model predictive control algorithm was presented which rejects slow disturbances in coating thicknesses for a class of industrial coating processes. An industrial scale adhesive coater was rigorously shown to be in this class. The overall plant gain was determined on-line, and the resulting controller rejected disturbances within two sampling times. The proposed controller can reduce the variance in coating thickness by as much as 80% compared to what is possible by manual control or simple control schemes.

Because this class of processes was particularly simple, many important process control considerations (such as the effect of design on control, and how to design low order controllers and handle actuator constraints and real parametric uncertainty) could easily be addressed which would have been much more difficult to address in general. This motivates the development of a general approach for addressing these control considerations.

Part III

A Unified Approach to Process Control

Chapter 4

Loopshaping for Robust Performance

Summary

Robust performance is said to be achieved if the performance specifications are met for all plants in a specified set. Classical loopshaping was developed decades ago by Bode to design for robust performance for single loop systems, where the uncertainty can be represented as a single complex Δ-block, and the sole performance specification is an upper bound on the closed loop sensitivity. Uncertainty and performance specifications are often not so simple—control problems may involve multiple performance specifications, and uncertainty may be more conveniently described as real parameter variations. Also, it is important for multivariable systems that uncertainty may be present at different locations, for example actuator uncertainty is located at the input of the plant whereas sensor uncertainty is located at the output of the plant. In this work classical loopshaping is extended to multiple parametric and unmodeled dynamic uncertainty descriptions, more general performance specifications, and to the design of decentralized controllers. The authors refer to this more general loopshaping technique as *robust loopshaping.*

4.1 Introduction

Loopshaping involves directly specifying a transfer function that parametrizes the controller based on magnitude bounds on the transfer function. These bounds are either necessary conditions or sufficient conditions so that the closed loop system satisfies desired stability and performance specifications. Examples of transfer functions that parametrize the controller include the sensitivity $S = (I + PK)^{-1}$, the complementary sensitivity $H = PK(I + PK)^{-1}$, and the open loop transfer function $L = PK$. The controller K is then calculated from the specified transfer function.

Robust performance is said to be achieved if the performance specifications are met for all plants in a specified set. Controller design methods can be classified as being either optimization methods, or not. The optimization approach involves minimizing an objective function over the set of stabilizing controllers. The optimization objective for robust control is to minimize the robust performance measure μ over the set of all stabilizing controllers, where μ is a function of the nominal model, the controller, the model uncertainty, and the performance specifications. How to solve this optimization problem for centralized controllers is an open question—the ad-hoc "DK-iteration" method proposed by Doyle [31] is the only method of tackling the optimization to date. The DK-iteration method assumes that all uncertainties are complex, involves iterative optimization, has many fragile steps, and produces high order controllers. The DK-iteration method cannot be used effectively to design decentralized controllers, or controllers that are tolerant to faults or failures in actuators or sensors.

Loopshaping can be classified as a non-optimization approach. The advantages of loopshaping over optimization approaches are that: 1) the controller can be kept simple, 2) decentralized controllers can be designed, and 3) the properties of interest to the engineer are often directly in terms of the designed loopshape.

The technique of loopshaping was introduced by Bode [7] for the design of feedback

amplifiers. Doyle et al. [32] review classical loopshaping, where the system is single-input single-output (SISO), the uncertainty can be represented as a single complex Δ-block, and the sole performance specification is an upper bound on the closed loop sensitivity. Uncertainty and performance specifications are often not so simple—control problems may involve multiple performance specifications, and uncertainty may be more conveniently described as real parameter variations. Also, it is important for multivariable systems that uncertainty may be present at different locations, for example actuator uncertainty is located at the input of the plant whereas sensor uncertainty is located at the output of the plant.

In this work classical loopshaping is extended to multiple parametric and unmodeled dynamic uncertainty descriptions, more general performance specifications, and to the design of decentralized controllers. The authors refer to this more general loopshaping technique as *robust loopshaping*. Robust loopshaping can handle performance specifications that are not addressed directly through "DK-iteration", such as gain and phase margins or fault and failure tolerance requirements.

Organization The remainder of this chapter is organized as follows. First we give formulas which parametrize several common transfer functions in terms of the controller. Second, we derive the robust loopshaping bounds. We discuss how to calculate these bounds, and how to use these bounds to design robust controllers. Robust loopshaping is shown to agree with and extend the original classical loopshaping bounds derived by Bode [7] when applied to SISO systems. Then we show how to include gain and phase margin specifications, and to handle multiple performance specifications.

4.2 Parametrize Controller in Terms of T

Robust performance is described in detail in Chapter 2. In this section we show how to parametrize the controller in terms of a transfer function of interest T. For exam-

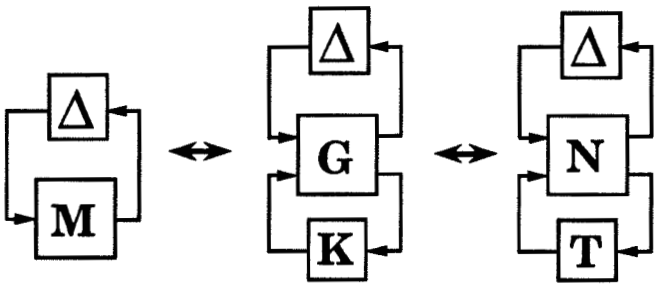


Figure 4.1: Equivalent representations of system M with perturbation Δ. The transfer function T is chosen to be a parametrization of the controller K.

ple, T could be the *complementary sensitivity* $H = PK(I + PK)^{-1}$, the *sensitivity* $S = (I+PK)^{-1}$, the open loop transfer function $L = PK$, or the controller K. Mathematically, we need to find an LFT in terms of T which describes M (see Fig. 4.1). In many cases, this is done by inspection. When this is not possible, the equations given below (which are derived in the appendix of this chapter) can be used.

To get an LFT in terms of T, begin with the interconnection structure in terms of G and K. The generalized plant G is determined by the nominal model, the location and magnitude of the uncertainties, and the performance specifications. The generalized plant G is found directly by rearranging the system's block diagram (the subroutine *sysic* does this in μ-tools [3]). We calculate N for $T = H$ (denoted as N^H) directly from G:

$$N^H = \begin{bmatrix} G_{11} & G_{12}P^{-1} \\ G_{21} & 0 \end{bmatrix}. \tag{4.1}$$

For $T = S$, L, and K, respectively, we have

$$N^S = \begin{bmatrix} G_{11} + G_{12}P^{-1}G_{21} & -G_{12}P^{-1} \\ G_{21} & 0 \end{bmatrix}, \tag{4.2}$$

$$N^L = \begin{bmatrix} G_{11} & G_{12}P^{-1} \\ G_{21} & G_{22}P^{-1} \end{bmatrix}, \tag{4.3}$$

$$N^K = G. \tag{4.4}$$

A simple program can be written that calculates N^H, N^S, N^L, and N^K, given the transfer functions describing the system components and the location of the uncertainty blocks Δ_i.

4.3 Robust Loopshaping Bounds

Controllers which satisfy robust performance can be designed via robust loopshaping. To perform robust loopshaping, the robust performance conditions are expressed as norm bounds on the transfer function T.

Consider a system in $M - \Delta$ form as shown in Fig. 4.1. The interconnection structures in Fig. 4.1 are equivalent. The closed loop transfer matrix M is written as a linear fractional transformation of the transfer function of interest, namely T. We define the set of perturbations

$$\mathbf{\Delta}_T \equiv \{\Delta_T \,|\Delta_T \text{ has the same structure as } T\}\,, \tag{4.5}$$

the set of norm-bounded perturbations

$$\gamma\mathbf{B}\mathbf{\Delta}_T \equiv \{\Delta_T \,|\Delta_T \in \mathbf{\Delta}_T, \overline{\sigma}\,(\Delta_T) \leq \gamma\}\,, \tag{4.6}$$

and its near-complement

$$\gamma\overline{\mathbf{B}\mathbf{\Delta}}_T \equiv \{\Delta_T \,|\Delta_T \in \mathbf{\Delta}_T, \overline{\sigma}\,(\Delta_T) \geq \gamma\}\,. \tag{4.7}$$

The following theorem gives a sufficient upper bound on the transfer function T for robust performance to be achieved.

Theorem 4.1 (Sufficient Upper Bound for Robust Performance [104, 105])

Let $M = F_l(N,T) = N_{11}+N_{12}T(I-N_{22}T)^{-1}N_{21}$, *let* k *be a given constant, and define*

$$f(c_T) = \max_{\Delta_T \in c_T \mathbf{B}\mathbf{\Delta}_T} \mu_\Delta(F_l(N,\Delta_T)). \tag{4.8}$$

Assume

$$\begin{array}{ll} (i) & \det(I - N_{22}T) \neq 0, \\ (ii) & f(0) = \mu_\Delta(N_{11}) < k, \text{ and} \\ (iii) & f(\infty) > k. \end{array} \tag{4.9}$$

Let c_T^{su} *solve*

$$f(c_T^{su}) = k. \tag{4.10}$$

Then $\mu_\Delta(M) < k$ *if*

$$\overline{\sigma}(T) < c_T^{su}. \tag{4.11}$$

Proof: Assumption (i) is a necessary (and sufficient) condition for the LFT $M = F_l(N,T)$ to be well-defined. In general, T will have some block structure as in (2.6), i.e. $T \in \mathbf{\Delta}_T$. If $\overline{\sigma}(T) < c_T$, then the inequality

$$\max_{\Delta_T \in c_T \mathbf{B}\mathbf{\Delta}_T} \mu_\Delta(F_l(N,\Delta_T)) > \mu_\Delta(F_l(N,T)) = \mu_\Delta(M) \tag{4.12}$$

holds because the right-hand side is an element of the set maximized on the left-hand side. (The equality cannot hold because of monotonicity and that the set maximized on the left-hand side is strictly larger than required to cover the element of the set on the right-hand side.) It follows that

$$\max_{\Delta_T \in c_T \mathbf{B}\mathbf{\Delta}_T} \mu_\Delta(F_l(N,\Delta_T)) = k \Longrightarrow \mu_\Delta(M) < k, \tag{4.13}$$

provided that there exists c_T which satisfies $f(c_T) = k$. Since the function $f(c_T)$ is monotonically nondecreasing with c_T (see Fig. 4.2), we need $f(0) < k < f(\infty)$ for

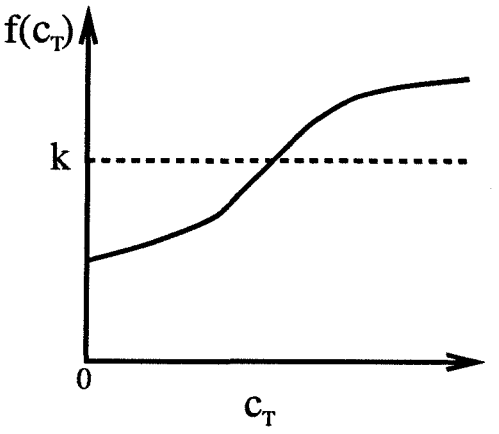


Figure 4.2: The function $f(c_T)$ is monotonically nondecreasing.

$f(c_T) = k$ to be satisfied for some positive c_T. QED.

Remark 4.1 (Interpretation of Assumptions) *Assumption* (i) *will hold for any well-posed problem. If assumption* (iii) *does not hold, then* any T *will give* $\mu_\Delta(M) < k$*—the uncertainty and performance weights would have to be very weak for this to be the case. Assumption* (ii) *may or may not hold. For reasonable choices of uncertainty and performance weights, assumption* (ii) *will hold for low frequencies when* $T = S$ *and will hold for high frequencies when* $T = H$*,* L*, or* K*. This will be illustrated in more detail later.*

The next theorem gives a sufficient lower bound for robust performance.

Theorem 4.2 (Sufficient Lower Bound for Robust Performance) *Let* $M = F_l(N,T) = N_{11} + N_{12}T(I - N_{22}T)^{-1}N_{21}$*, let* k *be a given constant, and define*

$$e(c_T) = \max_{\Delta_T \in c_T \overline{\mathbf{B}\boldsymbol{\Delta}_T}} \mu_\Delta(F_l(N, \Delta_T)). \tag{4.14}$$

Assume

$$\begin{array}{ll} (i) & \det(I - N_{22}T) \neq 0, \\ (ii') & e(0) = \max\limits_{\Delta_T \in \boldsymbol{\Delta}_T} \mu_\Delta(F_l(N, \Delta_T)) > k, \text{ and} \\ (iii') & e(\infty) < k. \end{array} \tag{4.15}$$

Let c_T^{sl} *solve*

$$e(c_T^{sl}) = k. \tag{4.16}$$

Then $\mu_\Delta(M) < k$ *if*

$$\overline{\sigma}(T) > c_T^{sl}. \tag{4.17}$$

Proof: Similar to the proof of Theorem 4.1.

Remark 4.2 (Interpretation of Assumptions) *Assumption* (i) *will hold for any well-posed problem. If assumption* (ii') *does not hold, than* any T *will give* $\mu_\Delta(M) < k$*—the performance specifications are trivially achieved in this case. For reasonable choices of uncertainty and performance weights, assumption* (iii') *will hold for low frequencies when loopshaping* K *or* L*. This bound does not exist when loopshaping with closed loop transfer functions (with reasonably chosen weights).*

Next we will derive the necessary upper and lower bounds on $\overline{\sigma}(T)$ for the transfer function T to achieve robust performance. In each case, we will begin by deriving a sufficient bound for *not* achieving robust performance. These sufficient bounds for *not* achieving robust performance are also necessary bounds *for* achieving robust performance.

Lemma 4.1 (Sufficient Upper Bound for Not Achieving Robust Performance) *Let* $M = F_l(N,T) = N_{11} + N_{12}T(I - N_{22}T)^{-1}N_{21}$*, let* k *be a given constant, and define*

$$g(c_T) \equiv \min_{\Delta_T \in c_T \overline{\mathbf{B}\mathbf{\Delta}}_T} \mu_\Delta(F_l(N, \Delta_T)). \tag{4.18}$$

Assume

$$\begin{array}{ll} (i) & \det(I - N_{22}T) \neq 0, \\ (ii'') & g(0) = \min\limits_{\Delta_T \in \mathbf{\Delta}_T} \mu_\Delta(F_l(N, \Delta_T)) < k, \text{ and} \\ (iii'') & g(\infty) > k. \end{array} \tag{4.19}$$

Let c_T^{nu} *solve*

$$g(c_T^{nu}) = k. \tag{4.20}$$

Then $\mu_\Delta(M) \geq k$ *if*

$$\overline{\sigma}(T) \geq c_T^{nu}. \tag{4.21}$$

Proof: Assumption (i) is a necessary (and sufficient) condition for the LFT $M = F_l(N,T)$ to be well-defined. Now if $\overline{\sigma}(T) \geq c_T$ then the inequality

$$\min_{\Delta_T \in c_T \overline{\mathbf{B}\mathbf{\Delta}_T}} \mu_\Delta(F_l(N,\Delta_T)) \leq \mu_\Delta(F_l(N,T)) = \mu_\Delta(M) \tag{4.22}$$

holds because the right-hand side is an element of the set minimized on the left-hand side. It follows that

$$\min_{\Delta_T \in c_T \overline{\mathbf{B}\mathbf{\Delta}_T}} \mu_\Delta(F_l(N,\Delta_T)) = k \implies \mu_\Delta(M) \geq k, \tag{4.23}$$

provided that there exists c_T which satisfies $g(c_T) = k$. Since the function $g(c_T)$ is monotonically nondecreasing with c_T, we need $g(0) < k < g(\infty)$ for $g(c_T) = k$ to be satisfied for some positive c_T. QED.

The next theorem which gives the necessary upper bound requirement for robust performance follows directly from the above lemma.

Theorem 4.3 (Necessary Upper Bound for Robust Performance) *Let* $M = F_l(N,T) = N_{11} + N_{12}T(I - N_{22}T)^{-1}N_{21}$, *let* k *be a given constant, and define*

$$g(c_T) = \min_{\Delta_T \in c_T \overline{\mathbf{B}\mathbf{\Delta}_T}} \mu_\Delta(F_l(N,\Delta_T)). \tag{4.24}$$

Assume

$$\begin{array}{ll} (i) & \det(I - N_{22}T) \neq 0, \\ (ii'') & g(0) = \min\limits_{\Delta_T \in \mathbf{\Delta}_T} \mu_\Delta(F_l(N, \Delta_T)) < k, \text{ and} \\ (iii'') & g(\infty) > k. \end{array} \tag{4.25}$$

Let c_T^{nu} *solve*

$$g(c_T^{nu}) = k. \tag{4.26}$$

Then $\mu_\Delta(M) < k$ *only if*

$$\overline{\sigma}(T) < c_T^{nu}. \tag{4.27}$$

Remark 4.3 (Interpretation of Assumptions) *Assumption* (i) *will hold for any well-posed problem. If assumption* (ii'') *does not hold, than* no T *will give* $\mu_\Delta(M) < k$*—when* T *parametrizes the controller* K*, this implies that no controller with the given structure exists that will achieve robust performance. If assumption* (iii'') *does not hold, then the optimization (4.24) is too conservative to give a useful necessary upper bound on* T*. For reasonable choices of uncertainty and performance weights, assumption* (iii'') *holds for high frequencies when loopshaping* K *or* L*. This bound exists for all frequencies when loopshaping with closed loop transfer functions (with reasonably chosen weights).*

Next we will perform a similar development to get a necessary lower bound for a transfer function to achieve robust performance. The following lemma gives a sufficient condition for not achieving robust performance.

Lemma 4.2 (Sufficient Lower Bound for Not Achieving Robust Performance) *Let* $M = F_l(N,T) = N_{11} + N_{12}T(I - N_{22}T)^{-1}N_{21}$*, let* k *be a given constant, and define*

$$h(c_T) \equiv \min_{\Delta_T \in c_T \mathbf{B}\mathbf{\Delta}_T} \mu_\Delta(F_l(N, \Delta_T)). \tag{4.28}$$

Assume

$$\begin{array}{ll} (i) & \det(I - N_{22}T) \neq 0, \\ (ii''') & h(\infty) = \min\limits_{\Delta_T \in \mathbf{\Delta}_T} \mu_\Delta(F_l(N, \Delta_T)) < k, \text{ and} \\ (iii''') & h(0) = \mu_\Delta(N_{11}) > k. \end{array} \tag{4.29}$$

Let c_T^{nl} *solve*

$$h(c_T^{nl}) = k. \tag{4.30}$$

Then $\mu_\Delta(M) \geq k$ *if*

$$\overline{\sigma}(T) \leq c_T^{nl}. \tag{4.31}$$

Proof: Similar to that of Lemma 4.1.

The next theorem follows immediately from the above lemma.

Theorem 4.4 (Necessary Lower Bound for Robust Performance) *Let* $M = F_l(N,T) = N_{11} + N_{12}T(I - N_{22}T)^{-1}N_{21}$, *let* k *be a given constant, and define*

$$h(c_T) = \min_{\Delta_T \in c_T \mathbf{B}\mathbf{\Delta}_T} \mu_\Delta(F_l(N, \Delta_T)). \tag{4.32}$$

Assume

$$\begin{array}{ll} (i) & \det(I - N_{22}T) \neq 0, \\ (ii''') & h(\infty) = \min\limits_{\Delta_T \in \mathbf{\Delta}_T} \mu_\Delta(F_l(N, \Delta_T)) < k, \text{ and} \\ (iii''') & h(0) = \mu_\Delta(N_{11}) > k. \end{array} \tag{4.33}$$

Let c_T^{nl} *solve*

$$h(c_T^{nl}) = k. \tag{4.34}$$

Then $\mu_\Delta(M) < k$ *only if*

$$\overline{\sigma}(T) > c_T^{nl}. \tag{4.35}$$

Remark 4.4 (Interpretation of Assumptions) *Assumption* (i) *will hold for any well-posed problem. If assumption* (ii''') *does not hold, then* no T *will give* $\mu_\Delta(M) < k$*—when* T *parametrizes the controller* K*, this implies that no controller with the*

given structure exists that will achieve robust performance. Assumption (iii''') *may or may not hold. For reasonable choices of uncertainty and performance weights, assumption* (iii''') *will hold for high frequencies when* $T = S$ *and will hold for low frequencies when* $T = H$, L, *or* K. *This will be illustrated in more detail later.*

General Remarks

Remark 4.5 *When the necessary upper and the sufficient upper bounds are very close to each other, we have essentially a* necessary and sufficient *upper bound for robust performance in terms of* $\overline{\sigma}(T)$. *A similar statement holds for the necessary lower and sufficient lower bounds.*

Remark 4.6 *Note that the sufficient upper bound and the necessary lower bound cannot both exist at the same frequency. Actually, when a robustly performing controller exists (assumption* (ii''') *must hold in this case), and provided that robust performance is not trivial to satisfy (so assumption* (iii) *holds),* exactly one *bound exists for each frequency. Similarly, the sufficient lower bound and the necessary upper bounds cannot both exist at the same frequency. When a robustly performing controller exists, and provided that robust performance is not trivial to satisfy,* exactly one *bound exists for each frequency.*

Remark 4.7 *Many parameterizations* T *exist for the controller* K, *for example* K *can be parameterized by the sensitivity* S, *the complementary sensitivity* H, *the open loop transfer function* $L = PK$, *or just the controller* K. *Controllers can also be designed via loopshaping the IMC filter* F *[32] or the IMC filter time constant* λ *[51]. Parametrizations for decentralized controllers are given in Chapter 5.*

Remark 4.8 *The norm bounds on different* T*'s can be combined over different frequency ranges. For example, for* $T_1 = S$ *and* $T_2 = H$, *robust performance is achieved if* either *of the conditions* $\overline{\sigma}(S(jw)) < c_S^{su}$ or $\overline{\sigma}(H(jw)) < c_H^{su}$ *is met for each* ω.

Remark 4.9 *The bounds given by each theorem are the tightest bounds possible. For example, if we have a T_1 with $\overline{\sigma}(T_1)$ larger than c_T^{su} defined by Theorem 4.1, then there exists a T_2 with $\overline{\sigma}(T_2) = \overline{\sigma}(T_1)$ where T_2 does not meet robust performance.*

Remark 4.10 *The least conservative bounds are obtained when Δ_T is a repeated scalar block. For this reason, the repeated scalar block (i.e. assuming all loops are identical) is used when designing decentralized controllers for robust performance via loopshaping. When designing controllers to have failure tolerance properties, it can be useful to allow Δ_T to consist of independent 1×1 blocks when calculating sufficient bounds for robust stability. For further details on decentralized controller design and failure/fault tolerance, see Chapter 5.*

Remark 4.11 *It is straightforward to derive alternative bounds in terms of $\underline{\sigma}(T)$ using Lemma 4.3 which we present in the next section. We will not explore this further in this thesis.*

4.4 Efficient Calculation Procedures

For loopshaping design, $\mathbf{\Delta}_T$ is repeated scalar (see Remark 4.10). Below we provide methods for calculating the necessary and the sufficient bounds for loopshaping design.

Sufficient Bounds Theorem 4.1 which gives the sufficient upper bound is a restatement of a result in [104, 105]. In fact, Skogestad and Morari [104] show that Theorem 4.1 remains valid if $f(c_T)$ is replaced by

$$\hat{f}(c_T) \equiv \mu_{\left[\begin{smallmatrix}\Delta & \\ & \Delta_T\end{smallmatrix}\right]}\left(\begin{bmatrix} N_{11} & N_{12} \\ kc_TN_{21} & kc_TN_{22} \end{bmatrix}\right). \tag{4.36}$$

Since $\hat{f}(c_T)$ is monotonic, c_T^{su} that satisfies $\hat{f}(c_T^{su}) = k$ can be found by bisection, which involves multiple μ calculations. These multiple μ calculations can be avoided by using the following explicit expression for c_T^{su}:

$$c_T^{su} = \min_{\substack{\Delta \in \mathbf{B}\boldsymbol{\Delta} \\ \Delta_T \in \gamma \mathbf{B}\boldsymbol{\Delta}_T}} \left\{ \gamma \left| \det \left(I + \begin{bmatrix} k^{-1}N_{11} & k^{-1}N_{12} \\ N_{21} & N_{22} \end{bmatrix} \begin{bmatrix} \Delta & \\ & \Delta_T \end{bmatrix} \right) = 0 \right\}. \quad (4.37)$$

The above minimization is solved exactly as for standard μ calculation except that only the subblocks in Δ_T are scaled by γ [107].

For loopshaping design the sufficient lower bound c_T^{sl} (when it exists) can be defined in terms of the same scaled μ problem as above but for a different "M" matrix. To show this, we need the following lemma:

Lemma 4.3 (LFT of Inverse) *Consider an LFT in terms of A, $F_l(N, A)$. Assume N_{22} is invertible. Define*

$$\hat{N} = \begin{bmatrix} N_{11} - N_{12}N_{22}^{-1}N_{21} & N_{12}N_{22}^{-1} \\ -N_{22}^{-1}N_{21} & N_{22}^{-1} \end{bmatrix}. \quad (4.38)$$

Then the following equality holds:

$$F_l(N, A) = F_l(\hat{N}, A^{-1}). \quad (4.39)$$

Proof: The equality follows from the definition of the lower LFT and some algebraic manipulation. QED.

Since $\boldsymbol{\Delta}_T$ is repeated scalar for loopshaping design, we have (assuming N_{22} is invertible)

$$e(c_T) \equiv \max_{\Delta_T \in c_T \overline{\mathbf{B}\boldsymbol{\Delta}_T}} \mu_{\Delta}(F_l(N, \Delta_T)) \quad (4.40)$$

$$= \max_{\delta I \in c_T \overline{\mathbf{B}\boldsymbol{\Delta}_T}} \mu_\Delta(F_l(N, \delta I)) \tag{4.41}$$

$$= \max_{(1/\delta) I \in (1/c_T)\mathbf{B}\boldsymbol{\Delta}_T} \mu_\Delta\left(F_l\left(\hat{N}, \delta^{-1} I\right)\right) \tag{4.42}$$

$$= \max_{\hat{\Delta}_T \in (1/c_T)\mathbf{B}\boldsymbol{\Delta}_T} \mu_\Delta\left(F_l\left(\hat{N}, \hat{\Delta}_T\right)\right) \tag{4.43}$$

$$\equiv f(1/c_T). \tag{4.44}$$

Thus the method for calculating the sufficient upper bound on $\overline{\sigma}(T)$ discussed above can also be used to calculate the sufficient lower bound on $\overline{\sigma}(T)$ whenever $\boldsymbol{\Delta}_T$ is repeated scalar (inspection of (4.1-4.4) shows that the assumption that N_{22} be invertible does not limit the usefulness of above transformation). This fact was used by Hovd [51] to design decentralized controllers via loopshaping the IMC filter parameter.

Necessary Bounds The necessary bounds c_T^{nu} and c_T^{nl} are defined in terms of more unfamiliar multivariable optimizations (4.18) and (4.28). Currently there exists no method to solve optimizations (4.18) and (4.28) for general block diagonal $\boldsymbol{\Delta}_T$. However, the optimizations *can* be solved for the special structure of $\boldsymbol{\Delta}_T$ used when designing SISO and fully-decentralized controllers via loopshaping.

For loopshaping design we have $\Delta_T \in \boldsymbol{\Delta}_T$, where

$$\boldsymbol{\Delta}_T = \{\delta I_r \,|\, \delta \in \mathcal{C}\}. \tag{4.45}$$

The set $\boldsymbol{\Delta}_T$ can be parametrized in terms of two parameters—the magnitude and phase of δ. We show here that the two-parameter minimization can be replaced by a one-parameter minimization. We show that both necessary bounds are roots of the *same* function calculated through the one-parameter minimization.

Using $\boldsymbol{\Delta}_T$ given by (4.45), Theorems 4.3 and 4.4 state that the upper and lower

necessary bounds c_T^{nu} and c_T^{nl} solve

$$g(c_T^{nu}) = \min_{\|\delta\| \geq c_T^{nu}} \mu_\Delta(F_l(N, \delta I_r)) = k, \tag{4.46}$$

and

$$h(c_T^{nl}) = \min_{\|\delta\| \leq c_T^{nl}} \mu_\Delta(F_l(N, \delta I_r)) = k. \tag{4.47}$$

Define

$$m(c_T) \equiv \min_{\|\delta\| = c_T} \mu_\Delta(F_l(N, \delta I_r)), \tag{4.48}$$

then it follows that

$$g(c_T) = \min_{\gamma \geq c_T} m(\gamma) \qquad h(c_T) = \min_{\gamma \leq c_T} m(\gamma) \tag{4.49}$$

$$g(c_T) \leq m(c_T) \qquad h(c_T) \leq m(c_T) \tag{4.50}$$

$$m(\infty) = g(\infty) \qquad m(0) = h(0). \tag{4.51}$$

Let assumptions (i), (ii''), and (iii'') hold so that the necessary upper bound exists. Then $m(\infty) = g(\infty) > k$ and $m(c_T) < k$ for some c_T, and we have from Theorem 4.3 that c_T^{nu} is given by

$$g(c_T^{nu}) = k \iff \min_{\gamma \geq c_T^{nu}} m(\gamma) = k \iff c_T^{nu} \text{ is the largest zero of } m(c_T) - k. \tag{4.52}$$

Let assumptions (i), (ii'''), and (iii''') hold so that the necessary lower bound exists. Then $m(0) = g(0) > k$ and $m(c_T) < k$ for some c_T, and we have from Theorem 4.4 that c_T^{nl} is given by

$$h(c_T^{nl}) = k \iff \min_{\gamma \leq c_T^{nl}} m(\gamma) = k \iff c_T^{nl} \text{ is the smallest zero of } m(c_T) - k. \tag{4.53}$$

The minimizations in (4.46) and (4.47) are over two parameters—the phase and mag-

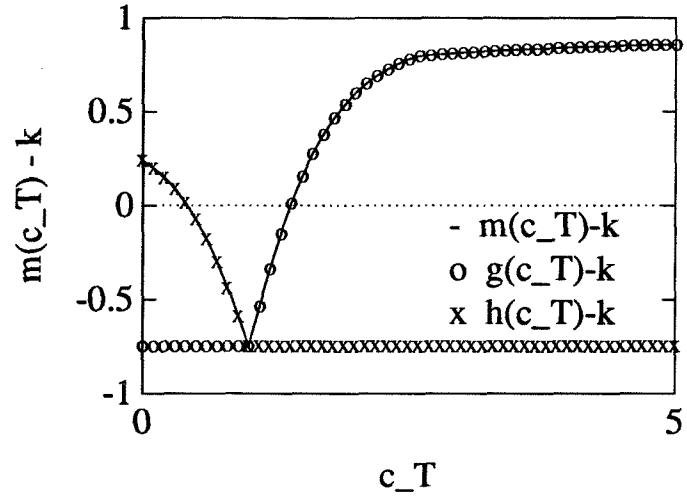


Figure 4.3: The function $m(c_T) - k$ for a given frequency. The N matrix is for loopshaping the diagonal complementary sensitivity for $\omega = \sqrt{10}$ for the high-purity distillation column in the next chapter.

nitude of δ. The necessary bounds c_T^{nu} and c_T^{nl} are solved from (4.46) and (4.47) by locating the one zero of $g(c_T) - k$ and $h(c_T) - k$, respectively. The functions $g(c_T) - k$ and $h(c_T) - k$ are monotonic, so these zeros are easy to locate via bisection.

In contrast, the minimization in (4.48) is over only one parameter—the phase of δ, which is between 0 and 2π. The necessary bounds c_T^{nu} and c_T^{nl} are solved by locating the largest and smallest zeros of the *same* function $m(c_T) - k$.

In general $m(c_T)$ is not monotonic. Fig. 4.3 is a plot of $m(c_T) - k$ used for determining c_T^{nu} and c_T^{nl} in the loopshaping design of a decentralized controller in Chapter 5. In this case, assumptions (i), (ii''), (iii''), (ii'''), and (iii''') hold, so both necessary bounds exist. We can use (4.49) to determine $g(c_T) - k$ and $h(c_T) - k$ from the plot of $m(c_T) - k$ (as shown in Fig. 4.3).

There is slight tradeoff in solving for the bounds using $m(c_T)$ instead of by the original equations (4.46) and (4.47). The roots found in (4.46) and (4.47) are of monotonic functions and so were very easy to find, for example by bisection with zero and a large c_T as interval endpoints. The function $m(c_T) - k$ can have multiple zeros. This could make finding the zeros of $m(c_T) - k$ more difficult, but in practice we have

found $m(c_T)$ to have one global minimum (for $c_T > 0$), so that $m(c_T) - k$ would have two zeros when both bounds exist and one zero when only the upper bound exists.

Remark 4.12 *Explicit expressions for c_T can be derived for the necessary bounds as was done for the sufficient upper bound. Unfortunately, it is not as clear what to do with the explicit expressions because they do not reduce to standard μ calculation as for the sufficient upper bound.*

4.5 Controller Design via Loopshaping

In robust loopshaping design, the nominal closed loop transfer functions are specified directly based on necessary bounds and sufficient bounds for robust performance. We need to satisfy separate conditions to guarantee nominal stability. For example, when designing an SISO controller via loopshaping closed loop transfer functions, nominal stability is guaranteed by specifying stable S and H and by satisfying the interpolation conditions [32]

$$H(z_i) = 0 \text{ and } S(z_i) = 1 \text{ for all closed right half plane zeros } z_i, \tag{4.54}$$

$$S(p_i) = 0 \text{ and } H(p_i) = 1 \text{ for all closed right half plane poles } p_i. \tag{4.55}$$

The interpolation conditions are equivalent to the condition that the right half plane poles and zeros of the plant cannot be canceled by the controller. These conditions are easy to satisfy when there are few right half plane poles and zeros; when there are more then the Internal Model Control (IMC) method can be used to stabilize the system, and the filter can be designed via loopshaping (for details see [76, 32]). Guaranteeing nominal stability is more difficult in the multivariable case. A detailed discussion of stability for multivariable systems is given in [114]. A loopshaping method for guaranteeing nominal stability is described in Chapter 6.

There are many advantages to designing controllers via loopshaping *closed loop* transfer functions. One advantage is that the properties of interest to the engineer are specified directly by the nominal closed loop transfer functions. For example, the sensitivity is directly related to the capability of the closed loop system to reject disturbances at the output of the plant. The complementary sensitivity is directly related to the closed loop speed of response and the insensitivity of the output to measurement noise. Thus directly specifying the closed loop transfer functions allows "intuition" in the design procedure. Also, gain and phase margins can be specified when loopshaping the sensitivity S or H, as will be shown in Section 4.8. When designing controllers via loopshaping closed loop transfer functions, robust performance can be guaranteed using sufficient bounds on the sensitivity and complementary sensitivity (as will be shown in Section 4.7). It is also shown that robust performance cannot be guaranteed using sufficient bounds on open loop transfer functions.

A simple form is usually chosen for S and H, and the controller is calculated via $K = P^{-1}HS^{-1}$. A disadvantage of designing controllers via loopshaping *closed loop* transfer functions is that in practice the order of the controller is larger than the order of the plant. It is difficult to design a controller with a specified structure when specifying closed loop transfer functions. The advantage of loopshaping design using *open loop* transfer functions (e.g. L or K) is that the controller complexity (e.g. PID, or low order) is directly specified. It is difficult to do this using other robust controller design methods. For example, the DK-iteration method proposed by Doyle [31] gives controllers of very high order, though the order can be somewhat reduced using model reduction [3].

A general advantage of loopshaping (whether loopshaping closed loop or open loop transfer functions) over other robust controller design methods is that *decentralized* controllers can be designed. Controllers can also be designed to meet specified gain and phase margins, multiple performance specifications, and failure and fault

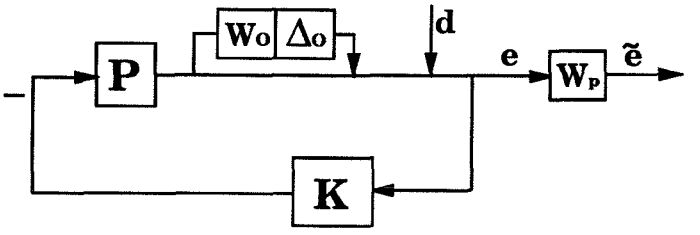


Figure 4.4: The plant with output uncertainty Δ_O of magnitude $w_O(s)$. Robust performance is satisfied if $\overline{\sigma}\left(w_P(I+\hat{P}K)^{-1}\right) \leq 1$ for all Δ_O with $\|\Delta_O\|_\infty \leq 1$.

tolerance specifications.

4.6 Robust Loopshaping Reduces to Classical Loopshaping for SISO Systems

Classical loopshaping was developed decades ago by Bode [7] to design for robust performance for single loop systems, where the uncertainty can be represented as a single complex Δ-block, and the sole performance specification is an upper bound on the closed loop sensitivity. We show below that in this case the classical loopshaping bounds can be derived via the robust loopshaping theorems. We also show the robust loopshaping bounds provide additional information which was not available from the original classical loopshaping bounds.

Loopshaping Closed Loop Transfer Functions

Assume that we are interested in disturbance attenuation, then our performance condition is to keep the norm of the sensitivity function $\overline{\sigma}(S) = |S|$ small. If we let our frequency dependent performance bound be $1/|w_P|$, then robust performance is satisfied if $\overline{\sigma}(S) < 1/|w_P|$ for all plants in our uncertainty description. Let the set of possible plants be given in terms of multiplicative uncertainty of magnitude $|w_O|$ (see Fig. 4.4). Robust performance is satisfied if and only if $\mu_\Delta(M) < 1$ for all frequencies

where

$$M = \begin{bmatrix} w_O H & w_O H \\ w_P S & w_P S \end{bmatrix}, \qquad \Delta = \begin{bmatrix} \Delta_O & \\ & \Delta_P \end{bmatrix}. \tag{4.56}$$

The generalized plant G is found from inspection to be

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix} = \left[\begin{array}{cc|c} 0 & 0 & w_O P \\ w_P & w_P & -w_P P \\ \hline 1 & 1 & -P \end{array}\right]. \tag{4.57}$$

We write M as an LFT of H either by inspection or using (4.1):

$$N_{11}^H = \begin{bmatrix} 0 & 0 \\ w_P & w_P \end{bmatrix}, N_{12}^H = \begin{bmatrix} w_O \\ -w_P \end{bmatrix}, N_{21}^H = \begin{bmatrix} 1 & 1 \end{bmatrix}, N_{22}^H = 0. \tag{4.58}$$

Sufficient Bounds Theorem 4.1 gives the sufficient upper bound on H for robust performance to be achieved. The upper bound is c_H^{su}, where c_H^{su} at each frequency solves

$$\max_{\|\Delta_H\| \le c_H^{su}} \mu_{\left[\begin{smallmatrix} \Delta_O & \\ & \Delta_P \end{smallmatrix}\right]} \left(F_l \left(\begin{bmatrix} 0 & 0 & w_O \\ w_P & w_P & -w_P \\ 1 & 1 & 0 \end{bmatrix}, \Delta_H \right) \right) = 1. \tag{4.59}$$

The structure of Δ_H is the same as for H, namely 1×1.

Theorem 4.1 gives us that $\mu_\Delta(M) < 1$ for all frequencies for which $|H|$ is lower than c_H^{su}. The minimization in (4.59) can be solved analytically (from the definition of μ) to give the following expression for c_H^{su}:

$$\mu_\Delta(M) < 1 \Longleftarrow |H| < c_H^{su} = \frac{1 - |w_P|}{|w_O| + |w_P|}. \tag{4.60}$$

Similarly, the sufficient upper bound theorem applied to the sensitivity S gives

$$\mu_\Delta(M) < 1 \Longleftarrow |S| < c_S^{su} = \frac{1 - |w_O|}{|w_O| + |w_P|}. \tag{4.61}$$

The expressions for c_H^{su} and c_S^{su} above are most easily derived from

$$\mu_\Delta(M) = \mu_{\left[\begin{smallmatrix}\Delta_O & \\ & \Delta_P\end{smallmatrix}\right]} \begin{bmatrix} w_O H & w_O H \\ w_P S & w_P S \end{bmatrix} = |w_O H| + |w_P S| \tag{4.62}$$

combined with the triangle inequality [e.g., use $|S| = |1 - H| \leq 1 + |H|$ to derive (4.60)].

Notice that the sufficient upper bound on $|H|$ is defined only for $|w_P| < 1$ and the sufficient upper bound on $|S|$ is defined only for $|w_O| < 1$. This corresponds to the requirement that $\mu_\Delta(N_{11}) < 1$ hold for the sufficient upper bound to be defined in Theorem 4.1.

For the sufficient lower bound on H to exist, we need

$$e(\infty) = \lim_{c_T \to \infty} \max_{\Delta_H \in c_T \overline{\mathbf{B}\boldsymbol{\Delta}}_T} \mu_{\left[\begin{smallmatrix}\Delta_O & \\ & \Delta_P\end{smallmatrix}\right]} \left(F_l \left(\begin{bmatrix} 0 & 0 & w_O \\ w_P & w_P & -w_P \\ 1 & 1 & 0 \end{bmatrix}, \Delta_H \right) \right) < 1. \tag{4.63}$$

Analytically solving the expression for $e(\infty)$ gives that $e(\infty) = \infty$, so from Theorem 4.2 we see that the sufficient lower bound on H does not exist. A similar development shows that the sufficient lower bound on S does not exist either.

Necessary Bounds For the necessary bounds to exist in Theorems 4.3 and 4.4, we need

$$\min_{\Delta_H} \mu_{\left[\begin{smallmatrix}\Delta_O & \\ & \Delta_P\end{smallmatrix}\right]} \left(F_l \left(\begin{bmatrix} 0 & 0 & w_O \\ w_P & w_P & -w_P \\ 1 & 1 & 0 \end{bmatrix}, \Delta_H \right) \right) < 1. \tag{4.64}$$

The above minimization can be solved analytically to give

$$\min_{\Delta_H} \mu_{\left[\begin{smallmatrix}\Delta_O & \\ & \Delta_P\end{smallmatrix}\right]} \left(F_l \left(\begin{bmatrix} 0 & 0 & w_O \\ w_P & w_P & -w_P \\ 1 & 1 & 0 \end{bmatrix}, \Delta_H \right) \right) = \min \{|w_O|, |w_P|\}. \tag{4.65}$$

This equation is most easily derived using (4.62). The left-hand minimization in (4.65) is achieved for $\Delta_H = 1$ when $|w_O| < |w_P|$ and for $\Delta_H = 0$ when $|w_O| \geq |w_P|$.

A necessary condition for robust performance is that for each frequency either $|w_O|$ or $|w_P|$ is less than one. This will be assumed in the following derivation of the necessary bounds, since if this condition is not met, then robust performance cannot be met for any controller.

Theorem 4.3 gives the tightest necessary upper bound on H for robust performance to be achieved. The upper bound is c_H^{nu}, where c_H^{nu} at each frequency solves

$$\min_{\|\Delta_H\| \geq c_H^{nu}} \mu_{\left[\begin{smallmatrix}\Delta_O & \\ & \Delta_P\end{smallmatrix}\right]} \left(F_l \left(\begin{bmatrix} 0 & 0 & w_O \\ w_P & w_P & -w_P \\ 1 & 1 & 0 \end{bmatrix}, \Delta_H \right) \right) = 1. \tag{4.66}$$

Theorem 4.3 gives us that for $\mu_\Delta(M)$ to be less than 1 it is required that $|H|$ is lower than c_H^{nu}. The minimization in (4.66) can be solved analytically to give the following expression for c_H^{nu}:

$$\mu_\Delta(M) < 1 \Longrightarrow |H| < c_H^{nu} = \begin{cases} \frac{1-|w_P|}{|w_O|-|w_P|} & \text{if } |w_O| > 1, \\ \frac{1+|w_P|}{|w_O|+|w_P|} & \text{if } |w_O| \leq 1. \end{cases} \tag{4.67}$$

Similarly, the necessary upper bound theorem applied to S gives

$$\mu_\Delta(M) < 1 \Longrightarrow |S| < c_S^{nu} = \begin{cases} \frac{1-|w_O|}{|w_P|-|w_O|} & \text{if } |w_P| > 1, \\ \frac{1+|w_O|}{|w_P|+|w_O|} & \text{if } |w_P| \leq 1. \end{cases} \tag{4.68}$$

Again the expressions for c_H^{nu} and c_S^{nu} above are most easily derived from (4.62) combined with the triangle inequality.

Notice that the necessary upper bound is defined for all frequencies provided that $\min\{|w_O|, |w_P|\} < 1$.

Theorem 4.4 gives the tightest necessary lower bound on H for robust performance

to be achieved. The upper bound is c_H^{nl}, where c_H^{nl} at each frequency solves

$$\min_{\|\Delta_H\| \le c_H^{nl}} \mu_{\left[\begin{smallmatrix}\Delta_O & \\ & \Delta_P\end{smallmatrix}\right]} \left(F_l \left(\begin{bmatrix} 0 & 0 & w_O \\ w_P & w_P & -w_P \\ 1 & 1 & 0 \end{bmatrix}, \Delta_H \right) \right) = 1. \tag{4.69}$$

Theorem 4.4 gives us that for $\mu_\Delta(M)$ to be less than 1 it is required that $|H|$ is larger than c_H^{nl}. The minimization in (4.69) can be solved analytically to give the following expression for c_H^{nl}:

$$\mu_\Delta(M) < 1 \Longrightarrow |H| > c_H^{nl} = \frac{1 - |w_P|}{|w_O| - |w_P|}. \tag{4.70}$$

Similarly, the necessary lower bound theorem applied to S gives

$$\mu_\Delta(M) < 1 \Longrightarrow |S| > c_S^{nl} = \frac{1 - |w_O|}{|w_P| - |w_O|}. \tag{4.71}$$

Again the expressions for c_H^{nl} and c_S^{nl} above are most easily derived from (4.62) combined with the triangle inequality.

Notice that the necessary lower bound on $|H|$ is defined only for $|w_P| > 1$ and the necessary lower bound on $|S|$ is defined only for $|w_O| > 1$. This corresponds to the requirement that $\mu_\Delta(N_{11}) > 1$ holds for the necessary lower bound to be defined.

Remark 4.13 *Consider the bounds on the sensitivity S. The necessary upper bound is defined for all $|w_O|$. The sufficient upper bound exists only for $|w_O| < 1$. The necessary lower bound exists only for $|w_O| > 1$. It makes sense that only one of these bounds can exist for a given frequency. It is interesting that, except for frequencies where $|w_O| = 1$, exactly one of these bounds must exist at each frequency. Similar statements hold for the complementary sensitivity H (but with w_P instead of w_O).*

Loopshaping Open Loop Transfer Functions

Now we derive loopshaping bounds on L for an SISO plant. The corresponding bounds for K are immediately given by the bounds for L, since $|L| = |P| \cdot |K|$.

To loopshape with L, we calculate N^L from G using (4.3):

$$N^L = \begin{bmatrix} G_{11} & G_{12}P^{-1} \\ G_{21} & G_{22}P^{-1} \end{bmatrix} = \begin{bmatrix} 0 & 0 & w_O \\ w_P & w_P & -w_P \\ 1 & 1 & -1 \end{bmatrix}. \tag{4.72}$$

Sufficient Bounds Theorem 4.1 gives the sufficient upper bound on L for robust performance to be achieved. The upper bound is c_L^{su}, where c_L^{su} at each frequency solves

$$\max_{\|\Delta_H\| \le c_L^{su}} \mu_{\left[\begin{smallmatrix}\Delta_O & \\ & \Delta_P\end{smallmatrix}\right]} \left(F_l \left(\begin{bmatrix} 0 & 0 & w_O \\ w_P & w_P & -w_P \\ 1 & 1 & -1 \end{bmatrix}, \Delta_L \right) \right) = 1. \tag{4.73}$$

The structure of Δ_L is the same as for L, namely 1×1.

Theorem 4.1 gives us that $\mu_\Delta(M) < 1$ for all frequencies for which $|L|$ is lower than c_L^{su}. The minimization in (4.73) can be solved analytically to give the following expression for c_L^{su}:

$$\mu_\Delta(M) < 1 \Longleftarrow |L| < c_L^{su} = \frac{1 - |w_P|}{1 + |w_O|}. \tag{4.74}$$

Similarly, the sufficient lower bound theorem gives

$$\mu_\Delta(M) < 1 \Longleftarrow |L| > c_L^{sl} = \frac{1 + |w_P|}{1 - |w_O|}. \tag{4.75}$$

The expressions for c_L^{su} and c_L^{sl} above are most easily derived from

$$\mu_\Delta(M) = \mu_{\left[\begin{smallmatrix}\Delta_O & \\ & \Delta_P\end{smallmatrix}\right]} \begin{bmatrix} w_O H & w_O H \\ w_P S & w_P S \end{bmatrix} = |w_O H| + |w_P S| = \left| w_O \frac{L}{1+L} \right| + \left| w_P \frac{1}{1+L} \right| \tag{4.76}$$

combined with the triangle inequality [e.g., use $|1 - |L|| \le |1 + L| \le 1 + |L|$ (see [32]

for details)].

Notice that the sufficient upper bound on $|L|$ is defined only for $|w_P| < 1$ (typically true for high frequencies), and the sufficient lower bound on $|L|$ is defined only for $|w_O| < 1$ (typically true for low frequencies). These conditions correspond to the requirement that $\mu_\Delta(N_{11}) < 1$ holds for the sufficient upper bound to be defined in Theorem 4.1, and to the requirement that $e(\infty) < k$ holds for the sufficient lower bound to be defined in Theorem 4.2.

Necessary Bounds For the necessary bounds to exist in Theorems 4.3 and 4.4, we need

$$\min_{\Delta_L} \mu_{\left[\begin{smallmatrix}\Delta_O & \\ & \Delta_P\end{smallmatrix}\right]} \left(F_l \left(\begin{bmatrix} 0 & 0 & w_O \\ w_P & w_P & -w_P \\ 1 & 1 & -1 \end{bmatrix}, \Delta_L \right) \right) < 1. \tag{4.77}$$

The above minimization can be solved analytically to give

$$\min_{\Delta_L} \mu_{\left[\begin{smallmatrix}\Delta_O & \\ & \Delta_P\end{smallmatrix}\right]} \left(F_l \left(\begin{bmatrix} 0 & 0 & w_O \\ w_P & w_P & -w_P \\ 1 & 1 & -1 \end{bmatrix}, \Delta_L \right) \right) = \min \left\{ |w_O|, |w_P| \right\}. \tag{4.78}$$

This equation is most easily derived using (4.76). The left-hand minimization in (4.78) is achieved for $\Delta_L^{-1} \to 0$ when $|w_O| < |w_P|$ and for $\Delta_L = 0$ when $|w_O| \geq |w_P|$.

A necessary condition for robust performance is that for each frequency either $|w_O|$ or $|w_P|$ is less than one. This will be assumed in the following derivation of the necessary bounds, since if this condition is not met, then robust performance cannot be met by any controller.

Theorem 4.3 gives the tightest necessary upper bound on L for robust performance to be achieved. The upper bound is c_L^{nu}, where c_L^{nu} at each frequency solves

$$\min_{\|\Delta_L\| \geq c_L^{nu}} \mu_{\left[\begin{smallmatrix}\Delta_O & \\ & \Delta_P\end{smallmatrix}\right]} \left(F_l \left(\begin{bmatrix} 0 & 0 & w_O \\ w_P & w_P & -w_P \\ 1 & 1 & -1 \end{bmatrix}, \Delta_L \right) \right) = 1. \tag{4.79}$$

Theorem 4.3 gives us that for $\mu_\Delta(M)$ to be less than 1 it is required that $|L|$ is lower than c_L^{nu}. The minimization in (4.79) can be solved analytically to give the following expression for c_L^{nu}:

$$\mu_\Delta(M) < 1 \Longrightarrow |L| < c_L^{nu} = \frac{1 - |w_P|}{|w_O| - 1}. \tag{4.80}$$

Again the expression for c_L^{nu} is most easily derived from (4.76) combined with the triangle inequality.

Theorem 4.4 gives the tightest necessary lower bound on L for robust performance to be achieved. The upper bound is c_L^{nl}, where c_L^{nl} at each frequency solves

$$\min_{\|\Delta_L\| \le c_L^{nl}} \mu_{\left[\begin{smallmatrix}\Delta_O & \\ & \Delta_P\end{smallmatrix}\right]} \left(F_l \left(\begin{bmatrix} 0 & 0 & w_O \\ w_P & w_P & -w_P \\ 1 & 1 & -1 \end{bmatrix}, \Delta_L \right) \right) = 1. \tag{4.81}$$

Theorem 4.4 gives us that for $\mu_\Delta(M)$ to be less than 1 it is required that $|L|$ is larger than c_L^{nl}. The minimization in (4.81) can be solved analytically to give the following expression for c_L^{nl}:

$$\mu_\Delta(M) < 1 \Longrightarrow |L| > c_L^{nl} = \frac{|w_P| - 1}{1 - |w_O|}. \tag{4.82}$$

Again the expression for c_L^{nl} is most easily derived from (4.76) combined with the triangle inequality.

Notice that the necessary upper bound on $|L|$ is defined only for $|w_O| > 1$ (typically true for high frequencies), and the necessary lower bound on $|L|$ is defined only for $|w_P| > 1$ (typically true for low frequencies). These conditions correspond to the requirement that $g(\infty) > 1$ holds for the necessary upper bound to be defined in Theorem 4.3, and to the requirement that $\mu_\Delta(N_{11}) > 1$ holds for the necessary lower bound to be defined in Theorem 4.4.

Comparison of Robust Loopshaping with Classical Loopshaping

The loopshaping bounds for the open loop transfer functions derived above agree with classical loopshaping. Similarly, the sufficient bounds on the closed loop transfer functions have been known for decades (see [32]); however, the necessary bounds were incomplete. The robust loopshaping theorems were used to calculate all the bounds for all frequencies. The distance between the necessary and the sufficient bounds quantifies the conservatism of the bounds near crossover. This complete picture also gives us *a priori* bandwidth ranges which must be satisfied by the controller, as will be shown in the next section. The real advantage of robust loopshaping over classical loopshaping is its ability to handle mixed real and complex uncertainty descriptions, multiple performance specifications, and decentralized controller design.

4.7 Example: DC Motor

Description Assume the nominal transfer function is the double integrator

$$P(s) = \frac{1}{s^2}. \tag{4.83}$$

This could describe a DC motor with negligible viscous damping. The nominal model, uncertainty description, and performance specifications for this example come from [32].

We are interested in good tracking over a bandwidth of about 1. If $|S| < 1/|w_P|$, where

$$w_P = \frac{10}{s^3 + 2s^2 + 2s + 1}, \tag{4.84}$$

then the tracking error is at most 10% over the desired closed loop bandwidth. The true plant is assumed to have a time delay, which was covered by a multiplicative

uncertainty of magnitude $|w_O|$ in [32], where

$$w_O = \frac{0.21s}{0.1s + 1}. \tag{4.85}$$

Analytical expressions for the robust loopshaping bounds are given in the previous section.

Closed Loop Design The upper plot in Fig. 4.5 gives the loopshaping bounds on H and the lower plot give the bounds on S. The complementary sensitivity H and sensitivity S are shown for an example design.

Our design approach is to find an S that satisfies nominal stability and has the sufficient bound on S satisfied for one part of the frequency range and the sufficient bound on H satisfied for the other part of the frequency range so that robust performance is guaranteed for all frequencies.

To have internal stability, the two plant poles at $s = 0$ cannot be canceled by the controller. So for nominal internal stability, $S = (1 + PK)^{-1}$ must have two zeros at $s = 0$ (interpolation condition (4.55)).

From Fig. 4.5 we see that the following form for S guarantees nominal stability, satisfies the necessary bounds on S for robust performance, and satisfies the sufficient bound on S for low frequencies:

$$S = \frac{s^2}{\lambda^2 s^2 + 2\lambda s + 1}. \tag{4.86}$$

Then the complementary sensitivity is

$$H = 1 - S = \frac{(\lambda^2 - 1)s^2 + 2\lambda s + 1}{\lambda^2 s^2 + 2\lambda s + 1}. \tag{4.87}$$

We have chosen $\lambda = 1/4$ for H and S in Fig. 4.5.

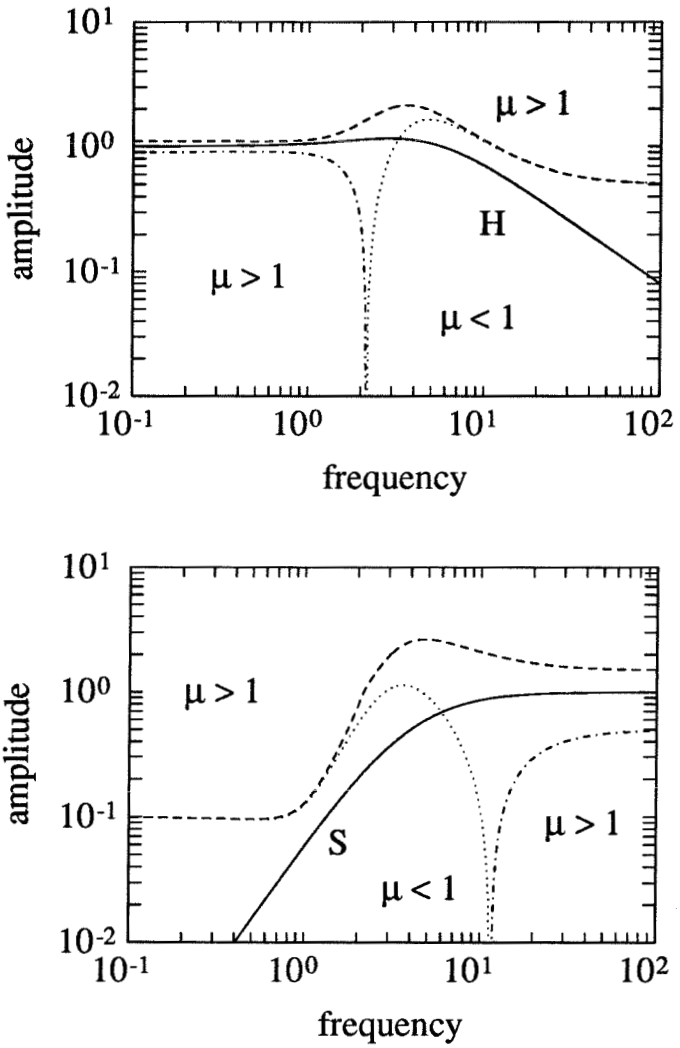


Figure 4.5: Loopshaping bounds on H and S for DC motor. The upper plot is for H and the lower plot is for S. The dashed lines are necessary upper bounds, the dashed and dotted lines are necessary lower bounds, and the dotted lines are sufficient upper bounds.

First focus on the bounds on H. We see the necessary lower bound exists only at low frequencies. This tends to always be the case—to meet high performance at lower frequencies, H must be very near one at low frequencies. The necessary lower bound requires that H have a bandwidth > 1.5 to meet robust performance at low frequencies.

The sufficient upper bound exists only at high frequencies; this tends to always be the case. The necessary upper bound at high frequencies requires that H roll off with a bandwidth < 20. The complementary sensitivity H meets the necessary and sufficient upper bounds at high frequencies. Note also that the necessary upper bound and sufficient upper bound coincide at high frequencies—this coincident bound is then a *necessary and sufficient* upper bound for robust performance. It is true in general that the bounds coincide when $|w_O| >> 1 > |w_P|$, since in this case the necessary bound and the sufficient bound (4.67,4.60) both approach

$$\frac{1 - |w_P|}{|w_O|}. \tag{4.88}$$

The inequality $|w_O| >> 1 > |w_P|$ usually holds at high frequencies because uncertainty is largest at high frequencies and performance requirements are smaller at high frequencies.

Now focus on the bounds on S. The necessary lower bound exists only at high frequencies; the sufficient upper bound exists only at low frequencies. The necessary lower bound requires that S have a bandwidth < 30 to meet robust performance at high frequencies. The necessary upper bound for $\omega < 2$ requires that S roll off at low frequencies with a bandwidth > 2. The sensitivity S meets the necessary upper bound and sufficient upper bound at low frequencies. Since P is strictly proper and the controller K must be proper, S must approach 1 at high frequencies. The necessary upper bound and sufficient upper bound coincide at low frequencies. This

is true in general when $|w_P| >> 1 > |w_O|$, since in this case the necessary upper bound and sufficient upper bound (4.68,4.61) both approach

$$\frac{1-|w_O|}{|w_P|}. \tag{4.89}$$

The inequality $|w_P| >> 1 > |w_O|$ usually holds at low frequencies because performance requirements are large at low frequencies (for example, for integral control $|w_P|$ approaches infinity as s approaches zero) and uncertainty is smallest at low frequency.

The most interesting region to consider is crossover. The sufficient upper bound on H ensures that robust performance is satisfied for ω greater than about 3. The sufficient upper bound on S ensures that robust performance is satisfied for ω less than about 6. Thus robust performance is satisfied for all frequency. The controller corresponding to S and H is

$$K = -\frac{15}{16}s^2 + \frac{1}{2}s + 1. \tag{4.90}$$

The controller K is augmented with a second-order filter to make K proper; then

$$K = (SP)^{-1}(1-S) = \frac{-\frac{15}{16}s^2 + \frac{1}{2}s + 1}{(\frac{1}{100}s+1)^2}. \tag{4.91}$$

Fig. 4.6 is a plot of the structured singular value for the proper controller. The value for μ is less than one for all frequencies, as implied by the satisfaction of the sufficient bound on S and/or H for each frequency.

The necessary bounds on S at high frequency are very lenient (far apart). This implies that the closed loop system can maintain robust performance for much more uncertainty than was used to cover time delay in the plant. At low frequencies, the bounds on H are also lenient, since H must approach 1 as ω approaches 0 (because of the integrators in the plant and the requirement of internal stability). The structured

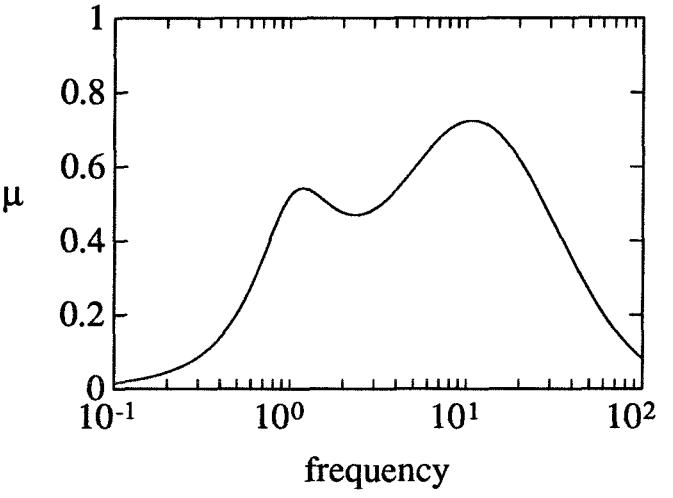


Figure 4.6: Robust performance test for DC motor: $\lambda = 0.25$.

singular value is much less than 1 at high and low frequencies, which confirms our judgment that the requirements on S and H are lenient at high and low frequencies. An increase in the steady-state performance requirement or the high frequency uncertainty would lead to a "flatter" structured singular value plot.

Our design goal was to meet the performance specification $|S| < 1/|w_P|$ for all plants described by the multiplicative uncertainty w_O. Since μ is much less than 1, better performance than specified can be achieved for a larger set of plants, i.e. the design is conservative. We will now remove the conservatism by designing our controller to give the fastest closed loop response while still meeting the sufficient bounds on S and H.

The value for λ was decreased until the sufficient bounds on S and H were barely satisfied—this was for $\lambda = 0.136$. The loopshaping plots are shown in Fig. 4.7. For $\lambda > 0.136$, the sufficient bound on H is no longer satisfied at $\omega = 15$. Since the necessary and sufficient bounds are essentially equal for $\omega = 15$, we expect the structured singular value for this design to be 1. Fig. 4.8 shows the structured singular value for this design. We see that the structured singular value is 1 at $\omega = 15$, so in this case loopshaping is not conservative.

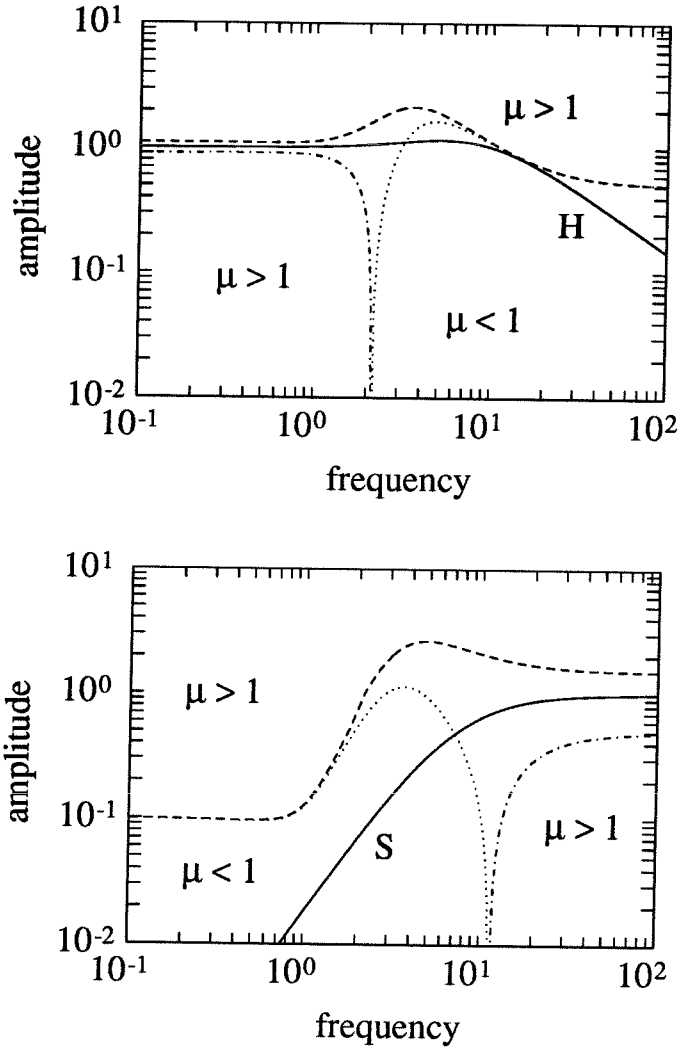


Figure 4.7: Loopshaping bounds on H and S for DC motor. Plots of S and H are for $\lambda = 0.136$. The upper plot is for H and the lower plot is for S. The dashed lines are necessary upper bounds, the dashed and dotted lines are necessary lower bounds, and the dotted lines are sufficient upper bounds.

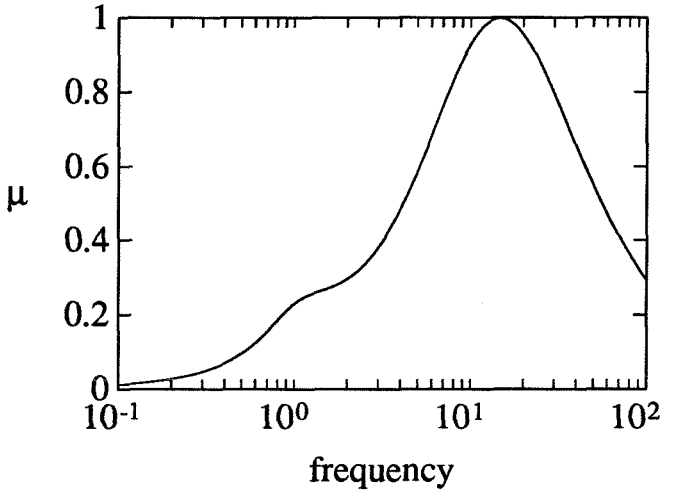


Figure 4.8: Robust performance test for DC motor: $\lambda = 0.136$.

We could have designed the controller without knowledge of where the necessary bounds were. The necessary bounds give the additional information of when the design is conservative and what are the minimum and maximum closed loop bandwidths required for robust performance to be achieved.

Open Loop Design The structure of the controller designed by specifying closed loop transfer functions is somewhat awkward, with a right half plane zero at $s = 4/3$. When loopshaping an open loop transfer function, we can directly specify the structure of the controller. We could try to design a PID controller, but it is clear that the integral term is not needed and would add additional phase lag which would be difficult to counteract using the derivative term. Thus we will design a *PD* controller by loopshaping L.

The formula for a PD controller, where the derivative action is assumed to be effective over one decade, is

$$K = k\frac{\tau_D s + 1}{0.1\tau_D s + 1}, \tag{4.92}$$

where k is the gain and τ_D is the derivative time.

The loopshaping bounds on L are given in Fig. 4.9. The open loop transfer

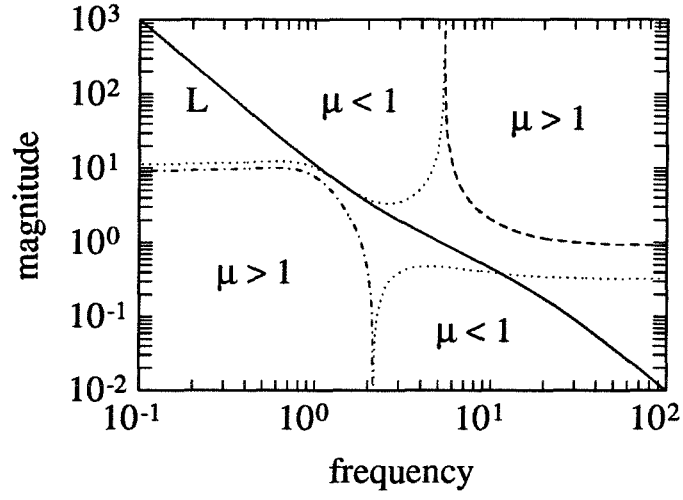


Figure 4.9: Loopshaping bounds on L for DC motor. The solid line is $|L|$, the dotted line at low frequencies is the sufficient lower bound, the dotted line at high frequencies is the sufficient upper bound, the dashed line is the necessary upper bound, the dashed-dotted line is the necessary lower bound.

function for an example design ($k = 10$, $\tau_D = 0.5$) is also shown. Dropping the second-order term in s in the numerator of K in (4.91) gives us a numerator time constant of 0.5—we take this to be the derivative time τ_D. The gain was then chosen to ensure that the loopshape L would satisfy the low and high frequency necessary conditions (this gave $k = 10$). The Bode magnitude and phase plots for the resulting loopshape are given in Fig. 4.10.

We see from Fig. 4.10 that the derivative term adds phase lead to the loopshape, which is needed to stabilize the system since the plant has phase lag of 180^o. We see from Fig. 4.9 that the sufficient bounds are satisfied for high and low frequencies. It is impossible to satisfy the sufficient bounds on L at crossover—this will be true in general when loopshaping with an open loop transfer function. Also, though nominal stability cannot be guaranteed *a priori* while loopshaping L, it was still easy to stabilize the system by adding the necessary phase lead at crossover. The closed loop poles with the above PD controller are $\{-13.8, -3.10 \pm 2.21i\}$.

Fig. 4.11 is a plot of the structured singular value for the PD controller. The

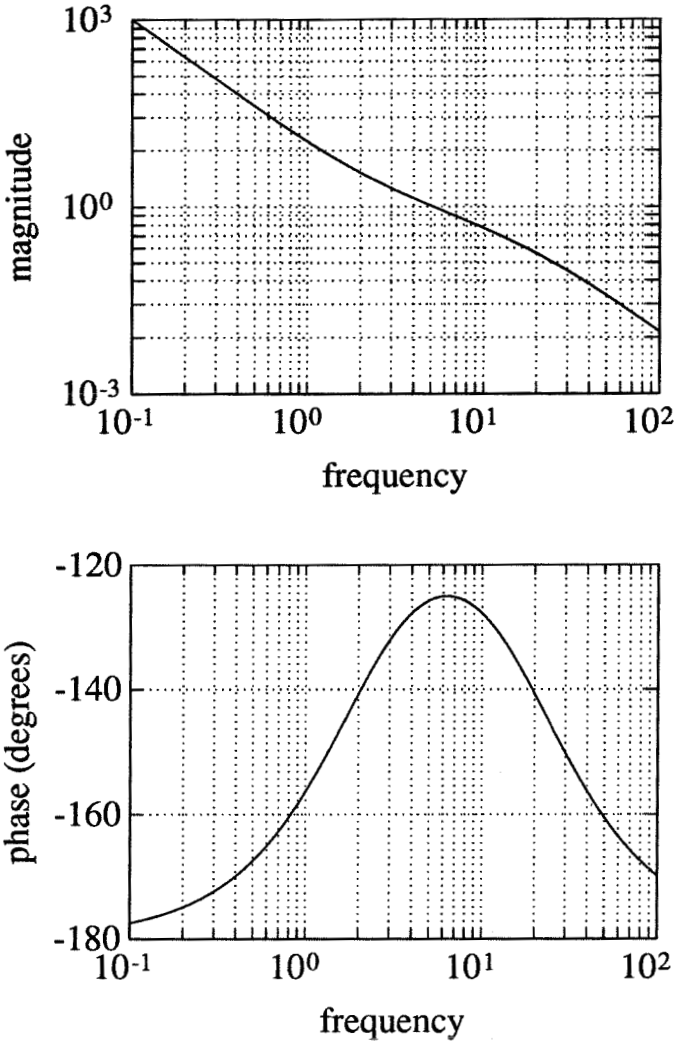


Figure 4.10: Bode magnitude and phase plots for the open loop transfer function L.

maximum μ is equal to 1.14, which is larger than for the previous design (4.91).

The optimal PD controller was found through optimization to have parameters

$$k = 8.395, \qquad \tau_D = 0.5594, \tag{4.93}$$

which gives a μ value of 1.10. We see that the PD controller designed via loopshaping L is very close to optimal.

A natural question to ask is why the μ value is so much larger for the PD controller than for the controller designed via closed loop loopshaping in (4.91). One reason is

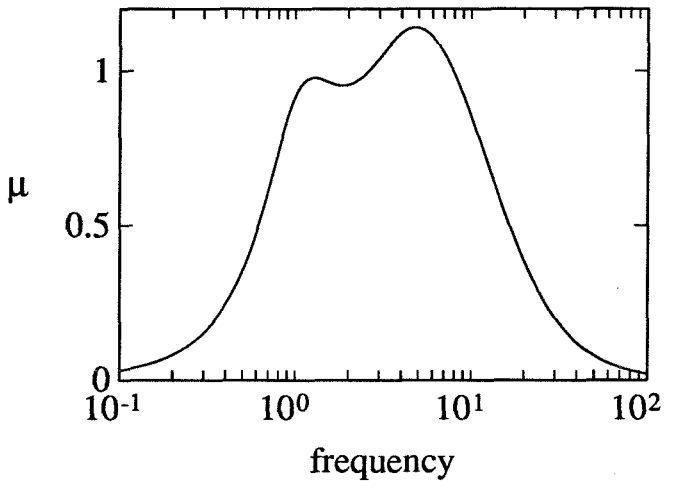


Figure 4.11: Robust performance test with the PD controller designed for the DC motor.

that the controller in (4.91) has a much faster time constant (0.01 vs. 0.05). Another reason is that the PD controller stabilizes the system by introducing only phase *lead* at crossover (no RHP zeros), and since the derivative time is active over only one decade, the PD controller can introduce only a limited amount of phase lead at crossover. It is interesting to note that the optimal PD controller which has derivative time active over two decades (and so has a faster time constant and can introduce more phase lead) has a peak μ value of 0.954, which satisfies robust performance.

It is interesting to consider the high and low frequency limits for the necessary and the sufficient bounds.

The necessary and the sufficient upper bounds exist only at high frequencies; this tends to always be the case. The necessary upper bound at high frequencies requires that L roll off sufficiently fast. Since $L = PK$ is strictly proper, L must approach 0 at high frequencies. Note that when $|w_O| >> 1 > |w_P|$ the upper bounds coincide—this coincident bound is then a *necessary and sufficient* upper bound for

robust performance. In this case, the upper bounds (4.74,4.80) both approach

$$\frac{1-|w_P|}{|w_O|}. \tag{4.94}$$

This would be expected to hold only at high frequencies because here the uncertainty is largest and the performance requirements are small. The upper bounds do not coincide for high frequencies in Fig. 4.9 because the uncertainty weight is very lenient for this example $[w_O(\infty) = 2.1]$.

The necessary and the sufficient lower bounds exist only at low frequencies. The necessary lower bound requires that L have sufficiently high gain at low frequencies. Since L has a double integrator, L must approach ∞ at low frequencies. The lower bounds coincide when $|w_P| >> 1 > |w_O|$. In this case, the lower bounds (4.75,4.82) both approach

$$\frac{|w_P|}{1-|w_O|}. \tag{4.95}$$

The inequality $|w_P| >> 1 > |w_O|$ usually holds at low frequencies because here the performance requirements are large (for example, for integral control $|w_P|$ approaches infinity as s approaches zero) and the uncertainty is small. As expected, the lower bounds nearly coincide for low frequencies in Fig. 4.9. The bounds would coincide at low frequencies if integral action had been an explicit performance requirement. Since the plant is a double integrator, integral action is satisfied automatically.

From Fig. 4.9 we see that we have $|L| >> c_L^{sl}$ at low frequencies and $|L| << c_L^{su}$ at high frequencies. This suggests that the closed loop system can meet a more stringent performance specification at low frequencies and be robust to more uncertainty at high frequencies. We see from Fig. 4.11 that the structured singular value is much less than 1 at low and high frequencies, which confirms our judgment that the performance and stability requirements are lenient at low and high frequencies. An increase in the steady-state performance and the high frequency uncertainty requirements would

lead to a "flatter" structured singular value plot.

4.8 Gain and Phase Margins for SISO Plants

The gain and phase margins are given directly by the open loop transfer function.

Definition 4.1 (Gain Margin) *The gain margin (GM) is defined by*

$$GM = -\frac{1}{L(j\omega_{gm})}, \tag{4.96}$$

where ω_{gm} is the frequency of the leftmost intersection of $L(j\omega)$ with the negative real axis in the SISO Nyquist plot.

Definition 4.2 (Phase Margin) *The phase margin (PM) is defined by*

$$PM = \pi + \angle L(j\omega_{pm}), \tag{4.97}$$

where ω_{pm} is the frequency of the leftmost intersection of $L(j\omega)$ with the unit circle in the SISO Nyquist plot.

The frequencies ω_{gm} and ω_{pm} are commonly referred to as the gain and phase crossover frequencies [55].

Gain and phase margin goals can be quantified in terms of the sensitivity and complementary sensitivity. The proofs of the following lemmas are left to the reader.

Lemma 4.4 *The sensitivity and complementary sensitivity are related to the gain margin by*

$$S(j\omega_{gm}) = \frac{GM}{GM-1}, \qquad H(j\omega_{gm}) = \frac{-1}{GM-1} \tag{4.98}$$

where ω_{gm} is the frequency of the leftmost intersection of $L(j\omega)$ with the negative real axis in the SISO Nyquist plot.

Lemma 4.5 *The sensitivity and complementary sensitivity are related to the phase margin by*

$$|S(j\omega_{pm})| = |H(j\omega_{pm})| = \frac{1}{\sqrt{2(1-\cos(PM))}}, \tag{4.99}$$

where ω_{pm} is the frequency of the leftmost intersection of $L(j\omega)$ with the unit circle in the SISO Nyquist plot.

Gain and phase margins can be specified by loopshaping the open loop transfer function L using (4.96-4.97) or by loopshaping the closed loop transfer functions using (4.98-4.99). For example, to meet specified gain and phase margins, the sensitivity is shaped to be less than the values given by the right-hand sides in (4.98) and (4.99). The loopshaping bounds (and perhaps a design iteration) suggest where the gain and phase crossovers will be. Below we give an example illustrating this technique.

Example: DC Motor with Time Delay Assume the nominal transfer function is

$$P(s) = \frac{1}{s^2} \cdot \frac{20-s}{20+s}. \tag{4.100}$$

This could describe a DC motor with negligible viscous damping and a time delay of 0.1 seconds. The time delay is modeled with a first order Padé approximation. The uncertainty description covers the error introduced by this approximation.

The uncertainty and performance specifications are the same as in Section 4.7, except that margin specifications must also be met—the gain margin must be greater than 3, and the phase margin must be greater than 45°. From (4.98) and (4.99) we see that specifying these margins is equivalent to specifying that $|S(j\omega_{gm})| < 1.5$ and $|S(j\omega_{pm})| < 1.31$.

In Section 4.6 we listed the bounds for S and H. The upper plot in Fig. 4.12 gives the loopshaping bounds on H and the lower plot gives the bounds on S. The complementary sensitivity H and sensitivity S are shown for an example design.

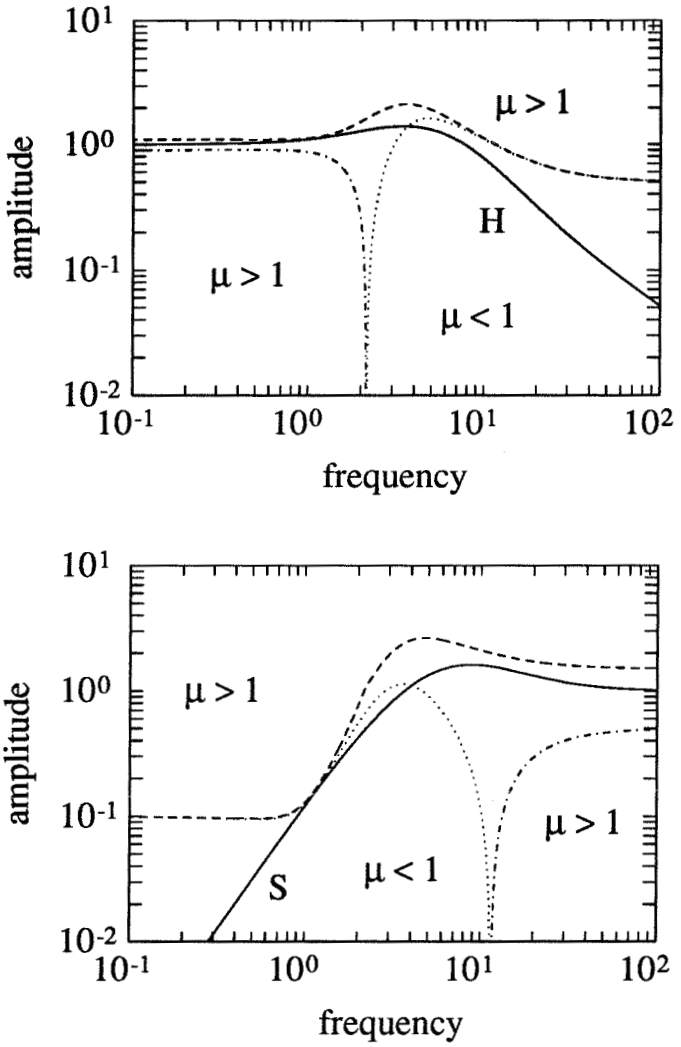


Figure 4.12: Loopshaping bounds on H and S for the first control design for the DC motor with time delay. The upper plot is for H and the lower plot is for S. The dashed lines are necessary upper bounds, the dashed and dotted lines are necessary lower bounds, and the dotted lines are sufficient upper bounds.

To have internal stability, the two plant poles at $s = 0$ and the plant zero at $s = 20$ cannot be canceled by the controller. So for nominal internal stability, $S = (1+PK)^{-1}$ must have two zeros at $s = 0$ (interpolation condition (4.55)), and must satisfy $S(20) = 1$ (interpolation condition (4.54)). Since the plant is strictly proper and the controller must be proper, S must also satisfy $S(\infty) = 1$.

Let us try the following form for S which gives a nominally stable system:

$$S = \frac{\lambda^2 s^2}{(\lambda s + 1)^2} \cdot \frac{s + a}{s + b}, \tag{4.101}$$

where

$$a = \frac{(20\lambda + 1)^2}{(20\lambda)^2}(20 + b) - 20, \tag{4.102}$$

and b is arbitrary. For simplicity, we initially take $b = 1/\lambda$ so that the denominator time constants of S are equal. The complementary sensitivity H is given by $H = 1-S$. The controller calculated from $K = (SP)^{-1}(1 - S)$ is improper, and so is augmented with the second-order filter

$$\frac{1}{(0.01s + 1)^2} \tag{4.103}$$

before calculating gain and phase margins and the structured singular value. The closed loop poles are calculated to ensure that nominal stability is still satisfied by the augmented controller.

The loopshaping bounds in Fig. 4.12 suggest that we try $\lambda \approx 0.2$. Plotting S and H for different values of λ near 0.2 shows that the necessary bounds on S and H are satisfied for $0.16 < \lambda < 0.18$. The design shown in Fig. 4.12 is for $\lambda = 0.18$. Since the sufficient bound on S is satisfied for $\omega < 3.8$ and the sufficient bound on H is satisfied for $\omega > 4.0$, we expect robust performance to be approximately satisfied. The structured singular value is plotted in Fig. 4.13, and the peak μ value is 0.998.

We calculated the margins for several controllers with λ in the range from 0.16 to 0.18. We found that $\lambda = 0.18$ gives the best margins of all the controllers that satisfy

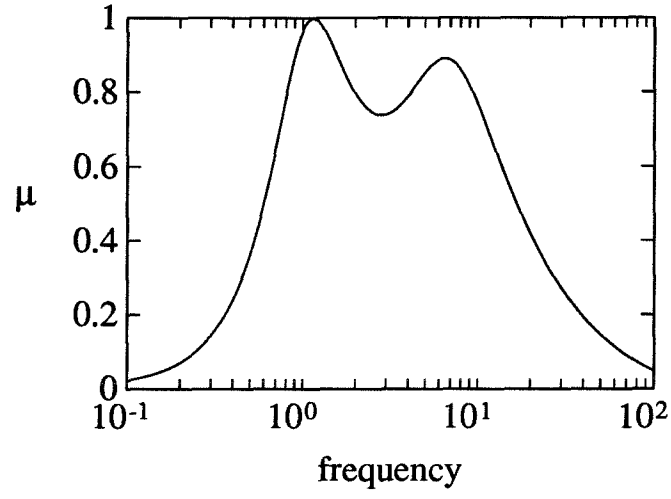


Figure 4.13: Robust performance test with the first control design for the DC motor with time delay.

robust performance and are given by (4.101), (4.102), and $b = 1/\lambda$. The margins, with their respective crossover frequencies, are

$$GM = 2.93 < 3, \qquad \omega_{gm} = 13.9,$$
$$PM = 38.7^o < 45^o, \qquad \omega_{pm} = 4.93.$$

The margin specifications are not satisfied.

Now we modify the design to improve the margins. The peak in the sensitivity occurs at $\omega = 7.83$, which is between ω_{pm} and ω_{gm}. This suggests that we should be able to improve the margins by reducing the peak in the sensitivity. This can be done by choosing $b = 10/\lambda$. Again we augment the controller with the second-order filter (4.103) before calculating margins and the structured singular value.

The loopshaping bounds are given for $\lambda = 0.28$ in Fig. 4.14. The necessary bounds on S and H are satisfied for all frequencies. Since the sufficient bound on S is satisfied for $\omega < 4.4$ and the sufficient bound on H is satisfied for $\omega > 3.5$, μ must be less than 1 for all frequencies. The structured singular value is plotted in Fig. 4.15. The peak μ value of 0.96 is less than 1, as implied by the satisfaction of the sufficient bound on S and/or H for each frequency. The closed loop poles are $\{-116.1 \pm 44.5i, -20.0, -15.5, -6.92, -2.75\}$, so the system is nominally stable.

The margins for this design are

$$GM = 3.02 > 3, \qquad \omega_{gm} = 19.6,$$
$$PM = 47^o > 45^o, \qquad \omega_{pm} = 5.42.$$

Both the crossover frequencies and the peak in the sensitivity is shifted to higher frequencies. The peak in the sensitivity is reduced from 1.85 to 1.65, and this results in the improved margins.

4.9 Multiple Performance Specifications

The control engineer is often interested in meeting multiple performance specifications. For example, one might want a controller that remains stable under slow sensor drift or variations in actuator or sensor gain (this is referred to as *fault tolerance*). If there are sensors or actuators that are prone to failures, then we would like to specify that the closed loop system remain stable or satisfy some minimum performance whenever these sensors/actuators fail (this is referred to as *failure tolerance*).

Often the designer prefers to give specifications not in terms of robust performance, but in terms of nominal performance plus robust stability. For example, the specifications for the 1990-92 ACC Benchmark Problem [9, 10] are that the overshoot and settling time should be minimized for the nominal plant, and that stability is satisfied for some set of plants. Separate specifications are used whenever the designer expects that an overall robust performance specification will lead to an overly conservative design. Multiple performance specifications are easy to handle using loopshaping—the bounds are calculated individually for each specification, and the most restrictive bounds are used for loopshaping. This is illustrated further in the next chapter.

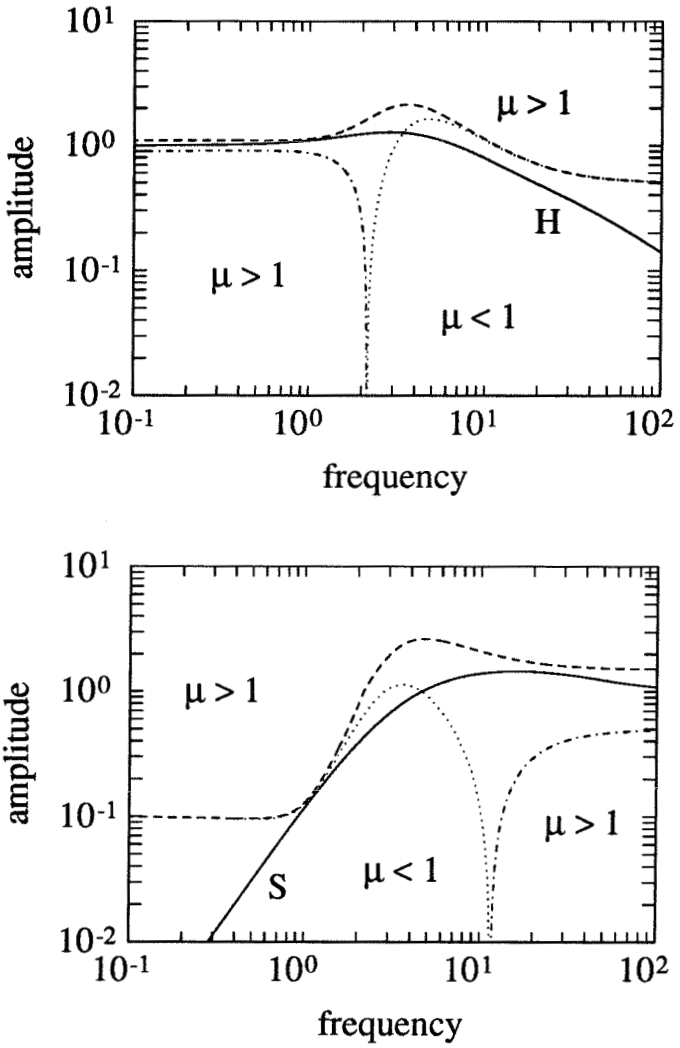


Figure 4.14: Loopshaping bounds on H and S for the second control design for the DC motor with time delay. The upper plot is for H and the lower plot is for S. The dashed lines are necessary upper bounds, the dashed-dotted lines are necessary lower bounds, and the dotted lines are sufficient upper bounds.

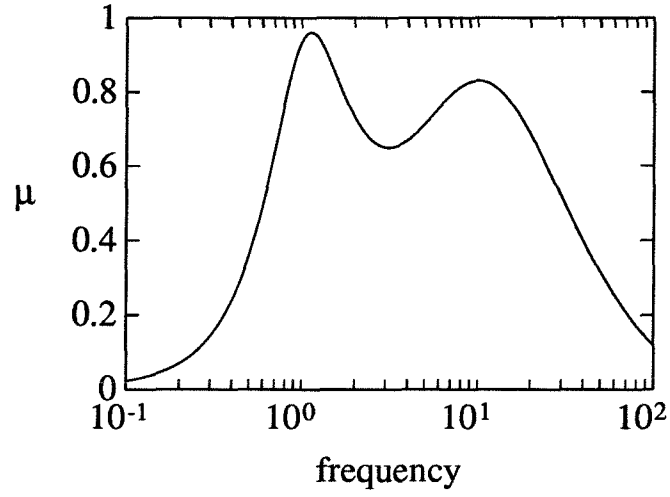


Figure 4.15: Robust performance test with the second control design for the DC motor with time delay.

4.10 Conclusions

Robust loopshaping bounds have been derived for general mixed real/complex uncertainties. Low order robust controllers can be designed by loopshaping open loop transfer functions. Either open loop or closed loop transfer functions can be loopshaped to meet gain and phase margin specifications. Robust controllers that meet multiple performance specifications are designed by using the most restrictive bounds for loopshaping. Robust loopshaping was shown to agree with and extend the original classical loopshaping bounds derived by Bode [7] when applied to simple SISO systems. The next chapter shows how to design robust decentralized controllers to meet failure and fault tolerance specifications.

4.11 Appendix

Here we derive the expressions given in Section 4.2.

To use the robust loopshaping theorems we need to find an LFT in terms on T which describes M (see Fig. 4.1). In many cases, this is done by inspection. When

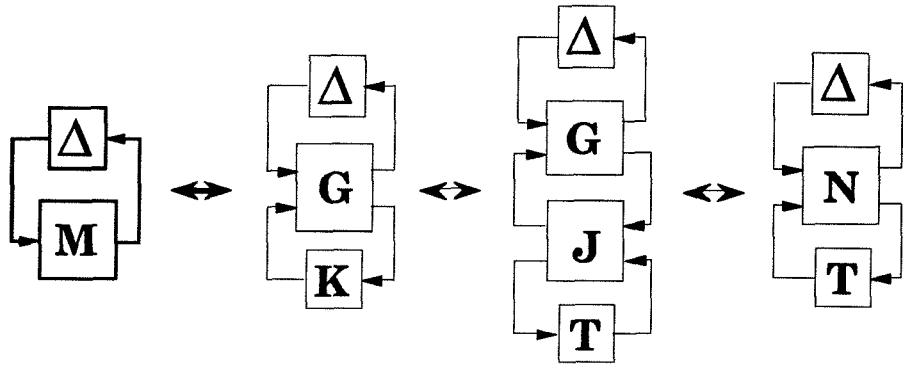


Figure 4.16: Equivalent representations of system M with perturbation Δ.

the controller K can be written as an LFT of T, the following procedure is often applicable:

1. The interconnection structure in terms of G and K is found directly by rearranging the system's block diagram.

2. Write the controller K as an LFT of T, i.e.

$$K = F_l(J,T) = J_{11} + J_{12}T(I - J_{22}T)^{-1}J_{21}. \tag{4.104}$$

3. Assume $I - G_{22}J_{11}$ is invertible. Then, given G and J, N follows (see Fig. 4.16) because any interconnection of LFT's is again an LFT:

$$N = \begin{bmatrix} N_{11} & N_{12} \\ N_{21} & N_{22} \end{bmatrix} = \begin{bmatrix} G_{11} + G_{12}J_{11}(I - G_{22}J_{11})^{-1}G_{21} & G_{12}(I - J_{11}G_{22})^{-1}J_{12} \\ J_{21}(I - G_{22}J_{11})^{-1}G_{21} & J_{22} + J_{21}G_{22}(I - J_{11}G_{22})^{-1}J_{12} \end{bmatrix}. \tag{4.105}$$

Many examples of using this procedure, and some more detailed comments, are given in [76]. We will use the above procedure to find N^H in terms of G and will get N^S by inspection from N^H. The transfer function G in step one of the procedure is calculated by the subroutine *sysic* [3] from the block diagram.

Using $K = P^{-1}H(I-H)^{-1}$, we have $K = F_l(J^H, H)$ where

$$J^H = \begin{bmatrix} 0 & P^{-1} \\ I & I \end{bmatrix}. \tag{4.106}$$

Using (4.105), we get N^H in terms of G

$$N^H = \begin{bmatrix} G_{11} & G_{12}P^{-1} \\ G_{21} & 0 \end{bmatrix}. \tag{4.107}$$

Substituting H by $I - S$ into

$$M = N_{11}^H + N_{12}^H H N_{21}^H \tag{4.108}$$

gives N^S

$$N^S = \begin{bmatrix} G_{11} + G_{12}P^{-1}G_{21} & -G_{12}P^{-1} \\ G_{21} & 0 \end{bmatrix}. \tag{4.109}$$

The expressions for N^L and N^K follow by inspection of Fig. 4.1.

Chapter 5

Fault/Failure Tolerant Decentralized Controller Design

Summary

Equipment never behaves perfectly all the time—actuator gains may vary, sensor outputs may slowly drift over time, valves may get stuck, composition analyzers are typically prone to failure. *Fault tolerance* refers to the ability of the control system to meet some performance specifications even when pieces of equipment become faulty. *Failure tolerance* refers to the ability of the control system to meet some (weaker than normal) performance specifications under equipment failure. Conventional feedback control designs for a multivariable plant may result in poor performance, or even instability, in the event of equipment faults or failures, even though it may be possible to control the plant using only the available inputs and outputs.

Even though the importance of designing fault/failure tolerant controllers is clear, no current design method exists with guarantees on system performance while taking into account the mismatch between the model and the plant. This chapter develops such a design technique for decentralized controllers, based on the *robust loopshaping*

paradigm presented in the previous chapter.

5.1 Introduction

Equipment never behaves perfectly all the time—actuator gains may vary, sensor outputs may slowly drift over time, valves may get stuck, composition analyzers are typically prone to failure. *Fault tolerance* refers to the ability of the control system to meet some performance specifications even when pieces of equipment become faulty. *Failure tolerance* refers to the ability of the control system to meet some (weaker than normal) performance specifications under equipment failure. Conventional feedback control designs for a multivariable plant may result in poor performance, or even instability, in the event of equipment faults or failures, even though it may be possible to control the plant using only the available inputs and outputs.

For a more detailed example illustrating the importance of designing failure tolerant controllers, consider a distillation column where the setpoints are the top and bottom compositions. Composition measurements are often too slow for effective control, so usually the controller is designed to use temperature measurements. The drawback of using temperature measurements only is that it is then impossible to have zero steady-state error in the compositions. Thus it is advantageous to design the control system that uses temperature measurements, and also uses composition measurements when these are available. Composition analyzers are typically prone to failure. When a composition measurement fails, the control system should be capable of giving acceptable performance using only temperature measurements. Such a controller is said to be failure tolerant.

Even though the importance of designing fault/failure tolerant controllers is clear, relatively few design methods have been proposed (see Veillette et al. [113] for a survey). None of these methods, except for the method of Veillette et al. [113], gives guarantees on system performance. Their method designs for nominal performance,

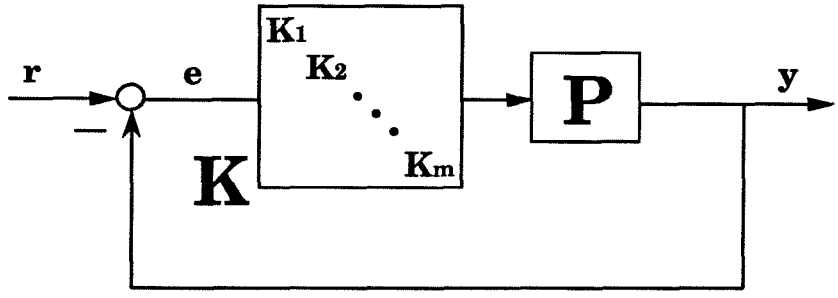


Figure 5.1: Decentralized control structure.

i.e., the satisfaction of a bound on the H_∞-norm of some transfer function of interest. The model is always an imperfect representation for the true process, thus it is needed to design the controller to be *robust*. This chapter presents the first method for designing robust fault/failure tolerant decentralized controllers, based on the *robust loopshaping* paradigm presented in the previous chapter.

Decentralized Control Decentralized control involves using a diagonal controller (see Fig. 5.1)

$$K = \text{diag}\{K_i\}. \tag{5.1}$$

This includes controllers that can be made diagonal by reordering the measured variables and manipulated variables.

Some reasons for using a decentralized controller are

- tuning and retuning is simple
- they are easy to understand
- they are easy to make failure tolerant
- implementation and maintenance is simpler

These reasons explain the predominance of decentralized controllers in applications.

The design of a decentralized control system involves two steps. First the *control structure* must be selected. This involves the choosing of the actuators and sensors and

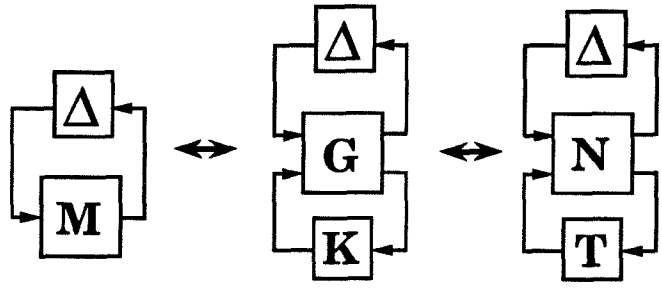


Figure 5.2: Equivalent representations of system M with perturbation Δ. The transfer function T is chosen to be a parametrization of the controller K.

the pairings between the chosen actuators and sensors, and is discussed in Chapter 6. The second step is the *design* of each single loop controller K_i. These controllers will be designed using the robust loopshaping method.

Diagonal Open Loop and Closed Loop Transfer Functions To design decentralized controllers via robust loopshaping, we will need the following definitions. Define $\tilde{P}$ to be the diagonal part

$$\tilde{P} = \text{diag}\{P_{ii}\}. \tag{5.2}$$

Define the *diagonal open loop transfer function* by $\tilde{L} = \tilde{P}K$, the *diagonal complementary sensitivity* by $\tilde{H} = \tilde{P}K(I + \tilde{P}K)^{-1}$ and the *diagonal sensitivity* by $\tilde{S}(I + \tilde{P}K)^{-1}$. Note that $\tilde{L}$, $\tilde{H}$, and $\tilde{S}$ parametrize the decentralized controller K. Decentralized controllers are designed by loopshaping these transfer functions. Note that $\tilde{H} + \tilde{S} = I$.

Parametrize Controller in Terms of T To design controllers via robust loopshaping, we need to find an LFT of T which describes M (see Fig. 5.2), where T parametrizes the controller we are to design. In the previous chapter we derived LFTs of H, S, L, and K which describe M. For decentralized control, T is usually chosen to be $\tilde{H}$, $\tilde{S}$, $\tilde{L}$, or K. We will derive these LFTs starting from the expression

for N^H in terms of G given in the previous chapter:

$$N^H = \begin{bmatrix} G_{11} & G_{12}P^{-1} \\ G_{21} & 0 \end{bmatrix}. \tag{5.3}$$

We have by definition of the LFT of M that

$$M = N_{11}^H + N_{12}^H H N_{21}^H. \tag{5.4}$$

The following expression can be verified:

$$H = P\tilde{P}^{-1}\tilde{H}(I + (P\tilde{P}^{-1} - I)\tilde{H})^{-1}. \tag{5.5}$$

Substituting the above equation into (5.4) and rearranging gives

$$N^{\tilde{H}} = \begin{bmatrix} G_{11} & G_{12}\tilde{P}^{-1} \\ G_{21} & I - P\tilde{P}^{-1} \end{bmatrix}. \tag{5.6}$$

Similarly, substituting $S = \tilde{S}(I - (I - \tilde{P}P^{-1})\tilde{S})^{-1}\tilde{P}P^{-1}$ into (5.4) gives

$$N^{\tilde{S}} = \begin{bmatrix} G_{11} + G_{12}P^{-1}G_{21} & -G_{12}P^{-1} \\ \tilde{P}P^{-1}G_{21} & I - \tilde{P}P^{-1} \end{bmatrix}. \tag{5.7}$$

The expressions for $N^{\tilde{L}}$ and N^K follow by inspection of Fig. 5.2.

$$N^{\tilde{L}} = \begin{bmatrix} G_{11} & G_{12}\tilde{P}^{-1} \\ G_{21} & G_{22}\tilde{P}^{-1} \end{bmatrix}, \tag{5.8}$$

$$N^K = G. \tag{5.9}$$

A simple program can be written that calculates $N^{\tilde{H}}$, $N^{\tilde{S}}$, $N^{\tilde{L}}$, and N^K given the transfer functions describing the system components and the location of the uncertainty blocks Δ_i.

Simultaneous Versus Sequential Design In industry decentralized controllers are usually designed sequentially, i.e., the SISO loops are tuned and closed one at a time. The advantage of sequential design is that at each step in the design procedure we have a SISO control problem. A drawback of this approach is that closing a loop during the design procedure may make previously closed loops perform poorly. A procedure for deciding the appropriate bandwidth for each SISO control loop, and the order in which to tune and close the SISO loops, is given in [51].

Another approach is to design the SISO controllers simultaneously—this is referred to by Skogestad and Morari as *independent designs* [105]. The SISO controllers are designed simultaneously by requiring that each SISO loop be designed with the same closed loop time constant. An advantage of this approach is that it is easy to design for quite advanced forms of failure and fault tolerance as will be shown later in this chapter. A disadvantage of this approach is that it may be required to reject disturbances much faster in some outputs than others, and so the loops corresponding to these outputs must be designed to have shorter closed loop time constants.

Robust loopshaping can be used to design controllers for either of these approaches to decentralized controller design. Simultaneous design of decentralized controllers via robust loopshaping will be illustrated through examples in this chapter. In sequential design, the SISO control loops can be designed via robust loopshaping. The speed of response for the currently-designed SISO controller is based on estimates of the speed of response for control loops yet to be tuned (see Hovd [51] for details).

Organization The remainder of this chapter is organized as follows. First we design a robust decentralized controller for a high purity distillation column. Second, we discuss how to handle multiple performance specifications in the robust loopshaping framework. We show how to analyze fault and failure tolerance, with reference to earlier research in the area. We discuss how to design fault/failure tolerant controllers,

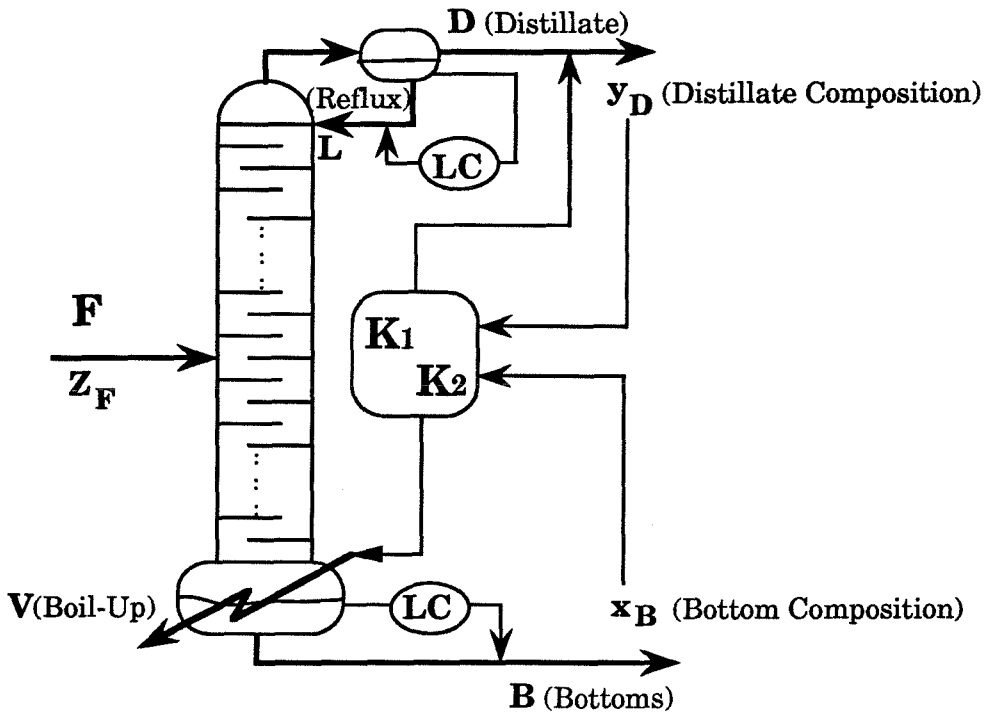


Figure 5.3: High purity distillation column in DV configuration.

and illustrate the technique on the high purity distillation column.

5.2 Example: High-Purity Distillation

Description We will now illustrate how to design robust decentralized controllers via robust loopshaping on a high-purity distillation column given in [105] and discussed in more detail in [106]. The nominal model is

$$P = \frac{1}{75s+1}\begin{bmatrix} -0.878 & 0.014 \\ -1.082 & -0.014 \end{bmatrix}. \tag{5.10}$$

This nominal model may correspond to a high-purity distillation column using distillate and boilup as manipulated inputs to control top and bottom composition (see Fig. 5.3) using measurements of the top and bottom compositions. The plant has a large condition number, so input uncertainty strongly affects robust performance

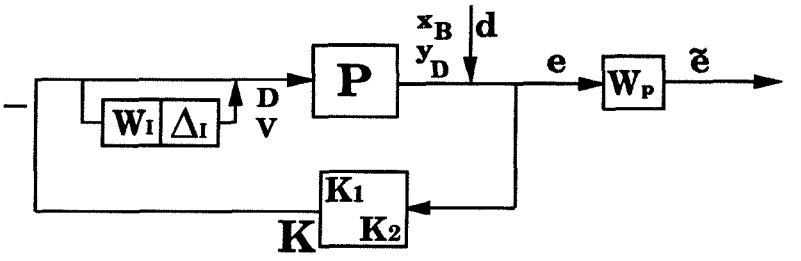


Figure 5.4: The plant with input uncertainty Δ_I of magnitude $w_I(s)$. Robust performance is satisfied if $\overline{\sigma}\left(w_P(I+\hat{P}K)^{-1}\right) \leq 1$ for all Δ_I with $\|\Delta_I\|_\infty \leq 1$.

[106]. The uncertainty and performance weights are

$$w_I(s) = 0.1\frac{5s+1}{0.25s+1}, \qquad w_P(s) = 0.25\frac{7s+1}{7s}. \tag{5.11}$$

The robust performance condition is a bound on the sensitivity, i.e., $\overline{\sigma}(S) < 1/|w_P|$, $\forall \hat{P} \in \Pi$. The input uncertainty includes actuator uncertainty and neglected right half plane zeros of the plant. The performance bound implies zero steady-state error and a closed loop time constant of 7 minutes. The uncertainty block Δ_I is a diagonal 2×2 matrix (independent actuators) and the performance block Δ_P is a full 2×2 matrix. Fig. 5.4 is a block diagram of the system.

Decentralized Control Design Robust performance is satisfied if and only if $\mu_\Delta(M) < 1$ for all frequencies where

$$M = \begin{bmatrix} -w_I P^{-1}HP & -w_I P^{-1}H \\ w_P SP & w_P S \end{bmatrix}, \qquad \Delta = \begin{bmatrix} \Delta_I & \\ & \Delta_P \end{bmatrix}. \tag{5.12}$$

The expressions for $N^{\tilde{H}}$ and $N^{\tilde{S}}$ can be found by inspection or G could be found and the equations in Section 5.1 could be used.

The transfer function $M = F_l(N^{\tilde{H}}, \tilde{H})$, where

$$N_{11}^{\tilde{H}} = \begin{bmatrix} 0 & 0 \\ w_P P & w_P I \end{bmatrix}, N_{12}^{\tilde{H}} = \begin{bmatrix} -w_I \tilde{P}^{-1} \\ -w_P P \tilde{P}^{-1} \end{bmatrix}, N_{21}^{\tilde{H}} = \begin{bmatrix} P & I \end{bmatrix}, N_{22}^{\tilde{H}} = I - P\tilde{P}^{-1}. \tag{5.13}$$

The transfer function $M = F_l(N^{\tilde{S}}, \tilde{S})$, where

$$N_{11}^{\tilde{S}} = \begin{bmatrix} -w_I I & -w_I P^{-1} \\ 0 & 0 \end{bmatrix}, N_{12}^{\tilde{S}} = \begin{bmatrix} w_I P^{-1} \\ w_P I \end{bmatrix}, N_{21}^{\tilde{S}} = \begin{bmatrix} \tilde{P} & \tilde{P}P^{-1} \end{bmatrix}, N_{22}^{\tilde{S}} = I - \tilde{P}P^{-1}. \tag{5.14}$$

Note that the equation (35b) given in [105] is incorrect.

As discussed in Chapter 4, the sufficient lower bound for closed loop transfer functions does not exist at any frequency for reasonable choices of performance and uncertainty descriptions. Theorems 4.1, 4.3, and 4.4 are applied with $N^{\tilde{H}}$ (and $N^{\tilde{S}}$) to give the sufficient upper bound and the necessary upper and lower bounds on $\tilde{H}$ (and $N^{\tilde{S}}$) for robust performance to be achieved.

The loopshaping bounds for $\tilde{H} = hI$ and $\tilde{S} = sI$ are plotted in Fig. 5.5 for

$$h = \frac{1}{4s+1}, \qquad s = \frac{4s}{4s+1}. \tag{5.15}$$

The necessary bounds are tight at low frequencies—forcing h to be near one for low frequencies. At high frequencies the necessary upper bound and the sufficient upper bounds are close, and rolling off h meets both of these bounds. Around crossover the bounds are very conservative. This tends be true in general, but is especially true for this plant because it has a large condition number ($\kappa(P) = \overline{\sigma}(P)/\underline{\sigma}(P) = 70.8$). It is well-known that loopshaping is conservative for systems with large condition number [76].

The given h meets the sufficient condition for robust performance for frequencies above about 0.45.

At zero frequency, the necessary bounds are equal to one; this implies that $h(0)$

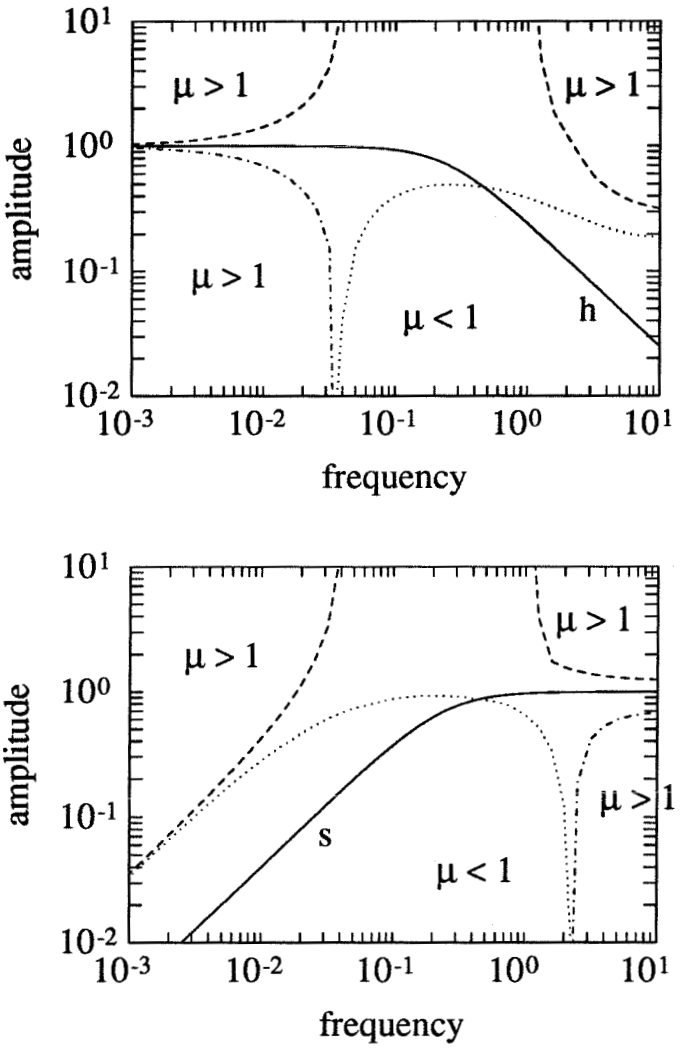


Figure 5.5: Loopshaping bounds on $\hat{H} = hI$ and $\hat{S} = sI$ for high purity distillation column. The plots for s and h are for $\lambda = 4$. The upper plot is for h and the lower plot is for s. The dashed lines are necessary upper bounds, the dashed and dotted lines are necessary lower bounds, and the dotted lines are sufficient upper bounds.

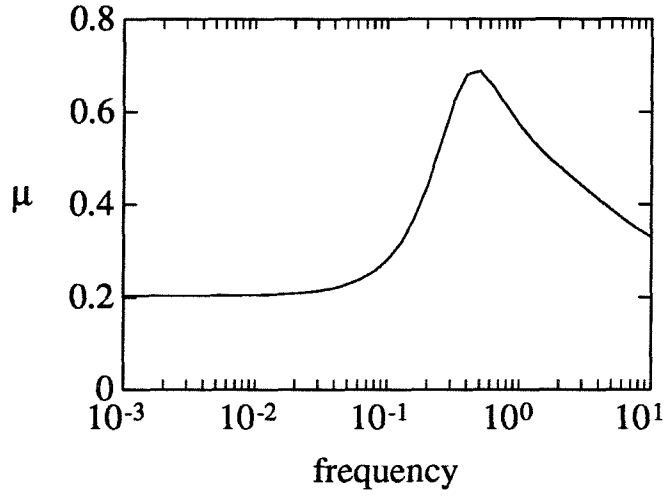


Figure 5.6: Robust performance test for decentralized controller with $\lambda = 4$.

must equal one. This is in agreement with the performance weight, which requires zero steady-state error.

Now look at the loopshaping bounds for $\tilde{S} = sI$ in Fig. 5.5. At low frequencies the necessary upper bound and the sufficient upper bounds converge—$s(0)$ must be zero to meet these bounds. Again, around crossover the bounds are very conservative. The necessary bounds approach one at high frequency. Since the plant is strictly proper, and the controller must be proper to be physically realizable, s must approach one for high frequencies anyway. The s shown in Fig. 5.5 meets the sufficient condition for robust performance for frequencies below about 0.45.

Combining the sufficient conditions over the different frequency ranges ensures that $\mu < 1$ for all frequency. Robust performance is guaranteed when $\mu < 1$ and nominal stability is satisfied. The plant is stable and minimum phase, so h stable implies nominal stability for the diagonal plant $\tilde{P}$. We calculated the closed loop poles using the full plant P and found that they are in the left half plane, so we have nominal stability. The plot of μ in Fig. 5.6 agrees that robust performance is satisfied.

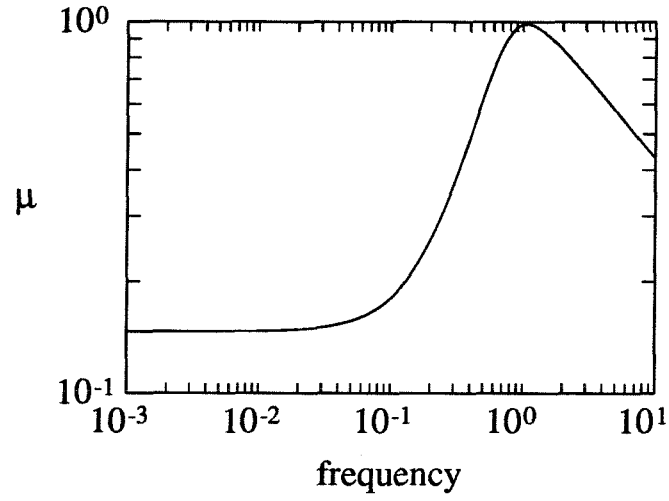


Figure 5.7: Robust performance test for decentralized controller with $\lambda = 1.8$.

The controller corresponding to $\tilde{S} = sI$ and $\tilde{H} = hI$ is

$$K = (\tilde{S}\tilde{P})^{-1}(I - \tilde{S}) = \frac{75s + 1}{4s} \begin{bmatrix} -\frac{1}{0.878} & 0 \\ 0 & -\frac{1}{0.014} \end{bmatrix}. \tag{5.16}$$

Fig. 5.6 shows the conservatism of the sufficient bounds. The design that just meets the sufficient bounds gives $\mu = 0.7$. The conservatism of the sufficient bounds was predicted by the large difference between the necessary and the sufficient upper bounds at crossover. This large difference suggests not to bother with meeting the sufficient conditions for frequencies near crossover.

The conservatism in the design was removed by increasing speed of response. Using $h = 1/(1.8s + 1)$, $s = 1.8s/(1.8s + 1)$ gives $\mu = 0.99$ (see Fig. 5.7). The loopshaping plots in this case are in Fig. 5.8. The sufficient bounds are not met for $0.7 < \omega < 1.5$.

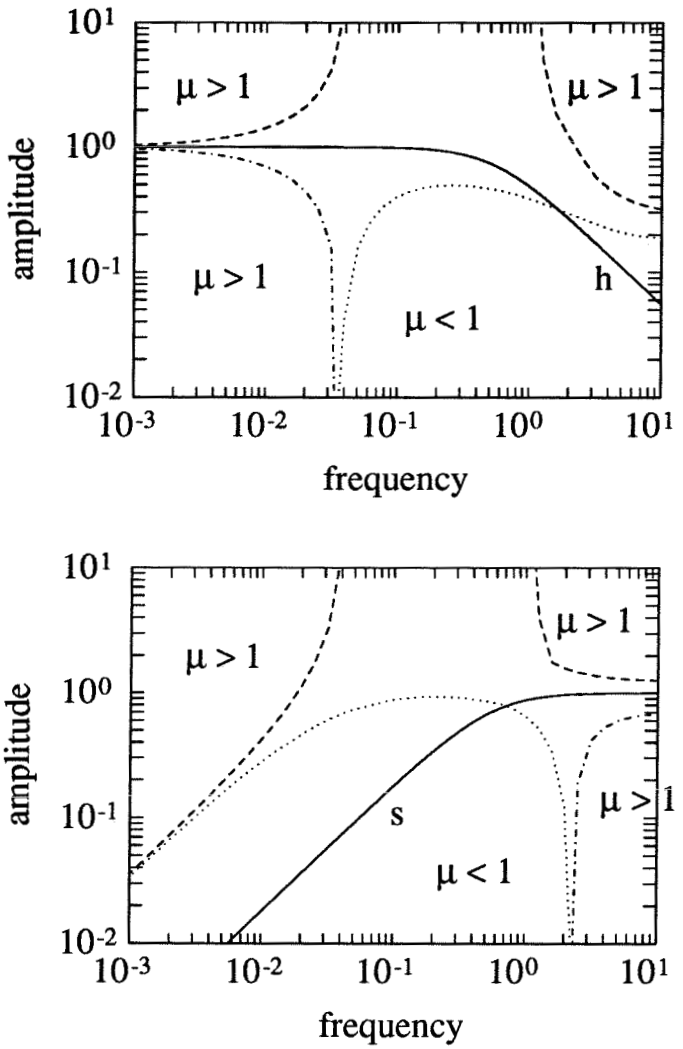


Figure 5.8: Loopshaping bounds on $\hat{H} = hI$ and $\hat{S} = sI$ for the high purity distillation column. The plots for s and h are for $\lambda = 1.8$. The upper plot is for h and the lower plot is for s. The dashed lines are necessary upper bounds, the dashed and dotted lines are necessary lower bounds, and the dotted lines are sufficient upper bounds.

5.3 Multiple Performance Specifications

The control engineer is often interested in meeting multiple performance specifications. For example, one might want a controller that remains stable under slow sensor drift or variations in actuator or sensor gain (this is referred to as *fault tolerance*). If there are sensors or actuators that are prone to failures, then we would like to specify that the closed loop system remain stable or satisfy some minimum performance whenever these sensors/actuators fail (this is referred to as *failure tolerance*).

Often the designer prefers to give specifications not in terms of robust performance, but in terms of nominal performance plus robust stability. For example, the specifications for the 1990-92 ACC Benchmark Problem [9, 10] are that the overshoot and settling time should be minimized for the nominal plant, and that stability is satisfied for some set of plants. Separate specifications are used whenever the designer expects that an overall robust performance specification will lead to an overly conservative design. Multiple performance specifications are easy to handle using loopshaping—the bounds are calculated individually for each specification, and the most restrictive bounds are used for loopshaping.

5.4 Fault Tolerance

Fault tolerance refers to the ability of the control system to meet some performance specifications even when actuators and sensors become faulty. Fault tolerance specifications can be included through an additional μ condition. Below we show how to do this for the commonly occurring faults of gain variation and slow drift.

Gain Variation To develop a system that maintains a given performance even under gain variation in the actuators or the sensors, just treat the gain variation as real parametric uncertainty.

Below we show how to treat actuator gain variations for two cases: 1) without additional uncertainty, and 2) with additional uncertainty. A similar development can be done for sensor gain variations or for combined variations in actuator and sensor gains. For stability of fully-decentralized control systems without additional uncertainty, sensor gain variations are equivalent to actuator gain variations.

The nominal controller is defined to be $K(s)$. Then the controller with gain variation can be described by $\hat{K}(s) = EK(s)$, where $E = \text{diag}\{\epsilon_i\}$, and $\epsilon_{i,low} \leq \epsilon_i \leq \epsilon_{i,high}$. We can write the set of E described by the gain variation as $E = \bar{E} + W_r\Delta^r$, where $\bar{E} = \text{diag}\{\bar{\epsilon}_i\}$, $W_r = \text{diag}\{w_i\}$,

$$\bar{\epsilon}_i = \frac{\epsilon_{i,high} + \epsilon_{i,low}}{2}, \tag{5.17}$$

$$w_i = \frac{\epsilon_{i,high} - \epsilon_{i,low}}{2}, \tag{5.18}$$

and Δ^r is a diagonal Δ-block with real independent uncertainties.

Standard block diagram manipulations are used to arrive at the $M - \Delta$ block structure in Fig. 2.1, where $\Delta = \Delta^r$ and

$$M = -(I + K(s)P(s)\bar{E})^{-1}K(s)P(s)W_r. \tag{5.19}$$

Stability is obtained for all variations in gain if and only if $\mu_{\Delta^r}(M) < 1$.

To design such controllers via loopshaping, we need to have the expression for the G matrix in Fig. 5.2. This matrix is

$$G = \begin{bmatrix} 0 & I \\ -PW_r & -P\bar{E} \end{bmatrix}. \tag{5.20}$$

The N^T matrices needed for calculating loopshaping bounds are determined using (5.3)-(5.8).

If we are interested in maintaining stability or performance with respect to other

perturbations, then the expressions for M and G are somewhat more complicated. The designer should avoid asking for the full performance under large variations in actuator/sensor gains; otherwise the designed controller will be conservative, i.e., will perform sluggishly even when the actuators and sensors behave perfectly. Let the original system be described by $\hat{G}(s)$ with uncertainty $\hat{\Delta}$.

The new Δ matrix is $\Delta = \text{diag}\{\hat{\Delta}, \Delta^r\}$. The new M matrix is

$$M = \begin{bmatrix} \hat{G}_{11} + \hat{G}_{12}\bar{E}K(I - \hat{G}_{22}\bar{E}K)^{-1}\hat{G}_{21} & \hat{G}_{12}(I + \bar{E}K(I - \hat{G}_{22}\bar{E}K)^{-1}\hat{G}_{22})W_r \\ K(I - \hat{G}_{22}\bar{E}K)^{-1}\hat{G}_{21} & K(I - \hat{G}_{22}\bar{E}K)^{-1}\hat{G}_{22}W_r \end{bmatrix}. \tag{5.21}$$

The new G matrix is

$$G = \begin{bmatrix} \hat{G}_{11} & \hat{G}_{12}W_r & \hat{G}_{12}\bar{E} \\ 0 & 0 & I \\ \hat{G}_{21} & \hat{G}_{22}W_r & \hat{G}_{22}\bar{E} \end{bmatrix}. \tag{5.22}$$

Slow Drift It is quite common for a sensor reading to slowly drift. This slow drift does not affect closed loop stability (provided the measurement sensitivity [gain] is unchanged), and can be treated as a slow disturbance at the output of the plant that must be rejected by the controller. This is included as an additional specification in defining robust performance. The disturbance weight is chosen to have higher gain at low frequency and a time constant approximately equal to the time constant of the sensor drift. For example, the disturbance weight could be chosen to be

$$w_d(s) = M\frac{(\tau_{drift}/10)s + 1}{(\tau_{drift})s + 1}, \tag{5.23}$$

where M is the magnitude and τ_{drift} is the time constant of the sensor drift. Fig. 5.9 is a bode magnitude plot of the disturbance weight for $M = 0.2$ and $\tau_{drift} = 10$.

Another reasonable choice for the disturbance weight is the integrator

$$w_d(s) = \frac{M}{s}. \tag{5.24}$$

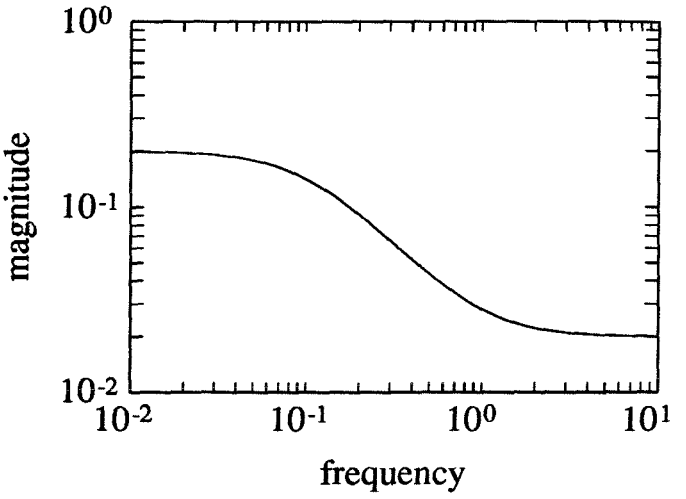


Figure 5.9: A disturbance weight to describe slow sensor drift.

5.5 Failure Tolerance

Failure tolerance refers to the ability of the control system to meet some (weaker than normal) performance specifications even though a prespecified set of actuators and sensors fail. Typically we will design the control system to be failure tolerant to only those actuators and sensors which we suspect might fail—otherwise the designed controller could be overly conservative.

The first step in designing a failure tolerant control system is to specify which sensor/actuator combinations are expected to fail. Then a performance specification is chosen for each set of sensor/actuator failures. Sometimes the requirement on the failed system is only that the closed loop remains stable. Once the different performance specifications are set, then robust loopshaping bounds can be calculated for each separate μ problem and the most restrictive robust loopshaping bounds are used to design the controller. This approach will be illustrated through an example in Section 5.6.

Below we define a very strong notion of failure tolerance in which closed loop stability is required for any combination of actuator failures. We then extend this

notion to uncertain systems. Similar definitions can be given for sensor failures.

Integrity *Integrity* is defined as follows [17].

Definition 5.1 *The closed loop system demonstrates* integrity *if* $\tilde{K}(s) = EK(s)$ *stabilizes* $P(s)$ *for all* $E \in \mathcal{E}_{1/0}$ *where*

$$\mathcal{E}_{1/0} \equiv \{E = \text{diag}(\epsilon_i) \mid \epsilon_i \in \{0,1\}, i = 1, \ldots, n\}\,. \tag{5.25}$$

Note that for a system to demonstrate integrity, the plant $P(s)$ must be stable.

A closed loop system which demonstrates integrity remains stable as subsystem controllers are arbitrarily brought in and out of service. Note that when the controller is unstable then integrity does not imply sensor or actuator failure tolerance unless the failure is recognized and the affected control loop taken out of service.

It is clear that whether a system demonstrates integrity can be tested through 2^n stability (eigenvalue) determinations.

Robust Integrity We can generalize the definition of integrity to include robustness. *Robust integrity* is defined below.

Definition 5.2 *The closed loop system demonstrates* robust integrity *if the system is stable with* $\tilde{K}(s) = EK(s)$ *for all* $E \in \mathcal{E}_{1/0}$ *and all* $\|\Delta\|_\infty \leq 1$ *where*

$$\mathcal{E}_{1/0} \equiv \{E = \text{diag}(\epsilon_i) \mid \epsilon_i \in \{0,1\}, i = 1, \ldots, n\}\,. \tag{5.26}$$

Note that for a system to demonstrate robust integrity, the plant must be stable under all allowed perturbations. Note also that robust integrity implies integrity.

A closed loop system which demonstrates robust integrity remains robustly stabilized as subsystem controllers are arbitrarily brought in and out of service. Robust integrity does not imply sensor or actuator failure tolerance unless the failure is rec-

ognized and the affected control loop taken out of service.

It is clear that whether a system demonstrates robust integrity can be tested through 2^n nominal stability (eigenvalue) and 2^n robust stability (μ) calculations.

5.6 Fault and Failure Tolerance

A very strong notion of fault tolerance was defined by Campo and Morari [17] for fully-decentralized controllers. The requirement is that the closed loop system remains stable under arbitrary detuning of the controller gains. For fully-decentralized control systems, this is equivalent to arbitrary detuning of the actuator/sensor gains.

Decentralized Unconditional Stability The following definition of *decentralized unconditional stability* is slightly modified from that of Campo and Morari [17].

Definition 5.3 *Assume $K(s)$ is fully-decentralized. The closed loop system is* decentralized unconditionally stable *(DUS) if $\tilde{K}(s) = EK(s)$ stabilizes $P(s)$ for all $E \in \mathcal{E}_D$ where*

$$\mathcal{E}_D \equiv \{E = \mathrm{diag}(\epsilon_i) \mid \epsilon_i \in (0,1), i = 1, \ldots, n\}. \tag{5.27}$$

Note that for DUS to make sense the plant $P(s)$ must be stable.

A closed loop system which is DUS remains stable as the gains of each controller subsystem are *independently* detuned. The following result is a computable necessary and sufficient condition for DUS.

Theorem 5.1 *Assume $K(s)$ is decentralized. Define Δ^r to be a diagonal Δ-block with independent real uncertainties. Then the closed loop system is DUS if and only if $\left(I + \frac{1}{2}K(s)P(s)\right)^{-1} K(s)P(s)$ is stable and*

$$\mu_{\Delta^r}\left(-\frac{1}{2}\left(I + \frac{1}{2}K(s)P(s)\right)^{-1} K(s)P(s)\right) \leq 1, \quad \forall\omega. \tag{5.28}$$

Proof: Let $\bar{E} = W_r = (1/2)I$ in (5.19). The conditions $\mu_{\Delta r}(-\frac{1}{2}(I + \frac{1}{2}K(s)P(s))^{-1} \cdot K(s)P(s)) \leq 1$, $\forall\omega$ and $(I + \frac{1}{2}K(s)P(s))^{-1}K(s)P(s)$ is stable ensure that the closed loop system is stable for all $\epsilon_i \in (0,1)$. QED.

The closed loop system cannot be DUS when the controller $K(s)$ has poles in the open right half plane—this is because some minimum amount of feedback is required to have closed loop stability.

To calculate loopshaping bounds to meet the μ condition in Theorem 5.1, we need the expression for the $G(s)$ matrix in Fig. 5.2. This matrix is given by (5.20) with $\bar{E} = W_r = (1/2)I$:

$$G = \begin{bmatrix} 0 & I \\ -(1/2)P & -(1/2)P \end{bmatrix}. \tag{5.29}$$

Robust Decentralized Unconditional Stability We can generalize the definition of DUS to include robustness. Clearly with arbitrary detuning of single loop controller gains it is not reasonable to ask for performance of the arbitrarily detuned system to be better than open loop. But it could be reasonable to expect that the system remains robustly stable under arbitrary detuning of single loop controller gains.

Definition 5.4 *Assume $K(s)$ is decentralized. The closed loop system is* robust decentralized unconditionally stable *(RDUS) if the system is stable with $\tilde{K}(s) = EK(s)$ for all $E \in \mathcal{E}_{1/0}$ and all $\|\Delta\|_\infty \leq 1$ where*

$$\mathcal{E}_D \equiv \{E = \mathrm{diag}(\epsilon_i) \mid \epsilon_i \in (0,1), i = 1, \ldots, n\}. \tag{5.30}$$

Note that for RDUS to be satisfied, the plant must be stable under all allowed perturbations.

The following result is a computable necessary and sufficient condition for RDUS.

Theorem 5.2 *Assume $K(s)$ is decentralized, and that the uncertain system is described by $\hat{G}(s)$ and $\hat{\Delta}$. Define Δ^r to be a diagonal Δ-block with independent real uncertainties. Then the closed loop system is RDUS if and only if $M(s)$ is stable and*

$$\mu_{\Delta}(M) \leq 1, \quad \forall \omega, \tag{5.31}$$

where $\Delta = \text{diag}\{\hat{\Delta}, \Delta^r\}$, and

$$M = \begin{bmatrix} \hat{G}_{11} + \frac{1}{2}\hat{G}_{12}K(I - \frac{1}{2}\hat{G}_{22}K)^{-1}\hat{G}_{21} & \frac{1}{2}\hat{G}_{12}(I + \frac{1}{2}K(I - \frac{1}{2}\hat{G}_{22}K)^{-1}\hat{G}_{22}) \\ K(I - \frac{1}{2}\hat{G}_{22}K)^{-1}\hat{G}_{21} & \frac{1}{2}K(I - \frac{1}{2}\hat{G}_{22}K)^{-1}\hat{G}_{22} \end{bmatrix}. \tag{5.32}$$

Proof: Let $\bar{E} = W_r = (1/2)I$ in (5.21). The conditions $\mu_{\Delta}(M) \leq 1$, $\forall \omega$ and $M(s)$ stable ensure that the closed loop system is stable for all $\epsilon_i \in (0,1)$ and $\|\hat{\Delta}\|_{\infty} \leq 1$. QED.

The closed loop system cannot be RDUS when the controller $K(s)$ has poles in the open right half plane—this is because some minimum amount of feedback is required to have closed loop stability.

To calculate loopshaping bounds to meet the μ condition in Theorem 5.2, we need the expression for the $G(s)$ matrix in Fig. 5.2. This matrix is given by (5.22) with $\bar{E} = W_r = (1/2)I$:

$$G = \begin{bmatrix} \hat{G}_{11} & (1/2)\hat{G}_{12} & (1/2)\hat{G}_{12} \\ 0 & 0 & I \\ \hat{G}_{21} & (1/2)\hat{G}_{22} & (1/2)\hat{G}_{22} \end{bmatrix}. \tag{5.33}$$

Remark 5.1 (CDUS, Part 1) *Actually, the definition of DUS given by Campo and Morari [17] requires that the system is stable for all $\epsilon_i \in [0,1]$—we will refer to this version as* CDUS *(*closed DUS*).*

When $K(s)$ is stable, a necessary and sufficient test for CDUS is given by Theorem 5.1 except with the condition $\mu < 1$ replacing $\mu \leq 1$ in (5.28). When $K(s)$ is

integral, μ in (5.28) will equal 1 at $\omega = 0$, because setting the proportional gain to zero in a controller with integral action will remove the feedback around the integrator, which will then be a limit of instability. Thus $\mu \leq 1$ in (5.28) will be a tight necessary condition for CDUS, but not necessarily sufficient. The following example shows that $\mu \leq 1$ is not sufficient:

$$P = \frac{1}{s+1}\begin{pmatrix} s & -1 \\ 1 & 1 \end{pmatrix}, \qquad K = \frac{1}{s}I. \tag{5.34}$$

It can be shown by using the Routh criterion that the above system is DUS and $\mu \leq 1$. The system is not CDUS because Loop #1 is not stable (for any *ϵ_1) when Loop #2 is open (due to a pole-zero cancellation at $s = 0$).*

CDUS can *be checked through a finite number of stability and μ tests, by using Theorem 5.1 to check the interior of the ϵ-hypercube, and testing the boundary (the points, edges, faces, etc.) through additional μ tests. The number of μ tests required grows rapidly with the size of the system. Alternative approaches are being investigated to see if CDUS can be evaluated using a single test.*

CRDUS can be defined similarly, and a similar discussion applies as for CDUS.

Remark 5.2 (CDUS, Part 2) *Nwokah and co-workers [62, 63, 81, 82, 83] consider conditions under which the controller $K(s) = (1/s)I$ together with $P(s)$ is CDUS. They claim (Theorem 3 of [62, 63], Theorem 1 of [81], Theorem 5.1 of [82], and Theorem 7 of [83]) that a* necessary *condition for $K(s) = (1/s)I$ to provide CDUS is that $P(0)$ is* all gain positive stable. *A matrix P is* all gain positive stable *if P, P^{-1}, and all their corresponding principal submatrices are* D-stable. *A matrix P is* D-stable *if* $\mathrm{Re}\{\lambda_i(PD)\} > 0$, $\forall i$, $\forall D > 0$, *where D is real and diagonal.*

The following plant (from [17]) illustrates that the condition by Nwokah and co-workers is not necessary*:*

$$P(s) = \begin{bmatrix} 1 & 0 & 2 \\ \frac{1}{s+1} & 1 & \frac{-4s}{s+1} \\ 0 & 4 & 1 \end{bmatrix}. \tag{5.35}$$

It can be shown via the Routh-Hurwitz stability criteria that the closed loop system for the above plant is stable for $K(s) = (1/s)I$ *and remains stable with arbitrary detuning of the SISO loop gains. The eigenvalues of* $P(0)$ *are* $\{\pm i\sqrt{3}, 3\}$*, so* $P(0)$ *is not D-stable, and* $P(0)$ *is not all gain positive stable.*

Robust Decentralized Detunability Detuning a controller refers to changing some parameter in the controller or in the control synthesis procedure so that the control action becomes less aggressive. For example, in Linear Quadratic Control detuning refers to increasing the magnitude of the control weight. In decentralized Internal Model Control, detuning refers to increasing the IMC filter time constants in each single loop controller [51]. The special case of detuning the single loop controller gains in a decentralized controller was discussed earlier in the sections on DUS and RDUS.

Hovd [51] introduced the following very general definition for *robust decentralized detunability.*

Definition 5.5 For a given design method, *a closed loop system is* robust decentralized detunable *(RDD) if each single loop controller can be detuned independently by an arbitrary amount without endangering robust stability.*

Whenever the controller is detuned by varying parameters in the controller, RDD can be tested via a μ test where the variation in parameters is covered by real uncertainty (the real uncertainty must be *independent* for *arbitrary* detuning). Both the robust performance and the "RDD" loopshaping bounds are plotted and the most restrictive

of the bounds are used in the design. The resulting controller meets robust performance and gives a system which is RDD. The procedure is illustrated in Section 5.7 below, where a decentralized controller is designed via loopshaping $\tilde{H}$ and $\tilde{S}$. *For this design procedure*, the closed loop system is RDD if the time constants for the single loop controllers can be increased independently by an arbitrary amount without endangering robust stability.

5.7 Example: Fault/Failure Tolerant Decentralized Controller Design

Description We will use the loopshaping bounds to design a robust fault/failure tolerant decentralized controller for the high-purity distillation column discussed in Section 5.2. The performance and uncertainty specifications are the same here, but we add fault/failure tolerance specifications. First we will design the controller so that the closed loop system is RDD. Then we test that the resulting closed loop system demonstrates integrity, robust integrity, DUS, and RDUS.

RDD To design for RDD, we plot in Fig. 5.10 the loopshaping bounds for robust stability where Δ_T is chosen to be a diagonal block with independent elements (the bounds are calculated by applying the Theorem 4.1 on the appropriate submatrices of $N^{\tilde{H}}$ and $N^{\tilde{S}}$ in (5.13) and (5.14)).

The closed loop system is RDD if the system remains robustly stable as the controller is dynamically detuned. Dynamic detuning for this example refers to increasing the single loop closed loop time constants λ. A careful consideration of Fig. 5.10 shows that either the sufficient bound on h or the sufficient bound on s is satisfied for all frequencies for all $\lambda_i \geq 1.8$; thus the system given by $\lambda = 4$ is RDD. A less conservative bound on the λ_i can be derived by directly loopshaping λ (see [51] for details),

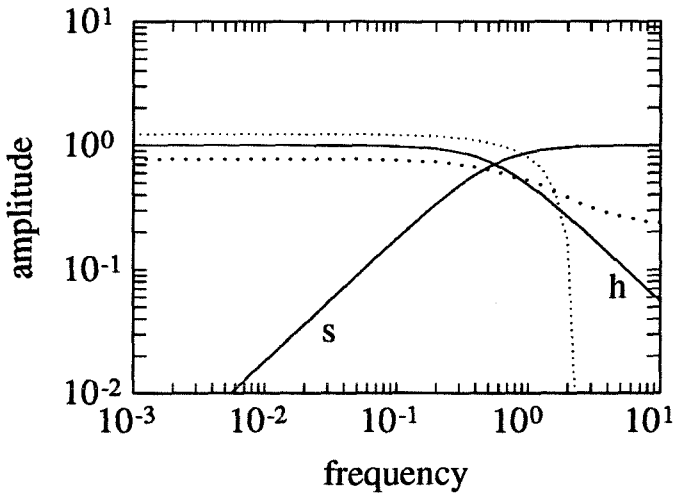


Figure 5.10: Robust stability loopshaping bounds for fault/failure tolerant decentralized control of a high purity distillation column. The solid lines are h and s for $\lambda = 1.8$. The widely spaced dotted line is the sufficient bound for h. The thinly spaced dotted line is the sufficient bound for s.

but deriving the bounds using h and s allows a direct comparison of the robust performance bounds in Fig. 5.5 and the RDD robust stability bounds in Fig. 5.10. This comparison shows that the bounds in Fig. 5.5 are more restrictive, so these can be used to loopshape the controller. The robust performance bounds are not necessarily more restrictive in general.

We will now test the closed loop system with the designed controller to ensure that is satisfies integrity, robust integrity, DUS, and RDUS.

Integrity The following four transfer functions are stable for $\lambda = 4$:

$$P, \qquad [(\epsilon_1, \epsilon_2) = (0,0)]; \tag{5.36}$$

$$M_{11} = -w_I P^{-1} H P, \qquad [(\epsilon_1, \epsilon_2) = (1,1)]; \tag{5.37}$$

$$-w_I K_1 (1 + P_{11} K_1)^{-1} P_{11}, \qquad [(\epsilon_1, \epsilon_2) = (1,0)]; \tag{5.38}$$

$$-w_I K_2 (1 + P_{22} K_1)^{-1} P_{22}, \qquad [(\epsilon_1, \epsilon_2) = (0,1)]; \tag{5.39}$$

thus the system has integrity.

Robust Integrity To test robust integrity for a 2×2 system, we need to check robust stability for four failure conditions. Nominal stability was tested above (for testing integrity), so we need test only the μ conditions here.

We have robust stability when all loops are turned off provided $P(I + w_I\Delta_I)$ is stable. That $P(I + w_I\Delta_I)$ is stable follows since P, w_I, and Δ_I are stable.

Robust stability for the overall system is satisfied since $\mu_{\Delta_I}(M_{11}) = 0.3 < 1$. Robust stability for the cases when exactly one loop has failed is satisfied since

$$\mu_{\Delta_{I,11}}(-w_I K_1(1 + P_{11}K_1)^{-1}P_{11}) = 0.12 < 1, \qquad [(\epsilon_1, \epsilon_2) = (1, 0)]; \quad (5.40)$$

$$\mu_{\Delta_{I,22}}(-w_I K_2(1 + P_{22}K_2)^{-1}P_{22}) = 0.12 < 1, \qquad [(\epsilon_1, \epsilon_2) = (0, 1)]. \quad (5.41)$$

Since all four μ conditions are satisfied, the system demonstrates robust integrity.

DUS, RDUS Since DUS is implied by RDUS, we will only test RDUS here.

The $\hat{G}$ and $\hat{\Delta}$ matrices needed to apply Theorem 5.2 are derived directly from the block diagram in Fig. 5.4:

$$\hat{G} = \begin{bmatrix} 0 & -w_I I \\ P & -P \end{bmatrix}, \qquad \hat{\Delta} = \Delta_I. \quad (5.42)$$

Fig. 5.11 is the μ plot to test condition (5.31). As expected, the value μ approaches 1 at zero frequency. We see that $\mu << 1$ for all frequencies away from $\omega = 0$. As expected, μ rapidly increases towards 1 at very low frequencies because the integrators cause a stability problem here as either of the ϵ_i approach zero. Since $\mu \leq 1$, the system demonstrates RDUS.

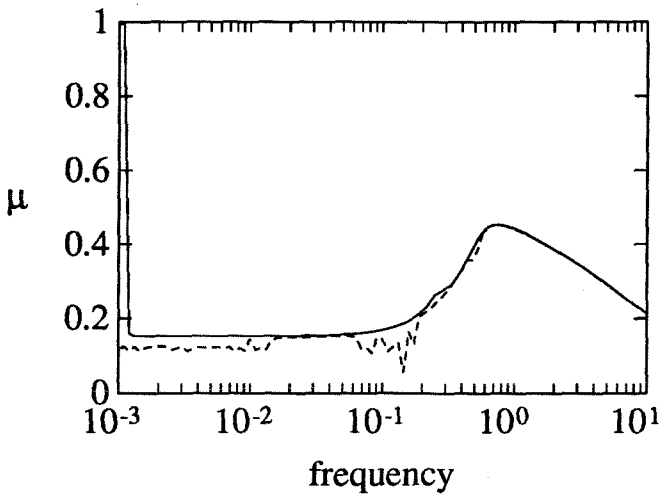


Figure 5.11: Test for RDUS (upper and lower bounds shown).

RDD vs. RDUS Let us look at the set of controllers which are given by dynamically detuning the system. This set is

$$K = (\tilde{S}\tilde{P})^{-1}(I - \tilde{S}) = \frac{75s+1}{s}\begin{bmatrix} -\frac{1}{0.878\lambda_1} & 0 \\ 0 & -\frac{1}{0.014\lambda_2} \end{bmatrix}. \tag{5.43}$$

We see that for this example dynamically detuning the system exactly corresponds to decreasing the single loop controller gains. Thus, for this example, RDD is equivalent to the interior μ test (5.31) for RDUS. This will not be true in general.

5.8 Conclusions

Decentralized controllers are the rule rather than the exception in industrial process control. We have shown how to analyze the reliability of control systems, and use robust loopshaping to design decentralized controllers which are inherently reliable to equipment faults or failures. These techniques are illustrated on a high purity distillation column.

Chapter 6

Control Structure Selection

Summary

The robust loopshaping framework is used to develop tools for choosing actuators and sensors to use for control purposes in the presence of model/plant mismatch. In decentralized controller design, the tools are also used for determining the appropriate partitions and pairings of controller inputs and outputs. New results are presented, as well as simplified and unified proofs of existing results. A branch-and-bound procedure for control structure selection can greatly reduce the number of candidates from further consideration. The tools developed in this chapter can also provide recommendations on how to modify the *plant design* to improve the closed loop performance.

6.1 Introduction

Practical control problems often involve more actuators and sensors than are needed for designing effective, economically viable control systems. On a distillation column, for example, there are at least four actuators and as many temperature measurements as the number of trays, that can be utilized for composition control. In practice, one

does not use all the available actuators and sensors since two of the four actuators must be used for inventory control and a control system based on all the tray temperatures will be unnecessarily complex and expensive. An appropriate set of actuators and sensors must be selected from the available candidates.

An additional consideration is that we may be interested in using decentralized controllers. Then we also need to choose the appropriate *partitions* and *pairings* of inputs and outputs. *Control structure selection* refers to both choosing the actuators and sensors, and choosing the appropriate partitions and pairings. Descriptions of control structure selection research to date are provided by Morari and Zafiriou [76] and Lee et al. [64].

Researchers have especially studied subsets of the *control structure selection* problem. This includes secondary measurement selection [60, 47, 117, 67], the decentralized integral controllability problem as studied by Morari and coworkers [71, 44, 76], and the "selection" and "partitioning" problems [78, 79]. In this chapter we consider the general control structure selection problem. By approaching the general problem, the results also apply to the subset problems.

Until recently, tools for control structure selection did not take plant/model mismatch into account. Numerous process examples are provided in [76, 64, 106, 67] which show that ignoring or improperly characterizing plant/model mismatch can lead to erroneous results, thus motivating the need for the tools developed in this chapter.

Framework For Control Structure Selection At this point, let us consider an approach to control structure selection illustrated in Fig. 6.1. The transfer function G^{ij} refers to the submatrix of the overall generalized plant matrix G with rows and columns corresponding to a specified subset of the available actuators and sensors. The controller K may be assumed to have some structure (e.g., decentralized), and

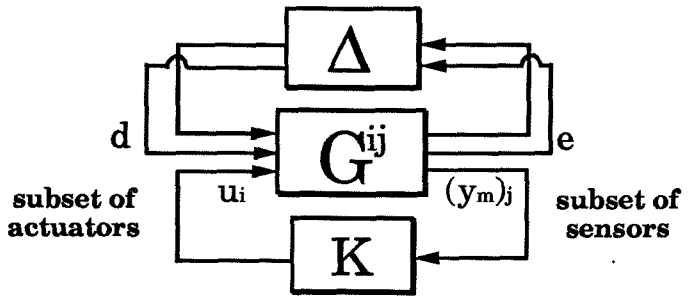


Figure 6.1: Block diagram for control structure selection.

the rows and columns of G^{ij} are rearranged to correspond to the pairing being considered. *Control structure candidates* consist of all possible combinations of the available actuators and sensors and pairings and partitions. Owing to the combinatorial nature of the problem, the number of candidates is often very large. Naturally, a method to reduce the number of candidates before applying detailed analyses is of significant practical value.

The first proposed step is to eliminate the candidates for which a controller achieving a desired level of robust performance does not exist regardless of what controller design method is used. The criteria that can be used to accomplish this screening will be referred to as *general screening tools.* This screening process leaves candidates for which a control system with satisfactory performance may potentially exist. However, this alone may not reduce the number of candidates to a low enough level. Also, it is not clear if control design methods available to the engineer can lead to a controller achieving the desired performance. Hence, an additional screening may be carried out subsequently in the context of a chosen design approach. That is, one may choose to further eliminate those candidates for which the particular design approach under consideration does not yield a controller achieving a desired level of robust performance. The criteria that can be used under a particular design approach will be called *design-dependent screening tools.*

All screening tools which allow a general uncertainty description, both new and

old, are derived via the robust loopshaping framework.

Organization The rest of this chapter is organized as follows. First we derive the screening tools which do not depend on the partitioning or pairing—these are referred to as *pairing-independent screening tools.* These tools are useful during initial screening, since they can remove from further consideration *all* control structure candidates associated with a given set of actuators and sensors. Then we derive tools which are dependent on the partitioning and pairing, which we refer to as *pairing-dependent screening tools.* This chapter connects up many tools which researchers previously considered independently—including those based on loopshaping, interaction measures, and decentralized integral controllability—and derives and extends these tools via a unified framework. Most proofs consist of only a few lines.

6.2 Basis for Control Structure Selection

Fig. 6.1 represents the general block diagram for linear systems with uncertainty. We would like to test if robust performance can be achieved with the ith set of actuators and the jth set of sensors, and with a choice of pairings and partitions which are given by the decentralized structure of K (the rows and columns of G^{ij} can always be rearranged so that K is block-diagonal). Mathematically, we would like to test if

$$\inf_{K \in \mathcal{K}_S} \sup_{\omega} \mu_{\Delta}\Big(F_l(G^{ij}, K)\Big) < k, \tag{6.1}$$

where $k = 1$ and $\mathcal{K}_S$ is the set of stabilizing controllers with given structure. There is no computable necessary and sufficient test for (6.1). This provides the motivation for developing computable necessary conditions for robust performance. These necessary conditions are used as screening tools which remove control structure candidates from further consideration.

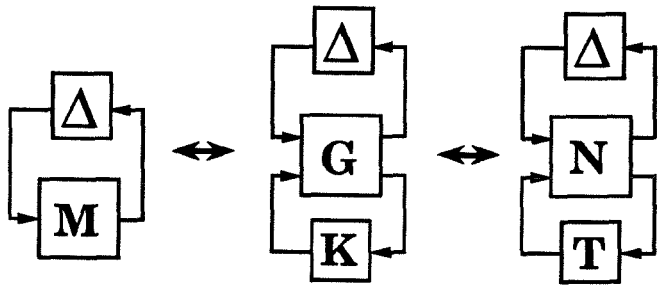


Figure 6.2: Equivalent representations of system M with perturbation Δ. The transfer function T is chosen to be a parametrization of the controller K.

For notational purposes we will drop the superscript to G^{ij} for the remainder of this chapter. Recall that the generalized plant G in Fig. 6.2 is found directly by rearranging the system's block diagram or by standard software [3]. Also recall the definitions of $\tilde{P}$, $\tilde{S}$, and $\tilde{H}$ given in the previous chapter, but with the obvious extensions to include block-diagonal controllers.

6.3 Pairing-Independent Screening Tools

Screening tools which do not depend on the partitioning or pairing of the control loops are referred to as *pairing-independent screening tools.* Screening tools for loopshaping design which were originally developed by Lee and Morari [67] are derived. These tools are shown to be strongly related to new general screening tools which are appropriate when the open loop transfer function is strictly proper, or when integral action is required—both of which are common requirements in practice.

Loopshaping Design Design-specific screening tools can be derived for existence of a controller *designed by loopshaping* to meet robust performance.

Recall from the two previous chapters that to design controllers via robust loopshaping, we need to find an LFT of T which describes M (see Fig. 6.2), where T parametrizes the controller we are to design. When designed decentralized controllers

in the previous chapter, T was chosen to be the diagonal sensitivity $\tilde{S}$ or the diagonal complementary sensitivity $\tilde{H}$. The nominal closed loop transfer functions are designed to meet sufficient conditions on the (block-)diagonal sensitivity $\tilde{S}$ for low frequency and sufficient conditions on the (block-)diagonal complementary sensitivity $\tilde{H}$ for high frequency. This directly leads to the following screening tools [67].

Theorem 6.1 (Screening Tool for Loopshaping with $\tilde{S}$) *Let $M = F_l(N,T) = N_{11} + N_{12}T(I - N_{22}T)^{-1}N_{21}$, and k be a given constant. There exists a controller designed via loopshaping $\tilde{S}$ that satisfies $\mu_\Delta(M) < k$ only if*

$$\mu_\Delta\Big(G_{11}(0) + G_{12}(0)P^{-1}(0)G_{21}(0)\Big) < k. \tag{6.2}$$

Proof: In loopshaping design, $\tilde{S}$ must satisfy the sufficient bound for low frequency. We know from the proof of Theorem 4.1 that the sufficient bound exists if and only if assumptions $(i - iii)$ hold. Assumption (ii) is that $\mu_\Delta\Big(N_{11}^{\tilde{S}}\Big) < k$. The expression for $\mu_\Delta\Big(N_{11}^{\tilde{S}}\Big)$ in terms of G and P is given by (5.7). QED.

Theorem 6.2 (Screening Tool for Loopshaping with $\tilde{H}$) *Let $M = F_l(N,T) = N_{11} + N_{12}T(I - N_{22}T)^{-1}N_{21}$, and k be a given constant. There exists a controller designed via loopshaping $\tilde{H}$ that satisfies $\mu_\Delta(M) < k$ only if*

$$\mu_\Delta(G_{11}(j\infty)) < k. \tag{6.3}$$

Proof: Similar to the proof of Theorem 6.1.

The above conditions [(6.1) and (6.2)] are required for the existence of the robust loopshaping bounds at the appropriate frequencies. It is interesting that, though the

magnitude of the sufficient upper loopshaping bounds (when they exist) are *dependent* on the partitioning/pairing, the *existence* of the sufficient upper loopshaping bounds is *independent* of the partitioning/pairing. The existence of the necessary loopshaping bounds is *dependent* on the partitioning/pairing, and later in this chapter it is shown that this leads to screening tools useful for choosing between decentralized control structures [via (6.7)].

Note that $\mu_\Delta(G_{11})$ is the value of μ at *open loop* ($K = 0$). Since the controller K does not enter into condition (6.3), this condition cannot aid in control structure selection. Therefore it can be argued that calling Theorem 6.2 a *screening tool* is a misnomer. However, since condition (6.3) *is* a valid necessary condition for robust performance to be achieved, with a slight abuse of notation we will refer to it as a screening tool. Examples in Section 6.5 illustrate the use of this condition to rule out inappropriate choices of uncertainty and performance weights.

General Screening Tools New general screening tools are derived directly from the necessary bounds for robust performance.

Theorem 6.3 (Screening Tool for Integral Controllers) *Let* $M = F_l(N,T) = N_{11} + N_{12}T(I - N_{22}T)^{-1}N_{21}$, *and* k *be a given constant. There exists a controller with integral action in all channels that satisfies* $\mu_\Delta(M) < k$ *only if*

$$\mu_\Delta\Big(G_{11}(0) + G_{12}(0)P^{-1}(0)G_{21}(0)\Big) < k. \tag{6.4}$$

Proof: For integral action in all channels, $\tilde{S}(0) = 0$. To have $\mu_\Delta(M) < k$ at zero frequency, we need $\mu_\Delta(M(0)) = \mu_\Delta\Big(F_l(N^{\tilde{S}}(0), \tilde{S}(0))\Big) = \mu_\Delta\Big(F_l(N^{\tilde{S}}(0), 0)\Big) = \mu_\Delta\Big(N_{11}^{\tilde{S}}(0)\Big) = \mu_\Delta(G_{11}(0) + G_{12}(0)P^{-1}(0)G_{21}(0)) < k$. QED.

The above screening tool has a nice interpretation when the steady-state robustness requirements hold at open loop, i.e., $\mu_\Delta(G_{11}) < k$, and the controlled variable is equal to the measured variable plus disturbances, i.e., $G_{22} = -P$. The former condition would hold, for example, if robust stability was being considered and all plants given by the uncertainty description were open loop stable. If $\mu_\Delta(G_{11}) < k$ and $G_{22} = -P$ then it can be shown using the matrix inversion lemma [87] that (6.4) holds if and only if

$$\det(F_u(G, \Delta)) \neq 0, \quad \forall \|\Delta\|_\infty \leq 1/k. \tag{6.5}$$

This requirement is that the determinant of all plants given by the uncertainty description have the same sign. That this condition is necessary for the existence of an integral controller which stabilizes the set of plants can also be shown using the closed loop characteristic equation.

All real systems have vanishingly small gain at high frequencies, i.e., transfer functions describing system components should be strictly proper. If the product of $\tilde{P}$ and K are strictly proper, then $\tilde{H}(\infty j) = 0$. This leads to the following screening tool.

Theorem 6.4 (Screening Tool for Strictly Proper Open Loop Systems) *Let $M = F_l(N, T) = N_{11} + N_{12}T(I - N_{22}T)^{-1}N_{21}$, and k be a given constant. There exists a controller with $\tilde{P}K$ strictly proper that satisfies $\mu_\Delta(M) < k$ only if*

$$\mu_\Delta(G_{11}(j\infty)) < k. \tag{6.6}$$

Proof: If $\tilde{P}K$ is strictly proper, then $\tilde{H}(\infty) = 0$. To have $\mu_\Delta(M) < k$ at $\omega = \infty$, we need $\mu_\Delta(M(j\infty)) = \mu_\Delta\Big(F_l(N^{\tilde{H}}(j\infty), \tilde{H}(j\infty))\Big) = \mu_\Delta\Big(F_l(N^{\tilde{H}}(j\infty), 0)\Big) = \mu_\Delta\Big(N_{11}^{\tilde{H}}(j\infty)\Big) = \mu_\Delta(G_{11}(j\infty)) < k$. QED.

If $\tilde{P}$ is strictly proper (proper), then Theorem 6.4 gives a necessary condition for the existence of a proper (strictly proper) controller which achieves robust performance.

It is interesting that the general screening tool for controllers with integral action is the same as the design-specific screening tool when designing via loopshaping $\tilde{S}$. Also, the general screening tool for strictly proper open loop systems is the same as the design-specific screening tool when designing via loopshaping $\tilde{H}$. As discussed earlier, though condition in Theorem 6.4 does not depend on the effect of the controller and so is not useful for control structure selection, we will continue to refer to it as a screening tool with a slight abuse of notation.

6.4 Pairing-Dependent Screening Tools

Screening tools which depend on the partitioning or pairing of the control loops are referred to *pairing-dependent screening tools.* First we show that the existence of the necessary robust loopshaping bounds is equivalent to previously-derived general screening tools [65, 64, 66]. We then derive via the robust loopshaping framework tools for control structure selection based on interaction measures, and show that these are strongly related to decentralized integral controllability measures.

General Screening Tools A necessary condition that (6.1) is satisfied is for

$$\inf_{K \in \mathbf{\Delta}_K} \mu_{\Delta}\Big(F_l(G^{ij}, K)\Big) < k \tag{6.7}$$

to hold for each frequency ω, where $\mathbf{\Delta}_K$ is the set of all complex matrices with the structure of K. This condition is necessary because $\mathbf{K}_S \subset \mathbf{\Delta}_K$.

When the controller K in (6.7) is centralized then it can be parametrized by the Youla matrix Q to give $M = F_l(G^{ij}, K)$ as an affine function of Q. Replacing μ with

its upper bound (2.14) then leads to computable screening tools. It can be shown [66] that (6.7) holds for all frequencies if and only if there exists an *acausal* Q that satisfies $\mu_\Delta(M) < k$.

The following two theorems show that the existence of the necessary robust loop-shaping bounds is equivalent to (6.7), and hence the screening tools developed in [65, 64, 66] immediately follow.

Theorem 6.5 (Necessary Upper Bound and Control Structure Selection) *Let $M = F_l(N,T) = N_{11} + N_{12}T(I - N_{22}T)^{-1}N_{21}$, and k be a given constant. Assume for a given frequency ω that*

$$\begin{array}{ll} (i) & \det(I - N_{22}T) \neq 0, \text{ and} \\ (iii'') & g(\infty) > k. \end{array} \tag{6.8}$$

Then the necessary upper bound (given by Theorem 4.3) at frequency ω exists if and only if (6.7) holds at that frequency.

Proof: We see from the proof of Theorem 4.3 that assumptions (i), (ii''), and (iii'') are necessary and sufficient conditions for the existence of the necessary upper bound. Since (i) and (iii'') are assumed, to complete the proof we need to show that (ii'') holds if and only if (6.7) holds. But recall (see Fig. 4.1) that $F_l(G^{ij}, K) = F_l(N^{ij}, T)$, i.e., (6.7) and (ii'') are equivalent representations for the same problem. QED.

Theorem 6.6 (Necessary Lower Bound and Control Structure Selection) *Let $M = F_l(N,T) = N_{11} + N_{12}T(I - N_{22}T)^{-1}N_{21}$, and k be a given constant. Assume for a given frequency ω that*

$$\begin{array}{ll} (i) & \det(I - N_{22}T) \neq 0, \text{ and} \\ (iii''') & h(0) = \mu_\Delta(N_{11}) > k. \end{array} \tag{6.9}$$

Then the necessary lower bound (given by Theorem 4.4) at frequency ω exists if and only if (6.7) holds at that frequency.

Proof: Similar to the proof of Theorem 6.5. QED.

Interaction Measures The screening tools thus far measured the suitability of a control structure candidate solely in terms of robust performance. Sometimes further conditions are important when judging the suitability of a candidate. For example, tolerance of the resulting closed loop system to failures in actuators and sensors should be considered. More specifically, it may be desirable that the closed loop system will remain stable as any subset of loops can be detuned or taken out of service (put on "manual"). Clearly for such a closed loop system to exist, the plant P must be stable, so we will assume this in what follows.

An interaction measure indicates the effect of off-diagonal blocks of the plant on the performance of the decentralized controller. Grosdidier and Morari defined the μ *interaction measure* [43, 42] as

$$\mu^{-1}_{\Delta_{\tilde{H}}}\left(I - P(j\omega)\tilde{P}^{-1}(j\omega)\right), \tag{6.10}$$

where $\Delta_{\tilde{H}}$ has the same structure as $\tilde{H}$. The strength of the μ interaction measure is based on the following theorem, which we will prove via the robust loopshaping framework.

Theorem 6.7 (μ Interaction Measure) *Assume P is stable, and that a decentralized controller K is designed which stabilizes the block-diagonal plant $\tilde{P}$. Then the closed loop system is stable if*

$$\overline{\sigma}\left(\tilde{H}(j\omega)\right) < \mu^{-1}_{\Delta_{\tilde{H}}}(I - P(j\omega)\tilde{P}^{-1}(j\omega)) \quad \forall\omega. \tag{6.11}$$

Proof: If P is stable, then stability is assured if $K(I + PK)^{-1}$ is stable (this is a special case of the Youla parametrization of all stabilizing controllers [76]). From

inspection, the value for G in Fig. 6.2 is

$$G = \begin{bmatrix} 0 & I \\ I & -P \end{bmatrix}. \tag{6.12}$$

Applying (5.6) gives that

$$N^{\tilde{H}} = \begin{bmatrix} 0 & \tilde{P}^{-1} \\ I & I - P\tilde{P}^{-1} \end{bmatrix}. \tag{6.13}$$

Applying Theorem 4.1 (but using $\hat{f}(c_T)$ in (4.36) with $\Delta = 0$), and solving explicitly for $c_{\tilde{H}}^{su}(\omega)$, gives that (6.11) is a sufficient condition that the Nyquist plot of $\det(I - (I - P\tilde{P}^{-1})\tilde{H})$ does not encircle the origin. This implies that the closed loop system is stable, since $(I + PK) = (I - (I - P\tilde{P}^{-1})\tilde{H})(I + \tilde{P}K)$, and $(I + \tilde{P}K)^{-1}$ is stable by assumption. QED.

Theorem 6.7 gives conditions for which stability with the decentralized controller applied to the block-diagonal plant implies the stability of the overall system. When comparing the μ interaction measure with other interaction measures (for example, those based on diagonal dominance), Grosdidier and Morari [42] note that the μ interaction measure is *optimal*, since it provides the least conservative bound on $\overline{\sigma}(\tilde{H})$ (see Remark 4.9).

Stability is guaranteed by controllers designed by loopshaping $\tilde{H}$ based on the bound in (6.11). This controller will also have a very strong form of fault/failure tolerance—closed loop stability is maintained with the controller detuned either dynamically or statically.

A controller can always be detuned sufficiently so that (6.11) holds; thus satisfaction of the inequality in (6.11) does not directly provide a useful screening tool for controllers designed via the μ interaction measure. A useful screening tool can be obtained by also requiring that the controller have integral action. In the case, we have the following design-dependent screening tool which follows immediately from

Theorem 6.7.

Theorem 6.8 (Screening Tool for μ Interaction Measure Design) *Assume P is stable. A decentralized controller with integral action in all channels can be designed via the μ interaction measure only if*

$$\mu_{\Delta_{\tilde{H}}}(I - P(0)\tilde{P}^{-1}(0)) < 1. \tag{6.14}$$

The μ interaction measure and its screening tool do not take plant/model mismatch into account. The following theorem generalizes the μ interaction measure to handle model uncertainty.

Theorem 6.9 (Robust Interaction Measure) *Consider a system put into the general $G - K$ form in Fig. 6.2. Assume G is stable, and that a decentralized controller K is designed which stabilizes the block-diagonal plant $\tilde{P}$. Then the closed loop system is stable for all $\|\Delta\|_\infty \leq 1$ if*

$$\overline{\sigma}\left(\tilde{H}(j\omega)\right) < c_{\tilde{H}}^{su}(\omega) \tag{6.15}$$

where $c_{\tilde{H}}^{su}(\omega)$ solves

$$\mu_{\left[\begin{smallmatrix}\Delta & \\ & \Delta_{\tilde{H}}\end{smallmatrix}\right]}\left(\begin{bmatrix} G_{11}(j\omega) & G_{12}(j\omega)\tilde{P}^{-1}(j\omega) \\ c_{\tilde{H}}^{su}(\omega)G_{21}(j\omega) & c_{\tilde{H}}^{su}(\omega)(I - P(j\omega)\tilde{P}^{-1}(j\omega)) \end{bmatrix}\right) = 1. \tag{6.16}$$

Proof: If G is stable, then robust stability is assured if $K(I + PK)^{-1}$ is stable and $\mu_\Delta(F_l(G(j\omega), K(j\omega))) < 1$, $\forall\omega$. Because $c_{\tilde{H}}^{su}(\omega)$ is a lower bound to $\mu_{\Delta_{\tilde{H}}}^{-1}(I - P(j\omega)\tilde{P}^{-1}(j\omega))$, satisfaction of (6.15) implies stability as in the above theorem. Applying Theorem 4.1 gives the result. QED.

The robust interaction measure is *optimal*, i.e., it provides the least conservative bound on $\overline{\sigma}(\tilde{H})$ which guarantees robust stability of the overall system. For processes in which interactions significantly affect the performance of the block diagonal system, the bound in Theorem 6.9 will be unattainable at low frequencies. In this case, bounds on both $\tilde{S}$ and $\tilde{H}$ must be used to design the decentralized controller, and stability must be checked using separate stability conditions. This design procedure was illustrated in Chapter 5, and the screening tools given by Theorems 6.1 and 6.2 are then appropriate.

The following design-dependent screening tools follow immediately from Theorem 6.9.

Theorem 6.10 (Screening Tool #1 for Robust Interaction Measure Design) *Assume G is stable. A decentralized controller compatible with the block-diagonal plant $\tilde{P}$ which is stable for all $\|\Delta\|_\infty \leq 1$ can be designed via the robust interaction measure only if*

$$\mu_\Delta(G_{11}(j\omega)) < 1, \qquad \forall\omega. \tag{6.17}$$

Note that when the plant is stable, (6.17) is the necessary and sufficient condition for robustness to be achieved when the system is operating in open loop. Performance specifications will not be satisfied in open loop, so these should not be included in the robust interaction measure bounds. The best performance is achieved by choosing the fastest time constant on $\tilde{H}$ such that the robust stability bounds are achieved. Robust performance is checked after the design is complete.

Theorem 6.11 (Screening Tool #2 for Robust Interaction Measure Design) *Assume G is stable. A decentralized controller compatible with the block-diagonal plant*

$\tilde{P}$ which is stable for all $\|\Delta\|_\infty \leq 1$ and has integral action in all channels can be designed via the robust interaction measure only if

$$\mu_{\left[\begin{smallmatrix}\Delta & \\ & \Delta_{\tilde{H}}\end{smallmatrix}\right]}\left(\begin{bmatrix} G_{11}(0) & G_{12}(0)\tilde{P}^{-1}(0) \\ G_{21}(0) & I - P(0)\tilde{P}^{-1}(0) \end{bmatrix}\right) < 1. \tag{6.18}$$

We will now show that screening tools for the design of controllers via interaction measures are also sufficient conditions for decentralized integral controllability (defined below) and its generalization to include plant/model mismatch.

Decentralized Integral Controllability A common performance requirement is that the system rejects step disturbances at steady-state, i.e., integral control is desirable. The following property is then desirable from a practical point of view.

Definition 6.1 *A plant P is Decentralized Integral Controllable (DIC) if there exists a diagonal controller K with integral action in all channels such that $\hat{K}(s) = EK(s)$ stabilizes $P(s)$ for all $E \in \overline{\mathcal{E}}_D$ where*

$$\overline{\mathcal{E}}_D \equiv \{E = \text{diag}(\epsilon_i) | \epsilon_i \in [0,1], i = 1, \ldots, n\}. \tag{6.19}$$

If a system is DIC then it is possible to maintain stability while detuning the gain of each loop separately. DIC is a property of the plant P and the selected control structure, and it is desirable to select a control structure such that the system is DIC. No necessary and sufficient conditions for DIC is available. The most complete exposition of necessary conditions and sufficient conditions for DIC is given by Campo and Morari [17].

We will now derive a sufficient condition for DIC.

Theorem 6.12 (Sufficient Condition for DIC) *Stable P is DIC if*

$$\mu_{\Delta_{\tilde{H}}}(I - P(0)\tilde{P}^{-1}(0)) < 1, \tag{6.20}$$

where $\Delta_{\tilde{H}}$ has the block structure of $\tilde{H}$.

Proof: Equation (6.20) implies that $\det(\tilde{P}(0)) \neq 0$, so there exists a decentralized Proportional-Integral controller K which stabilizes the block-diagonal plant (for construction of such a controller, see [46]), so that $\tilde{H}(0) = I$. The right-hand side of (6.11) is greater than zero at all frequencies, and P is stable, so this controller can always be detuned such that (6.11) holds. Theorem 6.7 implies that the closed loop system will remain stable for all further detuning of the controller. QED.

The above theorem was previously stated in [76], but the above is the first rigorous proof to the author's knowledge. The value of μ in the above theorem is the inverse of the steady-state μ interaction measure. The above theorem is less conservative than (and immediately imply) all the computable conditions in Corollary 8 of [17].

Now we will develop a tool for determining whether there exists a decentralized integral controller for an uncertain system for which stability is maintained while detuning the gain of each loop separately.

Definition 6.2 *A plant P is Robust Decentralized Integral Controllable (RDIC) if there exists a decentralized controller K with integral action in all channels such that $\hat{K}(s) = EK(s)$ stabilizes $G(s)$ for all $E \in \overline{\mathcal{E}}_D$ and $\|\Delta\|_\infty \leq 1$ where*

$$\overline{\mathcal{E}}_D \equiv \{E = \mathrm{diag}(\epsilon_i) | \epsilon_i \in [0,1], i = 1, \ldots, n\}. \tag{6.21}$$

The following is a sufficient condition for RDIC.

Theorem 6.13 (Sufficient Condition for RDIC) *Stable G is RDIC if*

$$\mu_{\left[\begin{smallmatrix}\Delta & \\ & \Delta_{\tilde{H}}\end{smallmatrix}\right]}\left(\begin{bmatrix} G_{11}(0) & G_{12}(0)\tilde{P}^{-1}(0) \\ G_{21}(0) & I - P(0)\tilde{P}^{-1}(0) \end{bmatrix}\right) < 1, \tag{6.22}$$

and

$$\mu_{\Delta}(G_{11}(j\omega)) < 1, \qquad \forall\omega, \tag{6.23}$$

where $\Delta_{\tilde{H}}$ has the block structure of $\tilde{H}$.

Proof: Equation (6.22) implies that $\det(\tilde{P}(0)) \neq 0$, so there exists a stable decentralized Proportional-Integral controller K which stabilizes the block diagonal plant, so that $\tilde{H}(0) = I$. Equation (6.23) implies that $c^{su}_{\tilde{H}}(\omega)$ in (6.15) exists for all frequencies, and G is stable, so this controller can always be detuned such that (6.15) holds. This implies that the closed loop system will retain robust stability for all further detuning of the controller. QED.

The sufficient condition for DIC depends only on steady-state, whereas the sufficient condition for RDIC depends on all frequencies. Condition (6.22) for RDIC forces the controller to give the correct behavior at zero frequency whereas (6.23) is needed to guarantee that a controller exists which satisfies the robustness requirements at other frequencies.

Remark 6.1 *A necessary condition for RDIC is given by Theorem 6.3, though this condition ignores failure/fault tolerance.*

Remark 6.2 *The sufficient conditions in Theorems 6.12 and 6.13 are much stronger than DIC and RDIC, since these allow arbitrary (*static *and* dynamic*) detuning of H, and DIC and RDIC require stability when detuning only the controller single loop* gains.

Remark 6.3 *The decentralized integral controllability measures DIC and RDIC are closely related to CDUS and CRDUS defined in Chapter 5. A plant is DIC if there*

exists an integral controller which is CDUS, and similarly for RDIC and CRDUS. Decentralized integral controllability measures depend only of the plant (and possibly model uncertainty) and are useful for control structure selection, whereas decentralized unconditional controllability measures are specifications on the closed loop system, which depend on the plant (and possibly model uncertainty) and the controller.

Remark 6.4 *Nwokah et al. [81] claim that a necessary and sufficient condition for DIC is that* $P(0)$ *is* all gain positive stable. *The plant in Remark 5.2 shows that having* $P(0)$ *be all gain positive stable is* not necessary *for DIC.*

6.5 Examples

The following examples are used to illustrate the tools developed in this chapter. For further examples, see [67, 65, 64].

General Screening Tools for Multiplicative Input Uncertainty Though the purpose of the screening tools are for control structure selection of nontrivial systems, we will apply a few of the tools on a simple system with multiplicative input uncertainty (see Fig. 5.4) to provide an understanding of the nature of the screening tools. For simplicity of presentation we will chose the performance and uncertainty weights to be repeated scalar. The high purity distillation column studied in Chapter 5 was assumed to have this form of uncertainty.

The generalized plant G is determined by inspection of Fig. 5.4:

$$G_{11} = \begin{bmatrix} 0 & 0 \\ w_P P & w_P I \end{bmatrix}, G_{12} = \begin{bmatrix} -w_I I \\ -w_P P \end{bmatrix}, G_{21} = \begin{bmatrix} P & I \end{bmatrix}, G_{22} = -P. \qquad (6.24)$$

Application of Theorem 6.3 gives the following necessary condition for the existence

of a controller with integral action to provide robust performance:

$$\mu_\Delta\Big(G_{11}(0) + G_{12}(0)P^{-1}(0)G_{21}(0)\Big) = |w_I(0)| < 1 \iff \mu < 1. \tag{6.25}$$

The above necessary condition is independent of the structure of the uncertainty, and is required for the determinant of the steady-state gain not to change sign for all plants given by the uncertainty description.

Application of Theorem 6.4 gives the following necessary condition for the existence of a proper controller with to give robust performance:

$$\mu_\Delta(G_{11}(j\infty)) = |w_P(j\infty)| < 1 \iff \mu < 1. \tag{6.26}$$

The control engineer should always choose the performance weight so that this condition is satisfied, since it is *unreasonable* to expect high performance at infinite frequency where the system is essentially operating at open loop.

General Screening Tools for Inverse Multiplicative Output Uncertainty

In the above example, the necessary condition given by Theorem 6.4 depends on the performance weight, and not on the uncertainty. The following simple example illustrates that Theorem 6.4 *can* depend on the uncertainty.

We will assume inverse multiplicative output uncertainty (see Fig. 6.3), where the performance and uncertainty weights are chosen to be repeated scalar for simplicity of presentation. This form of uncertainty is commonly used to represent uncertainty associated with poles of the plant. The generalized plant G is determined by inspection of Fig. 6.3:

$$G_{11} = \begin{bmatrix} w_U I & 0 \\ w_P I & w_P I \end{bmatrix}, G_{12} = \begin{bmatrix} w_U P \\ w_P P \end{bmatrix}, G_{21} = \begin{bmatrix} -I & -I \end{bmatrix}, G_{22} = -P. \tag{6.27}$$

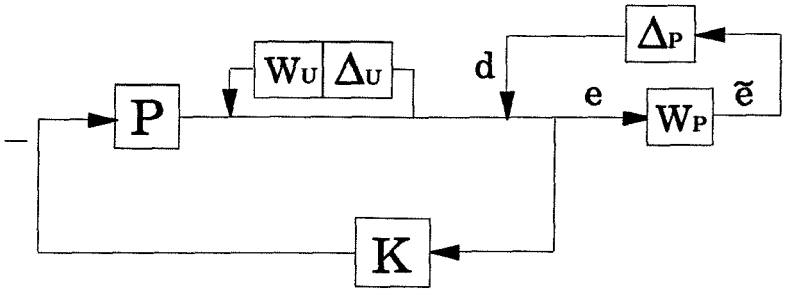


Figure 6.3: The plant with inverse multiplicative output uncertainty Δ_U of magnitude $w_U(s)$. Robust performance is satisfied if $\overline{\sigma}\left(w_P(I+\hat{P}K)^{-1}\right) \leq 1$ for all Δ_U with $\|\Delta_U\|_\infty \leq 1$.

Application of Theorem 6.3 gives the following necessary condition for the existence of a controller with integral action to provide robust performance:

$$\mu_\Delta\Big(G_{11}(0) + G_{12}(0)P^{-1}(0)G_{21}(0)\Big) = 0 < 1 \iff \mu < 1. \tag{6.28}$$

We see that in this case Theorem 6.3 does not give a useful necessary condition for robust performance. To understand why Theorem 6.3 does not give a useful condition, consider the set of plants given by the inverse multiplicative output uncertainty description:

$$\Big\{(I - w_U\Delta_U)^{-1}P \Big| \|\Delta_U\|_\infty \leq 1\Big\}. \tag{6.29}$$

Since $\det((I - w_U\Delta_U)^{-1}) = 1/\det(I - w_U\Delta_U) \neq 0$ for all finite Δ_U, the steady-state gain must be the same for all plants within the set. Thus inverse multiplicative output uncertainty cannot change the sign of the steady-state plant gain, and poses no limitations in terms of the stabilizability of the system under integral control. That Theorem 6.3 gives no useful necessary condition for robust performance agrees with this analysis.

Application of Theorem 6.4 gives the following necessary condition for the exis-

tence of a proper controller which provides robust performance:

$$\mu_\Delta(G_{11}(j\infty)) = \max\{|w_U(j\infty)|, |w_P(j\infty)|\} < 1 \iff \mu < 1. \tag{6.30}$$

The above necessary condition requires both that the uncertainty and performance weights are not too large at high frequency. We note that the condition on the performance and uncertainty weights are equivalent—this is because the uncertainty and performance enters the block diagram in the same manner in Fig. 6.3.

We again interpret Theorem 6.4 as providing a test for whether the uncertainty and performance weights are *reasonable* at infinite frequency. Only an unreasonable performance weight would have $|w_P(j\infty)| > 1$, since high performance cannot be expected at infinite frequency. The condition $|w_U(j\omega)| > 1$ at any frequency would be unreasonable since it would allow $I - w_U\Delta_U = 0$, leading to a poorly-defined set of plants in (6.29). We note that uncertainty weights in other locations (for example, multiplicative input uncertainty above) are commonly *expected* to be greater than 1 at high frequency.

Measurement Selection for a High-Purity Distillation Column Lee and Morari [67] studied secondary measurement selection for the high-purity distillation column described in detail in Appendix A of Morari and Zafiriou [76]. We include this example both to give a more involved illustration of the use of the tools, and to compare the screening tools presented in this chapter. The problem description will be brief since the distillation column is described elsewhere [76, 67].

The 41-tray distillation column is given in Fig. 5.3 except that in this case the manipulated variables are the reflux (L) and boilup (V), and the measurements are tray temperatures instead of compositions. Tray temperatures are typically measured in practice because composition measurements are often slow and unreliable. The disturbances are in the feed composition and flow rate, and measurement noise is due

to uncompensated pressure variation in the column. For simplicity of presentation we will restrict ourselves to two tray temperatures symmetric with respect to the feed tray. This is reasonable since the column is symmetric with respect to the feed tray. The uncertainty and performance weights are given in [67].

Lee and Morari chose an H_2 optimal estimator and controller (i.e., IMC) and applied Theorem 6.1 to test each measurement set for the existence of a diagonal filter designed by loopshaping which achieves robust performance. Fig. 6.4 is a plot of the left-hand side of (6.2) for the different measurement sets. The measurement set of T_7 and T_{35} is the only one that satisfies the condition (6.2). This result can be interpretted physically. The temperatures measured close to the reboiler and the condenser have poor signal/noise ratio because the gains from the feed disturbances to these measurements are small. On the other hand, the measurements far away from the reboiler and the condenser are sensitive to model uncertainty since the relationships between the end-point compositions and the measurements become less direct. The measurement set $\{T_7, T_{35}\}$ is apparently the best compromise between the signal/noise ratio and the sensitivity to model uncertainty. Lee and Morari go on to design a filter which achieves robust performance. This implies that the necessary condition given in Theorem 6.1 is *tight* for this distillation column.

Since the necessary test given by Theorem 6.1 is equal to the necessary test given by Theorem 6.3, we can immediately re-interpret the above results. With an H_2 optimal estimator and IMC controller *with integral action*, robust performance can potentially be achieved only for measurement set $\{T_7, T_{35}\}$. Since a controller which achieves robust performance can be designed for this measurement set, the screening tool given by Theorem 6.3 is tight for this problem.

We assumed above that the H_2 optimal estimator and controller would be designed, and then applied the design-independent screening tool given by Theorem 6.3. This leads to an important point—any *design-independent* screening tool can be ap-

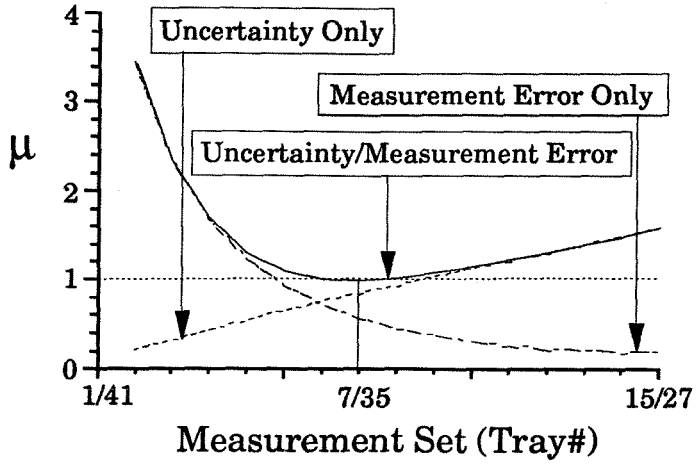


Figure 6.4: Screening tools for integral and loopshaping controllers: Tradeoff between model uncertainty and measurement noise.

plied as a *design-dependent* screening tool by pre-designing part of the control system before applying the tool. The H_2 optimal estimator was designed before application of Theorem 6.3 so that the test would be for the existence of a robust controller *with integral action on the composition estimates*, instead of integral action on the *measured variables.*

We also determined for which sets it were possible to achieve robust performance for the H_2 optimal estimator and *any* IMC controller. The H_2 optimal estimator and controller were designed, and Theorem 6.5 (or Theorem 6.6) applied at zero frequency to test whether an IMC filter F exists which achieves robust performance. The necessary condition implied by Theorem 6.5

$$\inf_{\Delta_F \text{ diagonal}} \mu_\Delta\Big(F_l(N^F, \Delta_F)\Big) \tag{6.31}$$

was solved at zero frequency via an off-the-shelf optimization package (the software was run repeated with different initial conditions—all runs converged to the same solution suggesting that the software converged to the global optimum). Solving this non-convex optimization is impractical in general and is only included here for

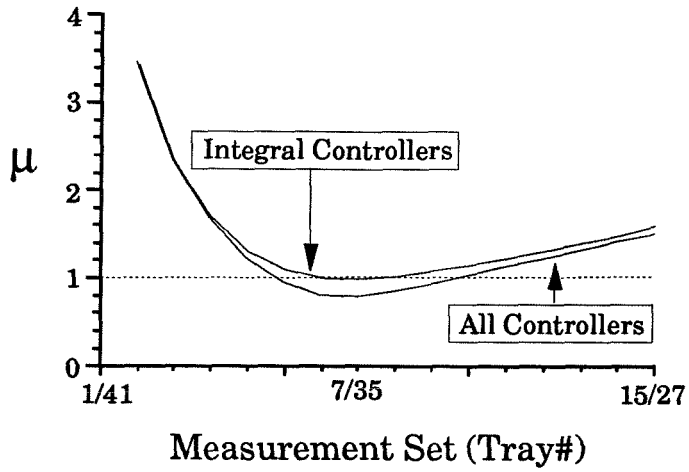


Figure 6.5: Comparison of screening tools for integral and general controllers.

purposes of comparison only.

The necessary conditions for integral and general controllers are given in Fig. 6.5. We see that, as expected, allowing the controller to not have integral action leads to a larger number of potential measurement sets. The performance weight specified by Lee and Morari [67] has a gain at zero frequency of 38, which is quite large and explains why the curves in Fig. 6.5 are close. This illustrates the general rule that the necessary condition of Theorem 6.3 approximates the more computationally complex necessary condition of Theorem 6.5 whenever the performance weight is sufficiently large. This can be proved rigorously for systems with at least one complex uncertainty block using the fact that μ for such systems is continuous [86] and that the conditions are equal when the performance weight includes an integrator in all channels.

Interaction Measure for Element-By-Element Uncertainty Many researchers have proposed to describe uncertainty as independent bounds on the individual transfer function elements. Though in general this is not a good representation of the actual sources of uncertainty, it is included here for illustration and to compare the results of this chapter with results from other researchers. In this example we determine the optimal interaction measures for systems with this uncertainty

description.

Skogestad and Morari [102] show that for testing robust stability with element-by-element uncertainty, M in Fig. 6.2 is given by

$$M = LP^{-1}HE \tag{6.32}$$

and Δ by

$$\Delta = \text{diag}\left\{\delta_{11}, \delta_{21}, \cdots, \delta_{nn}\right\}, \tag{6.33}$$

where n is the dimension of the plant, δ_{ij} is the perturbation on P_{ij}, and $E \in \mathcal{R}^{n \times n^2}$ and $L \in \mathcal{R}^{n^2 \times n^2}$ include the magnitudes of the uncertainty of each element of P and are given in [102].

By inspection, the expression for G in Fig. 6.2 is

$$G = \begin{pmatrix} 0 & L \\ E & -P \end{pmatrix}. \tag{6.34}$$

The robust interaction measure is given by $c^{su}_{\tilde{H}}$ in Theorem 6.9, where the structure of $\Delta_{\tilde{H}}$ is given by the desired structure of the controller. For example, if the controller is desired to be fully-decentralized, then $\Delta_{\tilde{H}}$ consists of independent 1×1 blocks. Thus robust stability is guaranteed by

$$|\tilde{h}_i(j\omega)| < c^{su}_{\tilde{h}_i I}(\omega) \quad \forall i, \forall \omega, \tag{6.35}$$

where $c^{su}_{\tilde{h}_i I}(\omega)$ is given by (6.16)

$$\mu_{\left[\Delta_{\delta_i \mathrm{I}}\right]}\left(\begin{bmatrix} 0 & E(j\omega)\tilde{P}^{-1}(j\omega) \\ c^{su}_{\tilde{h}_i I}(\omega)L(j\omega) & c^{su}_{\tilde{h}_i I}(\omega)(I - P(j\omega)\tilde{P}^{-1}(j\omega)) \end{bmatrix}\right) = 1. \tag{6.36}$$

The above equation can be used to design a controller which achieves robust stability. The tightest interaction measure is obtained by requiring the single loop designs to

have the same speed of response (replacing $\tilde{H} = \tilde{h}_i I$ above with repeated scalar $\tilde{H} = \tilde{h} I$).

The robust interaction theorem (Theorem 6.9) can also be used to quantify the limitations on the performance of inverse-based controllers on systems with element-by-element uncertainty. In this case $\tilde{P} = P$ (since the controller is centralized) and $H = hI$. Thus robust stability is guaranteed by

$$|h(j\omega)| < c_{hI}^{su}(\omega) \quad \forall \omega, \tag{6.37}$$

where $c_{hI}^{su}(\omega)$ is given by (6.16)

$$\mu_{\left[\begin{smallmatrix}\Delta \\ \delta I\end{smallmatrix}\right]}\left(\begin{bmatrix} 0 & E(j\omega)P^{-1}(j\omega) \\ c_{hI}^{su}(\omega)L(j\omega) & 0 \end{bmatrix}\right) = 1. \tag{6.38}$$

The value for c_{hI}^{su} can be solved explicitly using the definition of μ to give

$$c_{hI}^{su} = \mu_{\Delta}^{-1}(LP^{-1}E), \tag{6.39}$$

which was originally proven by Skogestad and Morari [102] to be the appropriate bound on hI to guarantee robust stability. Though the proof of Skogestad and Morari [102] is more direct then the above derivation, the robust interaction theorem also gives the appropriate bounds when using decentralized controllers (6.35-6.36). These bounds are more useful, since the use of an inverse-based controller is often impractical (for example, when the plant has high condition number).

The value for c_{hI}^{su} is determined via a μ calculation on a matrix of size n^2. The computations can be reduced by using the following inequality which has been found in practice to be tight [22, 58, 57]:

$$\mu_{\Delta}(LP^{-1}E) \leq \inf_{D_1, D_2} \overline{\sigma}(D_1 A D_2)\, \overline{\sigma}\left(D_2^{-1} P^{-1} D_1^{-1}\right), \tag{6.40}$$

where D_1 and D_2 are real positive diagonal matrices and A is given by the magnitude of the element-by-element uncertainties. The above optimization is shown in Chapter 9 to be equivalent to an upper bound μ calculation of a matrix of size $2n$. Thus a tight sufficient condition on $H = hI$ to satisfy robust stability is that

$$|h(j\omega)| \leq \left(\inf_{D_1(\omega), D_2(\omega)} \overline{\sigma}\left(D_1(\omega)A(j\omega)D_2(\omega)\right) \overline{\sigma}\left(D_2^{-1}(\omega)P^{-1}(j\omega)D_1^{-1}(\omega)\right) \right)^{-1}. \quad (6.41)$$

6.6 Branch-and-Bound

A centralized controller which includes all the actuators and sensors may be unnecessarily complex and expensive; whereas a control structure candidate with too few actuators and sensors or too restrictive of a decentralized structure may perform poorly. Screening tools provide a method to trade off control system complexity with closed loop performance. Owing to the combinatorial nature of the problem, however, the number of candidates is often very large. A branch-and-bound procedure can be used to ease the computational burden in choosing among control structure candidates.

The first step in the branch-and-bound procedure is to apply pairing-independent screening tools. One application of a pairing-independent tool can potentially remove from further consideration *all* control structure candidates associated with a given set of actuators and sensors (note that all pairing-dependent screening tools presented in this chapter can also be *applied* as pairing-independent tools by allowing the controller to be centralized in the tests). In this manner a large number of control structure candidates can be removed each time a pairing-independent tool is applied.

The second step is applied only to the candidates remaining from the first step. The screening tools are applied to decentralized controllers which consists of two full blocks—any partition/pairing which is a subset of a particular two-block decentralized

structure can potentially be removed from further consideration. At each additional step in the branch-and-bound procedure, the decentralized structure of the controller is further refined, potentially removing large sets of more refined structures from further consideration.

The complementary approach to reducing the computational burden would be to use sufficient conditions for a control structure candidate to achieve the performance specifications. The sufficient conditions would first be applied to fully-decentralized control structures. If a sufficient condition indicates that a fully-decentralized control structure satisfies the performance specifications, then the procedure can stop since an acceptable control structure has been found. If no fully-decentralized control structure satisfies the sufficient conditions, then the sufficient conditions would be applied to progressively less restrictive structures.

Unfortunately, the existing sufficient conditions either require the design of the controller which is computationally extensive, or guarantee only very simple performance specifications (for example, Theorems 6.12 and 6.13 do not address speed of response). Therefore it is currently suggested to only use necessary conditions in the branch-and-bound approach.

6.7 Interaction Between Design and Control

The approach of this chapter was to develop necessary conditions which must be satisfied by a control structure candidate for the performance specifications to be achievable. Control structure candidates which do not satisfy these necessary conditions can then be removed from further consideration.

This same approach can be used for providing recommendations on how to select *plant designs* which provide for the best achievable closed loop performance. Any plant design which does not satisfy the necessary conditions derived in this chapter

can be labeled as nonviable. When these necessary conditions are tight, they can also be used to rank both control structure candidates and plant designs in terms of their ability to achieve the performance specifications (see Lee et al. [65, 64, 66] for details). This removes the "yes-or-no" nature of the screening tools given in this chapter, and allows the exploration of how parameters associated with the plant design (for example, column width and height) affect the resulting closed loop performance.

6.8 Conclusions

Screening tools quickly reduce the potentially large number of control structure candidates to a manageable number for detailed analyses. New screening tools are presented for uncertain processes, as well as unified simple derivations of existing screening tools. The tools can also provide recommendations on how to modify the *plant design* to improve the closed loop performance.

The computation of the screening tools derived in this chapter, though manageable, is numerically more complex than conventional tools such as the RGA or the condition number. However, these other tools do not address the issue of plant/model mismatch in a general rigorous way like the above tools.

Part IV

Computational Issues

Chapter 7

Actuator and State Constraints

Summary

All real world control systems must deal with actuator and state constraints. Standard conic sector bounded nonlinearity stability theory provides methods for analyzing the stability and performance of systems under constraints, but it is well-known that these conditions can be very conservative. A method is developed to reduce conservatism in the analysis of constraints by representing them as nonlinear *real* parametric uncertainty.

7.1 Introduction

All real world control systems must deal with constraints. The control system must avoid unsafe operating regimes. In process control these constraints typically appear in the form of pressure or temperature limits. Further constraints are imposed by physical limitations—valves can only operate between fully open and fully closed, pumps and compressors have finite throughput capacity, surge tanks can only hold a certain volume.

One approach to controlling systems with constraints is to optimize the control

objective on-line subject to the constraints. This approach is referred to as *model predictive control* (MPC). A quadratic program must be solved at each sampling instance, and off-the-shelf software is available for performing these calculations [75]. Model predictive control does not completely solve the constrained control problem, however. MPC is computationally too complex for many industrial processes, which is part explains why MPC is typically implemented in a supervisory mode, i.e., *on top* of the regulatory control systems. Two additional disadvantages are that some operational requirements are impossible to express through a single objective function, and the stability and performance analysis with the resulting nonlinear controller is difficult.

The traditional method for dealing with constraints was to use simple static nonlinear elements (selectors and overrides) in the control system. Despite their considerable practical importance and extensive use, there is essentially no general theory to guide the design and analysis of these selector and override schemes. Furthermore, because they modify the control system configuration dynamically, they often cause severe performance degradation such as windup and "bumps" when switching modes. Though *ad hoc* design methods have been developed for avoiding windup, it has been shown that all of these techniques perform poorly (or may even lead to instability) in some situations.

A general method is needed for the design of robust constrained controllers which avoids the difficulties of model predictive control. This method should give robust controllers, be computationally simple on-line, and handle multiple performance objectives in a transparent manner. A general framework for the design of such controllers is provided by the *Anti-Windup Bumpless-Transfer* approach [16], and is illustrated by Fig. 7.1 for the case of actuator limitations. An additional linear compensator (R), called the anti-windup compensator, provides graceful performance degradation by modifying the error into the linear controller (K) when the constraints become

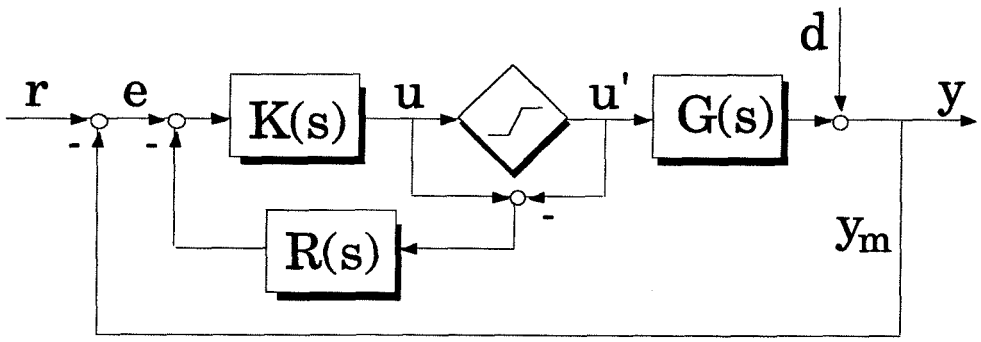


Figure 7.1: Anti-windup compensation.

active. When the constraints are inactive, the controller output equals the plant input and the anti-windup compensator does not affect the behavior of the closed loop system. This approach can be shown to be a generalization of the earlier *ad hoc* constraint-handling methods.

Note that the closed loop system involves linear systems with static memoryless nonlinearities. A necessary step in the further development of any anti-windup approach is to develop tools for analyzing stability and robustness for such systems. Campo [16] give sufficient conditions for analyzing stability and performance based on the standard conic sector bounded nonlinearity stability theory, but it is well-known that these conditions can be very conservative. This purpose of this chapter is to reduce the conservatism in these tools.

7.2 Conic Sector Bounded Nonlinearities

Since conic sector bounded nonlinearities are described in detail elsewhere (see, for example, [16]), here we will only illustrate the approach with an example. Fig. 7.2 shows a SISO saturation nonlinearity (this could due to either a state or actuator limitation—we will refer to the system component as being an actuator in what follows) covered by a conic sector. The actuator is assumed to behave linearly when the control output u is small, whereas the actuator output becomes limited when the

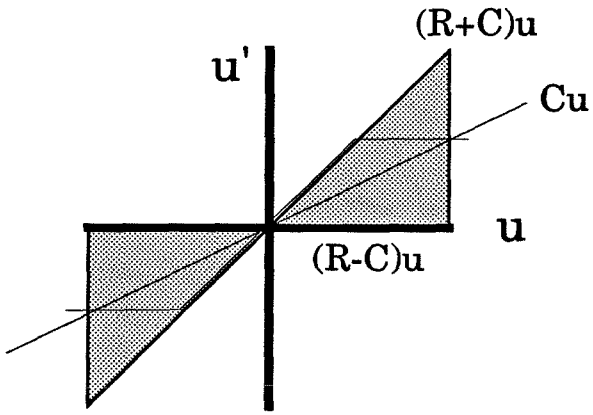


Figure 7.2: Conic sector bounded saturation nonlinearity.

control output becomes sufficiently large in magnitude. Two linear time invariant operators, denoted as the cone center C and the cone radius R, describe the conic sector and are shown in the figure. The purpose of covering the original nonlinearity by a conic sector is that the conic sector is described in terms of *linear operators*, and stability analysis for sets of nonlinearities bounded by linear operators is much more developed than stability with general nonlinearities. The standard approach [16] is then to analyze stability for all nonlinearities in the conic sector, giving a sufficient condition for stability for the original nonlinearity.

All nonlinearities in the system are covered by conic sectors, and the resulting conic sector descriptions are rearranged into the familiar leftmost block diagram in Fig. 7.3, where Δ has block structure as in the linear case [see (2.6)]. The difference in analyzing stability for this system, as opposed to the linear stability analysis used in the rest of this thesis, is that this "uncertainty" is a nonlinear time varying operator. The standard approach is to treat Δ as being complex, and the resulting stability condition is the optimally scaled small gain theorem [33].

Theorem 7.1 *The leftmost system in Fig. 7.3 is stable for all complex perturbations* $\overline{\sigma}(\Delta) \leq 1$ *if*

1. $M(s)$ *is stable, and*

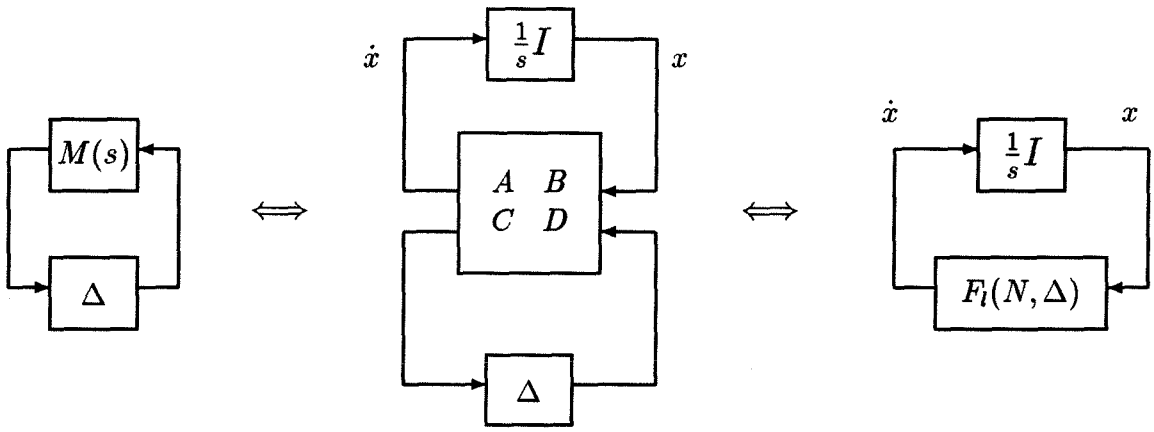


Figure 7.3: Equivalent block diagrams for continuous systems.

2. $\inf_{D \in \mathcal{D}} \|DM(s)D^{-1}\|_\infty \leq \beta < 1.$

Though the condition is necessary and sufficient for the set of unity norm bounded operators [98], it can be an extremely conservative stability test for the system with the original nonlinearities. One way to reduce this conservatism is to reduce the size of the set which covers the nonlinearities of interest. For example, actuator constraints are memoryless, i.e., the output of the actuator depends on its immediate input and not on past inputs. This means that the set of nonlinearities which cover the saturation nonlinearity can be taken to be *real*—this leads to Δ in Fig. 7.3 being real. This chapter uses this information to derive a less conservative condition for stability.

7.3 Stability with Memoryless Nonlinearities

Analyzing stability for discrete systems is simpler than for continuous systems, so we will first consider discrete systems and then show how to transform continuous systems into discrete. The following approach parallels that of Packard [87].

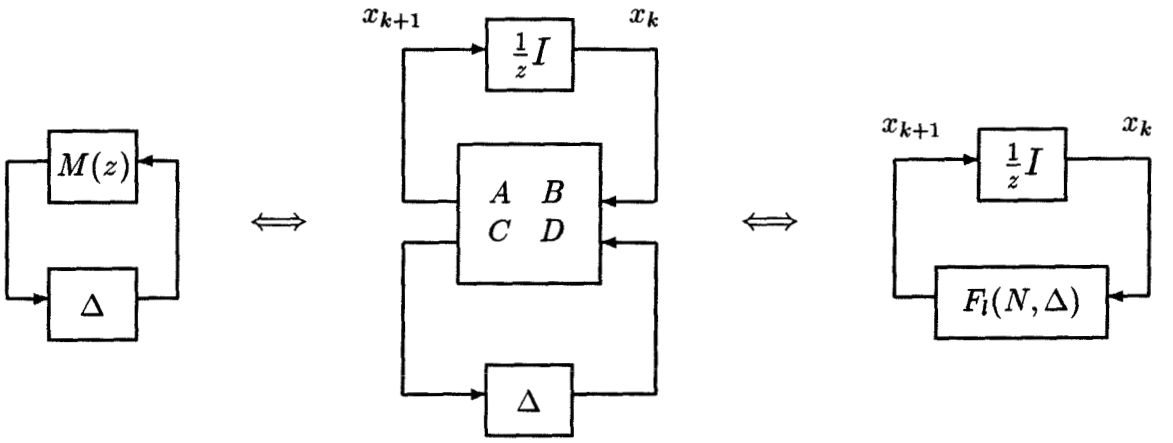


Figure 7.4: Equivalent block diagrams for discrete systems.

Discrete Time Systems Consider the block diagram in Fig. 7.4, where the discrete nominal transfer function $M(z) = C(zI - A)^{-1}B + D$. Define

$$\mathbf{\Delta}^k \equiv \{\Delta(k) \in \mathbf{\Delta}, \bar{\sigma}(\Delta(k)) \leq 1, \forall k\}. \tag{7.1}$$

We see that $\mathbf{\Delta}^k$ is allowed to vary over sampling instances, but must maintain the structure of $\mathbf{\Delta}$ [described in (2.6)]. The following theorem provides a sufficient condition for robust stability of a discrete system in terms of the upper bound of μ of a constant matrix.

Theorem 7.2 *The equivalent systems in Fig. 7.4 are stable for all* $\Delta \in \mathbf{\Delta}^k$ *if*

(i) $M(z) = F_l(\frac{1}{z}I_n, N)$ *is stable, and*

(ii) $\mu^{ub}_{\tilde{\Delta}}(N) < 1$, *where* $\tilde{\Delta} = \begin{bmatrix} \delta^c I_n & \\ & \Delta \end{bmatrix}, \delta^c \in \mathcal{C}, \Delta \in \mathbf{\Delta}$, *and* $\mu^{ub}_{\tilde{\Delta}}(N)$ *is the mixed upper bound [described in (2.14)] for* $\mu_{\tilde{\Delta}}(N)$.

Proof: Consider the rightmost of the equivalent block diagrams in Fig. 7.4. The system is described by the difference equation

$$x_{k+1} = F_l(N, \Delta)x_k. \tag{7.2}$$

Assume the nominal system $M(z)$ is stable. Then a sufficient condition for robust stability is that there exists an invertible $T \in \mathcal{C}^{n\times n}$ such that

$$\max_{\substack{\Delta \in \mathbf{\Delta} \\ \overline{\sigma}(\Delta) \leq 1}} \overline{\sigma}\left(TF_l(N,\Delta)T^{-1}\right) = \beta < 1, \tag{7.3}$$

since in this case the norm of x_k obeys

$$\|x_k\| \leq \kappa(T)\beta^k\|x_0\| \tag{7.4}$$

where $\kappa(T)$ denotes the condition number of T.

We have from Theorem 2.2 that

$$\max_{\substack{\Delta \in \mathbf{\Delta} \\ \overline{\sigma}(\Delta) \leq 1}} \overline{\sigma}\left(TF_l(N,\Delta)T^{-1}\right) < 1 \iff \mu_{\hat{\Delta}}\left[\begin{pmatrix} T & \\ & I \end{pmatrix} N \begin{pmatrix} T^{-1} & \\ & I \end{pmatrix}\right] < 1 \tag{7.5}$$

where $\hat{\Delta} = \begin{bmatrix} \Delta_1 & \\ & \Delta \end{bmatrix}$, $\Delta_1 \in \mathcal{C}^{n\times n}$, Δ_1 full block, and $\Delta \in \mathbf{\Delta}$. Combining (7.3) and (7.5) gives

$$\text{Robust Stability} \Longleftarrow \inf_{\substack{T \in \mathcal{C}^{n\times n} \\ T \text{ full}}} \mu_{\hat{\Delta}}\left[\begin{pmatrix} T & \\ & I \end{pmatrix} N \begin{pmatrix} T^{-1} & \\ & I \end{pmatrix}\right] < 1. \tag{7.6}$$

Calculating the minimization in the above equation is expected to be difficult, so we will replace μ with its upper bound [in (2.14)] to get

$$\text{Robust Stability} \Longleftarrow \max\left\{0, \sqrt{\inf_{\substack{T \in \mathcal{C}^{n\times n} \\ T \text{ full}}} \inf_{\substack{D \in \mathcal{D} \\ G \in \mathcal{G}}} \overline{\lambda}\left[\tilde{N}^*\tilde{N} + j\left(G\tilde{N} - \tilde{N}G\right)\right]}\right\} < 1, \tag{7.7}$$

where

$$\tilde{N} \equiv D \begin{pmatrix} T & \\ & I \end{pmatrix} N \begin{pmatrix} T^{-1} & \\ & I \end{pmatrix} D^{-1}, \tag{7.8}$$

$$D = \begin{pmatrix} d_1 I & \\ & D_2 \end{pmatrix}, \tag{7.9}$$

$$G = \begin{pmatrix} 0 & 0 \\ 0 & G_2 \end{pmatrix}, \tag{7.10}$$

$\overline{\lambda}(A)$ is the maximum eigenvalue of A, $d_1 \in \mathcal{R}$, $D_2 \in \mathcal{D}_2$, $G_2 \in \mathcal{G}_2$, and the sets $\mathcal{D}_2$ and $\mathcal{G}_2$ are specified by the structure of $\mathbf{\Delta}$. Absorbing d_1 into T and noticing that the structure of $\tilde{\Delta}$ is appropriate for the new "D" and "G" scalings gives the result. QED.

Continuous Time Systems Now we will consider stability of continuous time systems. We will need the following lemma from [87].

Lemma 7.1 *Let $n > 0$ be an integer, $A \in \mathcal{C}^{n\times n}$, and define a matrix B by*

$$B \equiv \begin{bmatrix} I_n & \sqrt{2}I_n \\ \sqrt{2}I_n & I_n \end{bmatrix}. \tag{7.11}$$

Let λ_i denote the eigenvalues of A, and $\rho(A)$ denote its spectral radius. Then

$$\text{Re}(\lambda_i) < 0, \forall i \iff I - A \text{ is invertible and } \rho\left(F_l(B, A)\right) < 1. \tag{7.12}$$

We will also need the definition of the star product. Assume that two matrices Q and M are partitioned such that

$$Q = \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix}, \qquad M = \begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix}, \tag{7.13}$$

and $Q_{22}M_{11}$ makes sense and is square. If $I - Q_{22}M_{11}$ is invertible, then the *star product* $Q * M$ is well defined and is given by

$$Q * M \equiv \begin{pmatrix} F_l(Q, M_{11}) & Q_{12}(I - M_{11}Q_{22})^{-1}M_{12} \\ M_{21}(I - Q_{22}M_{11})^{-1}Q_{21} & F_u(M, Q_{22}) \end{pmatrix}. \tag{7.14}$$

Now consider the block diagram in Fig. 7.3, where $M(s) = C(sI - A)^{-1}B + D$. Define

$$\mathbf{\Delta}^t \equiv \{\Delta(t) \in \mathbf{\Delta}, \overline{\sigma}(\Delta(t)) \leq 1, \forall t\}. \tag{7.15}$$

The perturbation $\mathbf{\Delta}^t$ may be any norm-bounded nonlinear time-varying operator, but must maintain the structure of $\mathbf{\Delta}$ [described in (2.6)]. Any subblock of $\mathbf{\Delta}^t$ which corresponds to a *real* subblock of $\mathbf{\Delta}$ is *memoryless*. The following theorem provides a sufficient condition for robust stability of a continuous system in terms of the upper bound of μ of a constant matrix.

Theorem 7.3 *The equivalent systems in Fig. 7.3 are stable for all* $\Delta \in \mathbf{\Delta}^t$ *if*

(i) $M(s) = F_l(\frac{1}{s}I_n, N)$ *is stable, and*

(ii) $\mu_{\tilde{\Delta}}^{ub}(B * N) < 1$, *where* $\tilde{\Delta} = \begin{bmatrix} \delta^c I_n & \\ & \Delta \end{bmatrix}, \delta^c \in \mathcal{C}, \Delta \in \mathbf{\Delta}$, *and* $\mu_{\tilde{\Delta}}^{ub}(B * N)$ *is the mixed upper bound [in (2.14)] for* $\mu_{\tilde{\Delta}}(B * N)$.

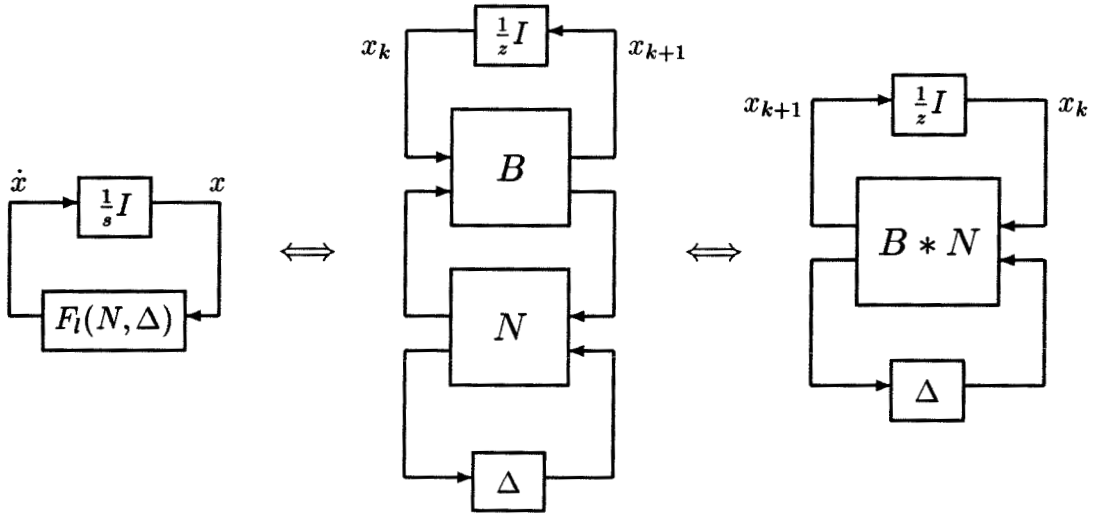


Figure 7.5: Transformation of the continuous stability test to the discrete stability test.

Proof: Consider the last of the equivalent block diagrams in Fig. 7.3. The system is described by the differential equation

$$\dot{x} = F_l(N, \Delta)x. \tag{7.16}$$

For robust stability, we want to test if the eigenvalues of $F_l(N, \Delta)$ are in the left half plane. The equivalence of the block diagrams in Fig. 7.5 follows from Lemma 7.1 with $A = F_l(N, \Delta)$. Thus we have converted the continuous robust stability problem to the discrete robust stability problem of Theorem 7.2. QED.

7.4 Robust Performance

The definition of robust performance is that (see Fig. 7.6)

$$\overline{\sigma}\left(F_l(M(j\omega), \Delta)\right) = \beta < 1 \qquad \forall\omega, \forall\Delta \in \mathbf{\Delta}^t. \tag{7.17}$$

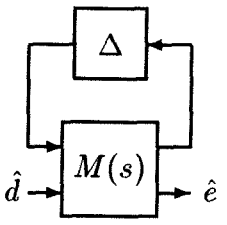


Figure 7.6: Continuous system with time varying uncertainty.

A similar development can be used to derive the following sufficient test for robust performance of continuous systems.

Theorem 7.4 *The system in Fig. 7.6 exhibits robust performance for all* $\Delta \in \mathbf{\Delta}^t$ *if*

(i) $M(s) = F_l(\frac{1}{s} I_n, N)$ *is stable, and*

(ii) $\mu^{ub}_{\tilde{\Delta}}(B*N) < 1$, *where* $\tilde{\Delta} = \begin{bmatrix} \delta^c I_n & & \\ & \Delta^c & \\ & & \Delta \end{bmatrix}$, $\delta^c \in \mathcal{C}$, Δ^c *a full complex block,* $\Delta \in \mathbf{\Delta}$, *and* $\mu^{ub}_{\tilde{\Delta}}(B * N)$ *is the mixed upper bound for* $\mu_{\tilde{\Delta}}(B * N)$.

Scaling of the uncertainty and the performance specifications can be incorporated into the above theorems to give greater flexibility (see [87] for details). We will now consider an example which shows a substantial reduction in conservatism when taking into account the memoryless nature of the common nonlinearities encountered in process industries.

7.5 Example

Consider the discrete 4×4 closed loop system (N) given by the following state space matrices:

$$A = \begin{pmatrix} -1.6662 & -3.2066 & 0.2522 & 4.6348 \\ -3.5907 & -6.5803 & 0.5290 & 9.3770 \\ -10.0332 & -20.5300 & 1.7744 & 27.3046 \\ -2.7552 & -4.9830 & 0.3936 & 7.2349 \end{pmatrix}, \tag{7.18}$$

$$B = \begin{pmatrix} -1.0801 & -0.3601 & -0.7408 & 2.0288 \\ -2.4983 & -0.4873 & -2.0047 & 3.8125 \\ -8.2313 & -2.7842 & -7.3722 & 11.1384 \\ -1.9733 & -0.4020 & -1.5071 & 3.1886 \end{pmatrix}, \tag{7.19}$$

$$C = \begin{pmatrix} -6.0014 & -11.6816 & 1.0568 & 17.1462 \\ -9.0589 & -18.0075 & 1.8934 & 24.7326 \\ -5.3812 & -10.7738 & 1.1081 & 14.9160 \\ -8.5404 & -16.9704 & 1.7568 & 23.4685 \end{pmatrix}, \tag{7.20}$$

$$D = \begin{pmatrix} -4.5765 & -1.1854 & -3.6748 & 7.0926 \\ -7.2120 & -2.2428 & -6.2854 & 10.1094 \\ -4.2841 & -1.3153 & -3.6850 & 6.1183 \\ -6.7764 & -2.0757 & -5.8558 & 9.6079 \end{pmatrix}. \tag{7.21}$$

The eigenvalues of A are $\{-0.1437, 0.3945, 0.3396, 0.1724\}$, which all have magnitude less than one so $M(z)$ is nominally stable. The nonlinearity $\mathbf{\Delta}^k$ consists of four memoryless repeated scalar 1×1 blocks. If we ignore that the nonlinearity is memoryless ($\mathbf{\Delta}$ complex), then the stability margin is

$$\mu_{\mathbf{\Delta}}^{ub}(B * N) = 3.57 > 1, \tag{7.22}$$

so stability of the closed loop system is not assured. If we take the memoryless nature of the nonlinearity into account ($\mathbf{\Delta}$ real), then the stability margin is

$$\mu_{\mathbf{\Delta}}^{ub}(B * N) = 0.98 < 1, \tag{7.23}$$

and so stability is guaranteed. The reduction in conservatism is 264%.

7.6 How Much Conservatism is Reduced?

Theorem 7.3 is equivalent to the standard conic sector stability test (Theorem 7.1 when the nonlinearity $\mathbf{\Delta}$ is complex [87]. Though Theorem 7.3 can substantially

reduce the conservatism over the standard conic sector stability test by taking into account the memoryless nature of the nonlinearity, the following lemma shows that there is no reduction in conservatism when all the subblocks of Δ are independent and 1×1.

Lemma 7.2 *Theorems 7.3 is no less conservative than the optimally scaled small gain theorem (Theorem 7.1) when all the subblocks of* $\Delta \in \mathbf{\Delta}^k$ *are* 1×1.

Proof: Follows from results in [118]. QED.

The example in Section 7.5 showed that the conservatism can be reduced when the nonlinearity was repeated scalar. This nonlinearity is appropriate under directionality compensation, which was discussed in Chapter 3 and is illustrated again in Fig. 7.7. When the control output cannot meet the constraints, the directionality compensator (which is placed immediately after the linear control and before the actuator constraints) scales back the control output *while keeping the same direction* until the control action becomes feasible. As discussed in Chapter 3, the directionality compensator was found to perform nearly as well as model predictive control for an industrial scale adhesive coater, but with much simpler computation. A detailed discussion of the importance of directionality compensation, especially when the controller is an inverse-based design, is provided by Campo [16].

The above approach was to reduce conservatism by accounting for the memoryless nature of the nonlinearity. To reduce the conservatism of the nonlinear stability conditions by a larger margin, it is needed to remove nonlinearities such as the one shown in Fig. 7.8, which can have arbitrary positive or negative instantaneous slope, and arbitrarily large magnitude as the input increases. The author is currently investigating the inclusion of bounds on the slope and magnitude of the nonlinearity in the problem formulation.

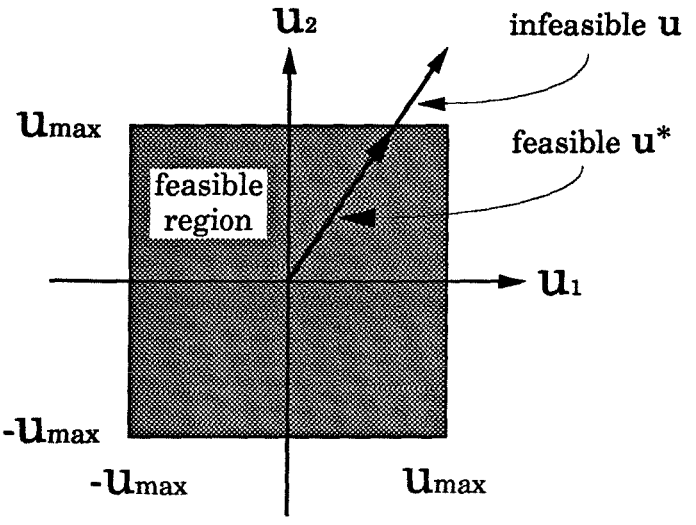


Figure 7.7: Directionality compensation.

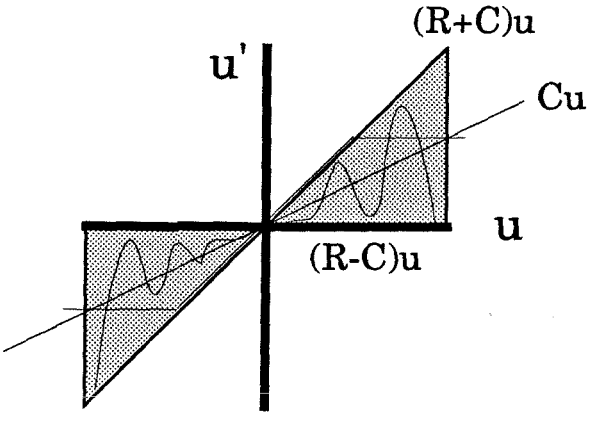


Figure 7.8: Conic sector bounded nonlinearities.

7.7 Nonlinear Stability and Performance

The stability and performance tests developed in this chapter can be used to test *local* stability and performance for general nonlinear systems. The nonlinear system is linearized, and the linear part rearranged to form the nominal system M in Fig. 7.6. A local operating region is defined in the phase plane, and the nonlinearity as a deviation from the linear system is covered by a conic sector in this region. The theorems developed in this chapter can be used to test stability and performance

for the system as long as the process stays in this operating region (for details and application to a packed bed reactor, see Doyle [34]).

In gain-scheduling, the nonlinear plant is treated as linear with time-varying parameters. The gain-scheduled controller is also linear, but dependent on the same time-varying parameters of the plant (which are assumed to be measured or estimated, see [88] for details). The tests in this chapter can be applied to analyze the *global* stability and performance for these systems, where the parameters are treated as time-varying uncertainty. Because both the controller and the plant depend on the parameters (i.e., the uncertainties are repeated), the tests can reduce conservatism by taking the real nature of the parameters into account.

7.8 Conclusions

Less conservative stability and performance tests are derived for memoryless nonlinearities. Though the tests cannot reduce conservatism for single-input single-output systems, the conservatism can be reduced substantially for multivariable systems with directionality compensation. The stability and performance tests developed in this chapter can also be used to test local stability and performance for general nonlinear systems, and global stability for gain-scheduled systems.

Chapter 8

Computational Complexity of μ Calculation

Summary

The matrix function, μ, is an integral part of both the robust loopshaping and the structured singular value frameworks. Numerous researchers over the last decade have worked on developing efficient methods for computing μ. In this chapter we consider the complexity of calculating μ with general mixed real/complex uncertainty in the framework of combinatorial complexity theory. In particular, it is proved that the μ recognition problem with either pure real or mixed real/complex uncertainty is NP-hard. This strongly suggests that it is futile to pursue exact methods for calculating μ of general systems with pure real or mixed uncertainty for other than small problems. This is strong motivation for the approach to the calculation of μ of Doyle and co-workers, which is to calculate tight polynomial-time upper and lower bounds instead.

8.1 Introduction

Robust stability and performance analysis with real parametric and dynamic uncertainties can be naturally formulated as a structured singular value (or μ) problem, where the block structured uncertainty description is allowed to contain both real and complex blocks. For a collection of papers describing the engineering motivation and the computational approaches, see [28] and the references contained within.

In this chapter we determine the computational complexity of μ calculation with either pure real or mixed real/complex uncertainty. To apply computational complexity theory, we formulate μ calculation as a *recognition problem* (a 'yes' or 'no' problem). We show that this recognition problem is NP-hard, i.e., at least as hard as the NP-complete problems.

The exact consequences of a problem being NP-complete is still a fundamental open question in the theory of computational complexity, and we refer the reader to Garey and Johnson [38] for an in-depth treatment of the subject. However, it is generally accepted that a problem being NP-complete means that it cannot be computed in polynomial time in the worst case. It is important to note that being NP-complete is a property of the problem itself, not of any particular algorithm. The fact that the mixed μ problem is NP-hard strongly suggests that, given *any* algorithm to compute μ, there will be problems for which the algorithm cannot find the answer in polynomial time.

The terminology of computational complexity theory is used extensively in this chapter. The definitions for NP-complete, NP-hard, recognition problems, and other terms agree with those in the well-known textbooks by Garey and Johnson [38] and Papadimitriou and Steiglitz [89].

The proofs are simple. First we show that indefinite quadratic programming can be cast as a μ problem of "roughly" the same size. Since the recognition problem for indefinite quadratic programming is NP-complete, the μ recognition problem must

be NP-hard.

This chapter has been accepted for publication in *IEEE Transactions on Automatic Control* [14].

Nomenclature Matrices are upper case; vectors and scalars are lower case. The set of real numbers is denoted by $\mathcal{R}$; the set of complex numbers by $\mathcal{C}$; and the set of rationals by $\mathcal{Q}$. The maximum singular value of matrix A is denoted by $\overline{\sigma}(A)$, and the $r \times r$ identity matrix by I_r. Define the set $\mathbf{\Delta}$ of block diagonal perturbations by

$$\begin{aligned}\mathbf{\Delta} \equiv \Big\{ & \operatorname{diag}\left\{\delta_1^r I_{r_1}, \cdots, \delta_k^r I_{r_k}, \delta_{k+1}^c I_{r_{k+1}}, \cdots, \delta_m^c I_{r_m}, \Delta_{m+1}, \cdots, \Delta_l\right\} \Big| \\ & \delta_i^r \in \mathcal{R}, \delta_i^c \in \mathcal{C}, \Delta_i \in \mathcal{C}^{r_i \times r_i}, \sum_{i=1}^{l} r_i = n \Big\}. \end{aligned} \tag{8.1}$$

Let $M \in \mathcal{C}^{n \times n}$. Then $\mu_{\Delta}(M)$ is defined as

$$\mu_{\Delta}(M) \equiv \begin{cases} 0 \text{ if there does not exist } \Delta \in \mathbf{\Delta} \text{ such that } \det(I - M\Delta) = 0, \\ \left[\min_{\Delta \in \mathbf{\Delta}} \left\{\overline{\sigma}(\Delta) \,|\det(I - M\Delta) = 0\right\}\right]^{-1} \quad \text{otherwise.} \end{cases} \tag{8.2}$$

Without loss of generality we have taken M and each subblock of Δ to be square.

8.2 Results

We first show that indefinite quadratic programming is a special case of a μ problem. Let $x, p, b_l, b_u \in \mathcal{R}^n$, $A \in \mathcal{R}^{n \times n}$, and $c \in \mathcal{R}$. Define the quadratic programming problem

$$\max_{b_l \le x \le b_u} \left| x^T A x + p^T x + c \right|, \tag{8.3}$$

where A can be indefinite. In the following theorem, we cast the above problem as a μ problem.

Theorem 8.1 (Quadratic Programming Polynomially Reduces to μ)

Define

$$M = \begin{bmatrix} 0 & 0 & kw \\ kA & 0 & kA\bar{x} \\ \bar{x}^T A + p^T & w^T & \bar{x}^T A\bar{x} + p^T\bar{x} + c \end{bmatrix}, \tag{8.4}$$

$$\mathbf{\Delta} = \{\mathrm{diag}[\delta_1^r, \cdots, \delta_n^r, \delta_1^r, \cdots, \delta_n^r, \delta^c] \,|\, \delta_i^r \in \mathcal{R}; \delta^c \in \mathcal{C}\}, \tag{8.5}$$

$$\tilde{\mathbf{\Delta}} = \left\{\mathrm{diag}[\delta_1^r, \cdots, \delta_n^r, \delta_1^r, \cdots, \delta_n^r, \delta_{n+1}^r] \,|\, \delta_i^r \in \mathcal{R}\right\}, \tag{8.6}$$

$$\bar{x} = \frac{1}{2}(b_u + b_l), \tag{8.7}$$

$$w = \frac{1}{2}(b_u - b_l). \tag{8.8}$$

Then $\mu_{\Delta}(M) = \mu_{\tilde{\Delta}}(M)$, *and*

$$\mu_{\Delta}(M) \geq k \Longleftrightarrow \max_{b_l \leq x \leq b_u} \left|x^T Ax + p^T x + c\right| \geq k. \tag{8.9}$$

This implies that the indefinite quadratic program (8.3) polynomially reduces *to both a real μ problem, and a mixed μ problem.*

Proof: The proof is trivial for $k = 0$, so assume $k > 0$. The idea is to treat the constraints as uncertainty and the objective function as the performance objective of a robust performance problem (see Chapter 2 for a description of the robust performance problem). The constraint set is

$$\{x \,|\, b_l \leq x \leq b_u\} = \{x \,|\, x = \bar{x} + \Delta^r w; \Delta^r = \mathrm{diag}[\delta_1^r, \cdots, \delta_n^r]; \delta_i^r \in [-1, 1]\}. \tag{8.10}$$

For convenience, define an artificial output $y \in \mathcal{R}$ and an artificial input $d \in \mathcal{R}$. Then the quadratic programming problem can be written as the block diagram in Fig. 8.1. Block diagram manipulations give us the block diagram in Fig. 8.2, where we have augmented the block diagram with a performance block δ^c. The optimization objective is the input-output relationship between d and y. Define $\Delta_U = \mathrm{diag}[\Delta^r, \Delta^r]$,

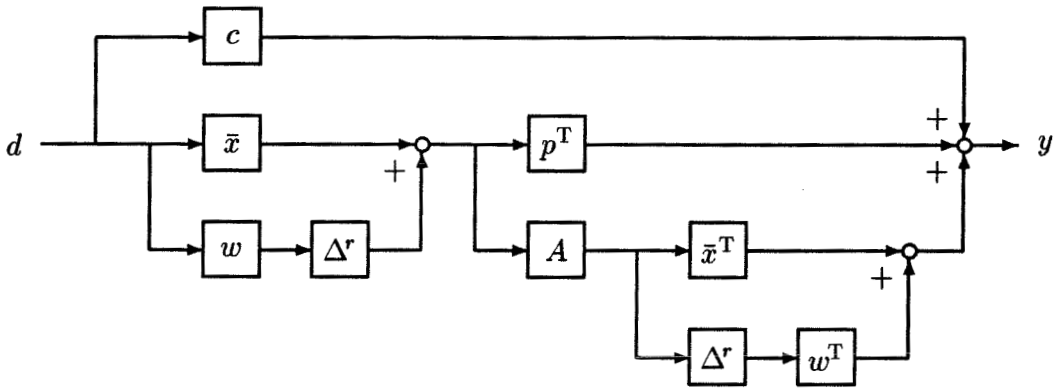


Figure 8.1: Equivalent block diagram for quadratic programming problem.

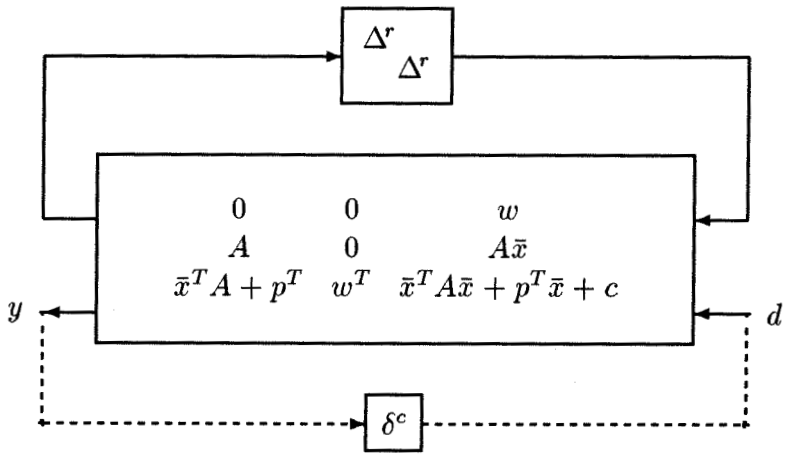


Figure 8.2: Quadratic programming as a robustness problem.

N by

$$N = \left[\begin{array}{c|c} N_{11} & N_{12} \\ \hline N_{21} & N_{22} \end{array}\right] = \left[\begin{array}{cc|c} 0 & 0 & w \\ A & 0 & A\bar{x} \\ \hline \bar{x}^T A + p^T & w^T & \bar{x}^T A\bar{x} + p^T\bar{x} + c \end{array}\right], \tag{8.11}$$

and the linear fractional transformation (LFT) $F_u(N, \Delta_U)$ by

$$F_u(N, \Delta_U) = N_{22} + N_{21}\Delta_U(I - N_{11}\Delta_U)^{-1}N_{12}. \tag{8.12}$$

Since $\det(I - N_{11}\Delta_U) = 1$, the inverse in (8.12) is well defined. We have

$$\max_{b_l \le x \le b_u} \left| x^T A x + p^T x + c \right| = \max_{\|\Delta_U\| \le 1} |F_u(N, \Delta_U)| = \max_{\|\Delta_U\| \le 1/k} \overline{\sigma}\left(F_u(M, \Delta_U)\right). \quad (8.13)$$

Since $\mu_{\Delta_U}(M_{11}) = 0 < k$, we can apply the robust performance theorem of Doyle [29] to give (8.9). Since $F_u(M, \Delta_U)$ has no dynamics and is 1×1, the complex perturbation δ^c can be replaced by a real perturbation.

It can easily be shown that the μ problem in (8.9) is described by less than four times the number of parameters of the quadratic program. QED.

Remark 8.1 *Theorem 8.1 can be generalized to handle general linear constraints instead of the simple ones in (8.3). Any unbounded linear constraints can be converted through a bilinear transform to bounded linear constraints. All bounded linear constraints can be treated as uncertainty—the details are left to the reader. Unfortunately, for general linear constraints the resulting μ problem is impractically large. Theorem 8.1 can also be modified to solve the optimization problem that does not have the absolute value in the objective. The idea is simple: the maximizing x does not depend on c, so choose $c > 0$ very large. Then solve the resulting "absolute value" μ problem. The maximizing x for this problem will solve the original problem. Minimizations can be handled just as easily as maximizations—choose $c < 0$ very large in magnitude and solve the resulting "absolute value" μ problem. We do not show the details of these generalizations here because the generality is not needed to prove the main results of this chapter.*

Remark 8.2 *Any nonlinear programming problem with an LFT of x and x^T as an objective and general linear constraints can be written as a block diagram like that of Fig. 8.1. The block diagram can always be rearranged to be in the form of Fig. 8.2, where $y = F_u(N, \Delta_U)d$, but with a different N and Δ_U. This block diagram has an equivalent μ problem. Therefore, any nonlinear programming problem with an*

LFT of x and x^T as an objective and general linear constraints can be cast as a μ problem. It is not clear how to efficiently write a given nonlinear (e.g., polynomial) objective as an LFT in terms of x and x^T except for the specific cases of linear and quadratic programming. But we have good methods for solving linear and quadratic (at least in the definite and semi-definite cases) programs—what might be interesting in terms of computation would be to solve optimizations with more difficult objective functions. The well-known lower and upper bounds (see Young et al. [119] for a summary) commonly used to approximate μ are bounds on the maximum of the "LFT" programming objective. The x that achieves the value of the lower bound can be calculated from the perturbation that achieves the lower bound from (8.7), (8.8), and (8.10). The error in the objective in using x from the lower bound algorithm instead of the optimal x is no greater than the difference between the upper and lower bounds.

To apply computational complexity theory, we must write the calculation of μ as a *recognition problem* (a 'yes or no' problem). Consider μ with $M \in \mathcal{Q}^{n\times n}$, $k \in \mathcal{Q}$, and mixed real/complex uncertainty blocks. Define the recognition problem $\Phi :=$ "Is $\mu \geq k$?" = "Does there exist a perturbation of magnitude k^{-1} that 'destabilizes' the system?"

The next lemma is essentially from Murty and Kabadi [77]. This paper is important because it is the first to use the techniques of discrete combinatorial complexity theory to study the computational difficulty of continuous optimization problems.

Consider $d_i \in \mathcal{Q}$ for $i = 0$ to n, and $k \in \mathcal{Q}$. Define the following nonconvex quadratic program

$$q := \max_{0\leq x_i\leq 1} \left(\sum_{i=1}^{n} d_i x_i - d_0\right)^2 + \sum_{i=1}^{n} x_i(1 - x_i). \tag{8.14}$$

Lemma 8.1 (NP-Completeness of Indefinite Quadratic Programming)
The recognition problem "Is $q \geq k$?" is NP-complete.

Proof: Murty and Kabadi [77] show that this problem is NP-hard. Vavasis [112] shows that the problem is in $\mathcal{NP}$. QED.

The following theorem states that the μ recognition problem is NP-hard.

Theorem 8.2 (NP-Hardness of μ Recognition) *Φ with general perturbation structure and general M is NP-hard.*

Proof: The indefinite program (8.14) can be written as (8.3) through multiplications and additions ($\sim \mathcal{O}(n^2)$ operations). This problem is NP-complete by Lemma 8.1, and the quadratic program (8.3) polynomially reduces to a μ problem by Theorem 8.1. Thus Φ is in general at least as difficult as indefinite quadratic programming, and Φ is NP-hard. QED.

Though the general μ recognition problem is NP-hard, special cases (i.e., with restrictions on the structure or field of M or Δ) may be simpler to compute. For example, when the M matrix is restricted to be rank one, the calculation of μ has sublinear growth in problem size, irrespective of the perturbation structure [18].

The case where μ has only real perturbations has received an especially large amount of attention in the μ calculation literature. The next result states that μ recognition is NP-hard for this case.

Theorem 8.3 (NP-Hardness of Real μ Recognition) *Φ is NP-hard when M and the perturbations are restricted to be real.*

Proof: Use the real μ problem of Thm 8.1 in the proof of Theorem 8.2. QED.

Models for real systems always have unmodeled dynamics associated with them. Unmodeled dynamics correspond to having at least one complex uncertainty *which enters nontrivially* in the μ problem. The next result states that μ recognition is NP-hard for this practically-motivated class of problems.

Theorem 8.4 (NP-Hardness of Mixed μ Recognition) *Let $\mathbf{\Delta}$ consist of both real and complex perturbations. Arrange the perturbations in $\mathbf{\Delta} = \text{diag}\{\Delta_1, \Delta_2\}$*

such that Δ_1 consists of pure real perturbations and Δ_2 consists of pure complex perturbations. Partition M compatibly, i.e.,

$$M = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}, \tag{8.15}$$

where $\mu_\Delta(M)$, $\mu_{\Delta_1}(M_{11})$, and $\mu_{\Delta_2}(M_{22})$ are well-defined. Consider the class of μ problems for which $\mu_{\Delta_1}(M_{11}) < \mu_\Delta(M)$. Φ is NP-hard for this class of problems.

Proof: Use the mixed μ problem of Thm 8.1 in the proof of Theorem 8.2. QED.

The *evaluation* problem "What is μ?" is at least as difficult to solve as the recognition problem "Is $\mu \geq k$?" since the solution of the recognition problem immediately follows from the solution to the evaluation problem.

8.3 Comparison with Previous Results

It can be shown from results of Rohn and Poljak and Demmel [93, 25] that the recognition problem for a special case of computing μ with only real perturbations is NP-complete. This implies that the μ recognition problems for both the pure real and general cases are *NP-hard* (Theorem 8.2 and 8.3).

Here we use a *control approach* to studying the computational complexity of μ. The proofs use only simple linear algebra—the approach in [93, 25] involves transformation to the "max-cut problem." Theorem 8.4, which shows that including complex perturbations (which appear to be better behaved numerically, see Young et al. [119]) in the μ problem does not remove the NP-hardness, follows naturally from the approach taken here. This result is important since practically-motivated μ problems are in this class.

Another immediate result (follows from [86]) of this chapter is that μ recognition remains NP-hard when the class of problems is restricted to those in which μ is a

continuous function of M.

8.4 Conclusion

The main results strongly suggest that it is futile to pursue exact methods for calculating μ of general systems with pure real or mixed uncertainty for other than small problems. In particular, one should not expect to find a polynomial time algorithm that calculates either real or mixed μ with general M exactly. These results do not mean, however, that practical algorithms are not possible. Practical algorithms for other NP-hard problems exist and typically involve approximation, heuristics, branch-and-bound, or local search [38, 89]. The results of Young et al. [119] strongly suggest that a combination of these techniques *which takes into account the structure of the μ calculation problem* can yield an algorithm which approximates μ in polynomial time for typical problems.

Chapter 9

Minimizing the Euclidean Condition Number

Summary

We consider the problem of determining the row and/or column scaling of a matrix A which minimizes the condition number of the scaled matrix. This problem has been studied by many authors. For the cases of the ∞-norm and the 1-norm the scaling problem was completely solved in the 1960s. It is the Euclidean norm case which has widespread application in robust control analyses. For example, it is used for integral controllability tests based on steady-state information, for the selection of sensors and actuators based on dynamic information, and for studying the sensitivity of stability to uncertainty in control systems.

Minimizing the scaled Euclidean condition number has been an open question—researchers proposed approaches to solving the problem numerically, but none of the proposed numerical approaches guaranteed convergence to the true minimum. In this chapter we provide a convex optimization procedure to determine the scalings which minimize the Euclidean condition number. This optimization can be solved in

polynomial-time with off-the-shelf software.

9.1 Introduction

Let $V_1 = C^n$ be the normed complex vector space with Hölder p-norm $\|\cdot\|_p$, $\|x\|_p = (\sum |x_j|^p)^{1/p}$. For an $n \times n$ matrix $A : V_1 \to V_1$, the following induced matrix norm is defined

$$\|A\|_{ip} = \max_{x \neq 0} \frac{\|Ax\|_p}{\|x\|_p}. \tag{9.1}$$

If the inverse A^{-1} exists then the condition number subordinate to the norm $\|\cdot\|_p$ is defined by

$$\kappa_p(A) = \|A\|_{ip}\|A^{-1}\|_{ip}. \tag{9.2}$$

Define $\mathcal{C}^{n\times n}$ to be the set of complex $n \times n$ matrices. Let $\mathcal{D}^{n\times n}$ be the set of all diagonal invertible matrices in $\mathcal{C}^{n\times n}$. If $A \in \mathcal{C}^{n\times n}$ is the matrix defining a system of linear equations, $Ax = b$, scaling the rows of this system is equivalent to pre-multiplying A by a diagonal matrix $D_1 \in \mathcal{D}^{n\times n}$. Scaling the unknowns is equivalent to post-multiplying A by a diagonal matrix $D_2 \in \mathcal{D}^{n\times n}$. The quality of numerical computations is generally better when the condition number of A is small. Since diagonal scalings of A are trivial modifications, researchers in the 1960s-1970s were led to investigating the following minimizations in order to get optimal scalings of a matrix

$$\begin{aligned}
(i) \qquad \kappa_p^l(A) &= \inf_{D_1 \in \mathcal{D}^{n\times n}} \kappa_p(D_1 A), \\
(ii) \qquad \kappa_p^r(A) &= \inf_{D_2 \in \mathcal{D}^{n\times n}} \kappa_p(A D_2), \\
(iii) \qquad \kappa_p^{lr}(A) &= \inf_{D_1, D_2 \in \mathcal{D}^{n\times n}} \kappa_p(D_1 A D_2).
\end{aligned} \tag{9.3}$$

Problem (i) was present for example in the error analysis of direct methods for the solution of linear equations [111, 4]. Problem (ii) is important for obtaining the best possible bounds for eigenvalue inclusion theorems [5], and is a natural measure of the linear independence of the column vectors which form A [4]. Problem (iii) was used for decreasing the error in calculation of the matrix inverse A^{-1} [50].

Later it was realized that the appropriate scalings depend on the error in the matrix, not the elements of the matrix itself [26, 108]. This implied, for example, that the scalings solving problem (iii) are not necessarily the best scalings of A to decrease the error in the calculation of A^{-1}. However, problems $(i - iii)$ still have widespread application in robust control analyses. For example, the minimized condition number (iii) is used for integral controllability tests based on steady-state information [44, 76] and for the selection of sensors and actuators using dynamic information [91, 78, 79]. The sensitivity of stability to uncertainty in control systems is given in terms of the minimized condition number in [102, 103].

Without loss of generality, for each of these problems we need only consider the infimum over the set of real positive diagonal invertible matrices $\mathcal{D}_+^{n\times n}$. This is because any matrix in $\mathcal{D}^{n\times n}$ can be decomposed into a matrix in $\mathcal{D}_+^{n\times n}$ and a unitary diagonal matrix. The unitary diagonal matrix does not affect the value of the condition number in (9.2) (see [4] for a simple proof). Conditions for the existence of scaling matrices which achieve the infimum are given by Businger [15].

The minimizations were solved for $p = 1$ and $p = \infty$ by Bauer [4] (the results are in Table 9.1). Many researchers consider the 2-norm as most important for applications [4, 50, 69]. Solving (i), (ii), and (iii) for the 2-norm has been an open question [101, 115]. In this chapter we solve the minimizations for the 2-norm by transforming the minimizations (9.3) so that they can be solved via convex programming.

Nonsquare A [110], block diagonal scalings [35, 100, 24, 27, 115], and cross-condition numbers (with B replacing A^{-1} in (9.2), see Chapter 6 for applications) have also received attention. For ease of notation, the results are derived for square matrices with fully diagonal scalings. The results (and proofs) hold for these other cases with the modifications given after the lemmas.

This chapter has been accepted for publication in *SIAM Journal on Control and Optimization* [11].

	$p = 1, \infty$	$p = 2$
$\inf_{D_1} \kappa_p(D_1 A)$	$\overline{\sigma}\left(\lvert A^{-1}\rvert \cdot \lvert A\rvert\right)$	?
$\inf_{D_2} \kappa_p(A D_2)$	$\overline{\sigma}\left(\lvert A\rvert \cdot \lvert A^{-1}\rvert\right)$	?
$\inf_{D_1, D_2} \kappa_p(D_1 A D_2)$	$\rho(\lvert A\rvert \cdot \lvert A^{-1}\rvert)$	?

Table 9.1: Minimized condition numbers. The matrix whose elements are the moduli of the corresponding elements of A is denoted by $|A|$. The spectral radius of A is denoted by $\rho(A)$. The maximum singular value $\overline{\sigma}(A)$ refers to $\|A\|_{i2}$.

9.2 Results

The induced matrix norm for the vector 2-norm is commonly referred to as the maximum singular value, $\overline{\sigma}(A) = \|A\|_{i2}$. To simplify notation, drop the subscript on κ_2, i.e., $\kappa_2 = \kappa$. Let $\mathcal{R}_+$ be the set of real positive scalars. Let I be the $n \times n$ identity matrix.

Lemma 9.1 *The following equality holds:*

$$\kappa(A) = \inf_{d_1, d_2 \in \mathcal{R}_+} \overline{\sigma}^2 \left(\begin{bmatrix} d_1 I & 0 \\ 0 & d_2 I \end{bmatrix} \begin{bmatrix} 0 & A^{-1} \\ A & 0 \end{bmatrix} \begin{bmatrix} (d_1)^{-1} I & 0 \\ 0 & (d_2)^{-1} I \end{bmatrix} \right). \tag{9.4}$$

Proof (similar to [80]):

$$\inf_{d_1, d_2 \in \mathcal{R}_+} \overline{\sigma}^2 \left(\begin{bmatrix} d_1 I & 0 \\ 0 & d_2 I \end{bmatrix} \begin{bmatrix} 0 & A^{-1} \\ A & 0 \end{bmatrix} \begin{bmatrix} (d_1)^{-1} I & 0 \\ 0 & (d_2)^{-1} I \end{bmatrix} \right) \tag{9.5}$$

$$= \inf_{d_1, d_2 \in \mathcal{R}_+} \overline{\sigma}^2 \left(\begin{bmatrix} 0 & \frac{d_1}{d_2} A^{-1} \\ \frac{d_2}{d_1} A & 0 \end{bmatrix} \right) \tag{9.6}$$

$$= \inf_{d_1, d_2 \in \mathcal{R}_+} \max \left\{ \overline{\sigma}^2 \left(\frac{d_1}{d_2} A^{-1} \right), \overline{\sigma}^2 \left(\frac{d_2}{d_1} A \right) \right\} \tag{9.7}$$

$$= \inf_{d_1,d_2\in\mathcal{R}_+} \max\left\{\frac{d_1^2}{d_2^2}\frac{\overline{\sigma}(A^{-1})}{\overline{\sigma}(A)}, \frac{d_2^2}{d_1^2}\frac{\overline{\sigma}(A)}{\overline{\sigma}(A^{-1})}\right\}\cdot\overline{\sigma}(A)\overline{\sigma}\left(A^{-1}\right) \tag{9.8}$$

$$= \inf_{x\in\mathcal{R}_+} \max\left\{x, x^{-1}\right\}\cdot\overline{\sigma}(A)\overline{\sigma}\left(A^{-1}\right) \tag{9.9}$$

$$= \overline{\sigma}(A)\overline{\sigma}\left(A^{-1}\right) = \kappa(A). \tag{9.10}$$

QED.

The following lemma gives similar expressions as in (9.4) for $\kappa^l(A)$, $\kappa^r(A)$, and $\kappa^{lr}(A)$.

Lemma 9.2 *The following equalities hold:*

$$\kappa^l(A) = \inf_{D_1\in\mathcal{D}_+^{n\times n}} \overline{\sigma}^2\left(\begin{bmatrix} I & 0 \\ 0 & D_1 \end{bmatrix}\begin{bmatrix} 0 & A^{-1} \\ A & 0 \end{bmatrix}\begin{bmatrix} I & 0 \\ 0 & D_1^{-1} \end{bmatrix}\right). \tag{9.11}$$

$$\kappa^r(A) = \inf_{D_2\in\mathcal{D}_+^{n\times n}} \overline{\sigma}^2\left(\begin{bmatrix} D_2^{-1} & 0 \\ 0 & I \end{bmatrix}\begin{bmatrix} 0 & A^{-1} \\ A & 0 \end{bmatrix}\begin{bmatrix} D_2 & 0 \\ 0 & I \end{bmatrix}\right). \tag{9.12}$$

$$\kappa^{lr}(A) = \inf_{D\in\mathcal{D}_+^{2n\times 2n}} \overline{\sigma}^2\left(D\begin{bmatrix} 0 & A^{-1} \\ A & 0 \end{bmatrix}D^{-1}\right). \tag{9.13}$$

Proof: Substituting D_1AD_2 for A in Lemma 9.1 and rearranging gives

$$\kappa(D_1AD_2) = \inf_{d_1,d_2\in\mathcal{R}_+} \overline{\sigma}^2\left(\begin{bmatrix} d_1D_2^{-1} & 0 \\ 0 & d_2D_1 \end{bmatrix}\begin{bmatrix} 0 & A^{-1} \\ A & 0 \end{bmatrix}\begin{bmatrix} (d_1D_2^{-1})^{-1} & 0 \\ 0 & (d_2D_1)^{-1} \end{bmatrix}\right), \tag{9.14}$$

where d_1 and d_2 are real positive scalars.

Take the infimum over D_1 and D_2 on both sides to give

$$\kappa^{lr}(A) = \inf_{D_1,D_2\in\mathcal{D}_+^{n\times n}} \inf_{d_1,d_2\in\mathcal{R}_+} \overline{\sigma}^2\left(\begin{bmatrix} d_1D_2^{-1} & 0 \\ 0 & d_2D_1 \end{bmatrix}\begin{bmatrix} 0 & A^{-1} \\ A & 0 \end{bmatrix}\begin{bmatrix} (d_1D_2^{-1})^{-1} & 0 \\ 0 & (d_2D_1)^{-1} \end{bmatrix}\right) \tag{9.15}$$

$$= \inf_{D_1,D_2\in\mathcal{D}_+^{n\times n}} \overline{\sigma}^2\left(\begin{bmatrix} D_2^{-1} & 0 \\ 0 & D_1 \end{bmatrix}\begin{bmatrix} 0 & A^{-1} \\ A & 0 \end{bmatrix}\begin{bmatrix} D_2 & 0 \\ 0 & D_1^{-1} \end{bmatrix}\right). \tag{9.16}$$

Letting $D = \text{diag}\left\{D_2^{-1}, D_1\right\}$ gives (9.13). Expressions (9.11) and (9.12) are proved similarly. QED.

Let I_r be the $r \times r$ identity matrix. Let $\mathcal{D}^{2n\times 2n} = \text{diag}\left\{[d_1 I_{r_1}, \cdots, d_m I_{r_m}] : d_j \in \mathcal{R}, r_1 + \cdots + r_m = 2n\right\}$, and $M \in \mathcal{C}^{2n\times 2n}$. Consider the following lemma:

Lemma 9.3 *The following optimization is convex:*

$$\inf_{D\in\mathcal{D}^{2n\times 2n}} \overline{\sigma}^2\left(e^D M e^{-D}\right). \tag{9.17}$$

Proof: See [97].

Since $\left\{e^D : D \in \mathcal{D}\right\} = \left\{D : D \in \mathcal{D}_+\right\}$, the optimizations in Lemmas 9.1 and 9.2 are equivalent to the optimization in Lemma 9.3. This means that the condition number κ and minimized condition numbers κ^l, κ^r, κ^{lr} can all be calculated through convex programming. Since the optimization (9.17) is convex, it can have only one minimum.

The optimization (9.17) has been studied extensively [84, 109, 85, 97], and off-the-shelf software is available for solving these polynomial-time problems (for example, see the program *mu* in [3]). The calculation of the minimized condition numbers is slow; however, since the minimization (9.17) requires repeated maximum singular value calculations.

The parallelism between expressions (9.4), (9.11), (9.12), and (9.13) for κ, κ^l, κ^r, and κ^{lr} is interesting. The same optimization can be used for the condition number calculations—the optimizations are just over different "scaling matrices." This is nice theoretically, since κ^l, κ^r, and κ^{lr} are just the *scaled* condition numbers.

Remark 9.1 *Conditions for the existence of scaling matrices which achieve the infimum are given by Businger [15]. When the infinum is achieved, any algorithm*

which solves (9.17) provides the minimizing scaling matrices for the condition number. When the infinum is not achieved, the algorithm provides scaling matrices such that the infinum is approached with arbitrary closeness.

Remark 9.2 *To generalize to nonsquare A, replace every occurrence of A^{-1} with the respective right or left inverse. More specifically, if $A \in \mathcal{C}^{m\times n}$ and has full row rank with $m < n$, then replace A^{-1} with $A^T(AA^T)^{-1}$ in all proofs and lemmas. For $m > n$ with A having full column rank, replace A^{-1} with $(A^TA)^{-1}A^T$.*

Remark 9.3 *The Euclidean cross-condition number is defined by*

$$\hat{\kappa}(A,B) \equiv \overline{\sigma}(A)\,\overline{\sigma}(B). \tag{9.18}$$

Minimized cross-condition numbers can be defined similarly as in (9.3), for example

$$\hat{\kappa}^{lr}(A,B) \equiv \inf_{D_1,D_2\in\mathcal{D}^{n\times n}} \hat{\kappa}(D_1AD_2, D_2^{-1}BD_1^{-1}). \tag{9.19}$$

Lemmas 9.1 and 9.2 follow with B replacing A^{-1}. This problem is important for testing stability of systems with element-by-element uncertainty [22, 58, 57].

Remark 9.4 *For block-diagonal scaling matrices, without loss of generality we can take each block to be positive definite Hermitian. This is because any nonsingular complex matrix can be decomposed into a positive definite Hermitian matrix and a unitary matrix [6], and the unitary matrix does not affect the value of the Euclidean condition number. The proofs of Lemmas 9.1 and 9.2 follow exactly as for the fully diagonal case. Lemma 9.3 does not hold for block-diagonal scalings. For block-diagonal scalings it is better to convert the singular value minimizations in Lemma 9.2 into generalized eigenvalue minimizations,*

$$\inf_{D\in\mathcal{D}_+} \overline{\sigma}(DMD^{-1}) = \inf_{D^2\in\mathcal{D}_+} \left\{\beta \,\middle|\, M^*D^2M - \beta D^2 < 0\right\}. \tag{9.20}$$

The condition $M^*D^2M - \beta D^2 < 0$ is convex in D^2, so any local minimum is global and off-the-shelf software is available [3]. Many researchers are working to develop improved computational approaches for these polynomial-time problems (the above problems are referred to as linear matrix inequalities, see [8] for details).

9.3 Conclusions

We have completed Table 9.1 in the sense that all values in the table can now be calculated with arbitrary precision.

All entries in the table including the now-filled entries require the inverse of A to calculate the minimizing scalings and the minimized condition numbers. There are algorithms for numerically determining the minimized condition numbers without pre-determining the matrix inverse [99, 115], but these methods are not guaranteed to converge to the true minima.

Part V

Conclusions and Suggestions for Future Work

Chapter 10

Conclusions and Suggestions for Future Work

10.1 Summary of Contributions

The strengths of the structured singular value framework are that it addresses uncertainty and performance specifications in a general, unified manner. A framework, referred to as *robust loopshaping*, was developed to address additional practical process control considerations. This framework is used to low-order controllers (e.g., PID) which are easier for operators to understand and maintain, and decentralized controllers which are reliable to equipment faults or failures. Screening tools for the general control structure selection problem (the selection, pairing, and partitioning of actuators and sensors for control purposes) follow immediately. These screening tools also provide recommendations on how to modify the *plant design* to improve the closed loop control.

Computational issues associated with both the structured singular value and robust loopshaping frameworks are then addressed. We develop a method to reduce conservatism in the analysis of constraints by covering them with a nonlinear *real*

parametric uncertainty description. Then we prove that *any* algorithm for exactly calculating μ has exponential growth in the size of the problem, which motivates the approach of calculating tight polynomial-time upper and lower bounds instead. We finish by developing a polynomial-time method for calculating the minimized scaled condition number, another matrix function which shows up in robustness analyses.

10.2 Suggestions for Future Work

The robust loopshaping framework was developed to address practical process control considerations in a general, unified manner. Work in the following areas are needed for *any* general approach to advanced process control to have widespread acceptance and application in industrial practice.

Modeling for Control Purposes The most difficult, time-consuming, and expensive step in the design of control systems is the modeling of the process. Modeling the process as an uncertain system for robust control purposes requires the development of both a nominal model and an uncertainty description. Too tight of an uncertainty description leads to aggressive control actions with large overshoots or possibly instability, whereas too broad of an uncertainty description leads to a controller which performs sluggishly.

Plants have many inputs and outputs, are strongly interacting, and have high-order dynamics. Techniques must be developed for building low order models which capture the essential behavior of these processes. Because modeling is expensive, these techniques should require a minimal amount of modeling effort. A clear understanding of the tradeoff between model accuracy and control quality is essential for determining whether increased modeling effort is justified, or whether the process must be redesigned to satisfy the specified performance.

Many researchers have developed methods to identify uncertainty bounds for

single-input single-output systems [41, 45, 48, 49, 56, 59, 70, 90]. These methods should be applied to real processes to assess their usefulness. If these methods are useful, then the methods need to be extended to multivariable systems. Expected to be of great importance in any multivariable approach is the plant directionality, since plants with high directionality are known to be inherently difficult to identify. A general method should be explored in which information on plant directionality acquired from physical considerations is used to improve the identification procedure.

Reliable Control A different approach to fault tolerant controller design than that described in this thesis is to design multiple controllers, in which a logic system switches between different controllers depending on the current operating conditions. This approach is more complicated than designing a single controller, but has the possibility of improved performance. Initial work suggests that methods developed for constrained control may be useful for the design of the multiple controllers. This design approach needs to be studied further. A robust method to detect faults is needed, both so that the logic scheme will know when to switch controllers to correct for the faults, and to alert the operator to the existence and location of the faulty component so it can be repaired.

Constrained Control The approach in Chapter 7 was to reduce conservatism by accounting for the memorylessness of the nonlinearity. To reduce the conservatism of the nonlinear stability conditions by a larger margin, it is needed to remove nonlinearities which can have arbitrary positive or negative instantaneous slope, and arbitrarily large magnitude as the input increases. Future research should investigate the inclusion of bounds on the slope and magnitude of the nonlinearity in the problem formulation. Once the analysis tools are adequately tight, constraints can be included in the robust loopshaping framework. This inclusion is important because it is expected that constraints can have a significant impact on control structure selection

and plant design decisions.

It is important to develop methods for designing the *Anti-Windup Bumpless-Transfer* controllers described in Chapter 7. The effect of model uncertainty and of intrinsic characteristics of the plant (such as right half plane zeros, time delays, or ill-conditioning) on the achievable performance should be quantified for constrained systems. The performance of AWBT controllers should be compared with the performance attainable by model predictive control.

Control of Coating and Paper Manufacturing Processes Most research results in cross-directional control assume negligible end effects and that the number of sensors and actuators are equal [54, 53, 61]. Effective control techniques need to be developed for processes in which these assumptions do not hold.

Fixed sensors were assumed in Chapter 3. Because of the high cost of sensors, a common practice in industry is to use a single sensor which moves back and forth transverse to the movement of the substrate. Filtering techniques are then used to estimate the correct profiles. A method is needed to design this filter so that the overall control system is robust and failure tolerant.

Process Applications The modeling, design, and control of coating processes were studied in Chapter 3. Because this class of processes was particularly simple, many important process control considerations (such as the effect of design on control, and how to design low-order controllers and handle actuator constraints and real parametric uncertainty) could easily be addressed which would have been much more difficult to address in general. Robust loopshaping was developed as an approach for addressing these (and other) control considerations for general processes. Though initial results on distillation column case studies (in this thesis and [65, 64, 66]) are promising, the ultimate effectiveness of any control approach must be judged on the basis of its application to real systems. Applications to real systems will suggest how

to modify the framework to improve its applicability.

Bibliography

[1] K. J. Åström and B. Wittenmark. *Computer Controlled Systems: Theory and Design.* Prentice-Hall, Inc., Englewood Cliffs, N.J., 1984.

[2] G. J. Balas. *Robust Control of Flexible Structures: Theory and Experiments.* PhD thesis, California Institute of Technology, Pasadena, 1990.

[3] G. J. Balas, A. K. Packard, J. C. Doyle, K. Glover, and R. S. R. Smith. Development of advanced control design software for researchers and engineers. In *Proceedings of the 1991 American Control Conference*, pages 996–1001, 1991.

[4] F. L. Bauer. Optimally scaled matrices. *Numer. Math.*, 5:73–87, 1963.

[5] F. L. Bauer and A. S. Householder. Absolute norms and characteristic roots. *Numer. Math.*, 3:241–246, 1961.

[6] R. Bellman. *Introduction to Matrix Analysis.* McGraw-Hill, New York, 1960.

[7] H. W. Bode. *Network Analysis and Feedback Amplifier Design.* D. Van Nostrand, Princeton, N.J., 1945.

[8] S. Boyd and L. E. Ghaoui. Method of centers for minimizing generalized eigenvalues. *Linear Algebra and Its Applications*, 1993. in press.

[9] R. D. Braatz and M. Morari. Robust control for a noncolocated spring-mass system. *AIAA Journal of Guidance, Control and Dynamics*, pages 1103–1110, 1992.

[10] R. D. Braatz and M. Morari. Robust control for a noncolocated spring-mass system. In *Proceedings of the 1992 American Control Conference*, pages 2061–2062, 1992.

[11] R. D. Braatz and M. Morari. Minimizing the Euclidean condition number. *SIAM Journal on Control and Optimization*, 1993. in press.

[12] R. D. Braatz, M. L. Tyler, and M. Morari. Avery final report: Identification and cross-directional control of coating processes. CDS Technical Memo CIT-CDS 92-004, California Institute of Technology, Pasadena, CA 91125, 1992.

[13] R. D. Braatz, M. L. Tyler, M. Morari, F. R. Pranckh, and L. Sartor. Identification and cross-directional control of coating processes. *AIChE Journal*, pages 1329–1339, 1992.

[14] R. D. Braatz, P. M. Young, J. C. Doyle, and M. Morari. Computational complexity of μ calculation. *IEEE Trans. on Auto. Control*, 1993. in press.

[15] P. A. Businger. Matrices which can be optimally scaled. *Numer. Math.*, 12:346–348, 1968.

[16] P. J. Campo. *Studies in Robust Control of Systems Subject to Constraints.* PhD thesis, California Institute of Technology, Pasadena, 1990.

[17] P. J. Campo and M. Morari. Achievable closed loop properties of systems under decentralized control: Conditions involving the steady state gain. *IEEE Trans. on Auto. Control*, 1993. in press.

[18] J. Chen, M. K. H. Fan, and C. N. Nett. The structured singular value and stability of uncertain polynomials: A missing link. *Control of Systems With Inexact Dynamic Models, ASME*, pages 15–23, 1991.

[19] E. D. Cohen. Coatings: Going below the surface. *Chem. Eng. Prog.*, 86(9):19–23, 1990.

[20] E. D. Cohen, E. J. Lightfoot, and E. B. Gutoff. A primer on forming coatings. *Chem. Eng. Prog.*, 86(9):30–36, 1990.

[21] *International Conferences on Chemical Process Control (CPC)* and the *International Symposia on Process Systems Engineering (PSE).*

[22] R. W. Daniel, B. Kouvaritakis, and H. Latchman. Principal direction alignment: A geometric framework for the complete solution to the μ-problem. *IEE Proceedings Part D*, 133:45–56, 1986.

[23] P. J. Davis. *Circulant Matrices.* Wiley, New York, 1979.

[24] J. W. Demmel. The condition number of equivalence transformations that block diagonalize matrix pencils. *SIAM. J. Numer. Anal.*, 20:599–610, 1983.

[25] J. W. Demmel. The component-wise distance to the nearest singular matrix. *SIAM. J. Matrix Anal. Appl.*, 13:10–19, 1992.

[26] J. J. Dongarra, C. B. Moler, J. R. Bunch, and G. W. Stewart. *LINPACK User's Guide.* Society for Industrial and Applied Mathematics, Philadelphia, 1979.

[27] P. V. Dooren and P. Dewilde. Minimal cascade factorization of real complex rational transfer matrices. *IEEE Trans. on Circuits and Systems*, 28:390–400, 1981.

[28] P. Dorato and R. K. Yedavalli, editors. *Recent Advances in Robust Control.* IEEE Press, New York, 1990.

[29] J. C. Doyle. Analysis of feedback systems with structured uncertainties. *IEE Proceedings Part D*, 129:242–250, 1982.

[30] J. C. Doyle. *Lecture Notes on Advances in Multivariable Control.* ONR/Honeywell Workshop on Advances in Multivariable Control, Minneapolis, MN, Oct. 1984.

[31] J. C. Doyle. Structured uncertainty in control system design. In *Proceedings of the 24th IEEE Conference on Decision and Control*, pages 260–265, 1985.

[32] J. C. Doyle, F. A. Francis, and A. Tannenbaum. *Feedback Control Design*, chapter 6,7. MacMillan, 1991.

[33] J. C. Doyle and A. Packard. Uncertain multivariable systems from a state space perspective. In *Proceedings of the 1987 American Control Conference*, pages 2147–2152, 1987.

[34] F. J. Doyle III. *Robustness Properties of Nonlinear Process Control and Implications for the Design and Control of a Packed Bed Reactor.* PhD thesis, California Institute of Technology, Pasadena, 1991.

[35] S. C. Eisenstat, J. W. Lewis, and M. H. Schultz. Optimal block diagonal scaling of block 2-cyclic matrices. *Lin. Alg. and Appl.*, 44:181–186, 1982.

[36] D. F. Enns. Structured singular value synthesis design example: Rocket stabilization. In *Proceedings of the 1990 American Control Conference*, pages 2514–2520, 1987.

[37] M. K. H. Fan, A. L. Tits, and J. C. Doyle. Robustness in the presense of joint parametric uncertainty and unmodeled dynamics. In *Proceedings of the 1988 American Control Conference*, pages 1195–1200, 1988.

[38] M. R. Garey and D. S. Johnson. *Computers and Intractibility: A Guide to NP-Completeness.* W. H. Freeman and Company, New York, 1983.

[39] K. Glover. All optimal Hankel-norm approximations of linear multivariable systems and their L^∞-error bounds. *International Journal of Control*, 39(6):1115–1193, 1984.

[40] K. Glover and J. C. Doyle. A state space approach to H_∞ optimal control. In H. Nijmeijer and J. M. Schumacher, editors, *Lecture Notes in Control and Information Sciences: Three Decades of Mathematical Systems Theory: A Collection of Surveys at the Occasion of the 50th Birthday of Jan C. Willems*, volume 135. Springer-Verlag, 1989.

[41] G. C. Goodwin, B. Ninness, and M. E. Salgado. Quantification of uncertainty in estimation. In *Proceedings of the 1990 American Control Conference*, pages 2400–2405, 1990.

[42] P. Grosdidier and M. Morari. Interaction measures for systems under decentralized control. *Automatica*, 22:309–319, 1986.

[43] P. Grosdidier and M. Morari. The μ-interaction measure. *Ind. Eng. Chem. Res.*, 26:1193–1202, 1987.

[44] P. Grosdidier, M. Morari, and B. R. Holt. Closed-loop properties from steady-state gain information. *Ind. Eng. Chem. Fund.*, 24:221–235, 1985.

[45] G. Gu and P. P. Khargonekar. A class of algorithms for identification in H_∞. *Automatica*, 28:299–312, 1992.

[46] G. Guardabassi and A. Locatelli. On the initialization problem in the parameter optimization of structurally constrained industrial regulators. *Large-Scale Systems*, 3:267–277, 1982.

[47] T. J. Harris, J. F. MacGregor, and J. D. Wright. Optimal sensor location with an application to a packed-bed tubular reactor. *AIChE Journal*, 26:910–916, 1980.

[48] A. J. Helmicki, C. A. Jacobson, and C. N. Nett. Control oriented system identification: a worst-case/deterministic approach in H_∞. *IEEE Trans. on Auto. Control*, AC-36:1163–1176, 1991.

[49] A. J. Helmicki, C. A. Jacobson, and C. N. Nett. Worst-case/deterministic identification in H_∞: the continuous-time case. *IEEE Trans. on Auto. Control*, AC-37:604–610, 1992.

[50] A. S. Householder. *The Theory of Matrices in Numerical Analysis*, page 123. Blaisdell Publishing Company, New York, 1964.

[51] M. Hovd. *Studies on Control Structure Selection and Design of Robust Decentralized and SVD Controllers.* Dr. Ing. thesis, University of Trondheim-NTH, Trondheim, Norway, 1992.

[52] M. Hovd, 1992. personal communication.

[53] M. Hovd, R. D. Braatz, and S. Skogestad. On the structure of the robust optimal controller for a class of problems. *Automatica*, 1992. submitted.

[54] M. Hovd, R. D. Braatz, and S. Skogestad. On the structure of the robust optimal controller for a class of problems. In *Proceedings of the 12th IFAC World Congress,* Mini-Symposium M-1, 1993. to be presented.

[55] P. Jackson. Applying μ-synthesis to missile autopilot design. In *Proceedings of the 29th IEEE Conference on Decision and Control*, pages 2993–2998, 1990.

[56] R. L. Kosut, M. Lau, and S. Boyd. Identification of systems with parametric and nonparametric uncertainty. In *Proceedings of the 1990 American Control Conference*, pages 2412–2417, 1990.

[57] B. Kouvaritakis and H. Latchman. Necessary and sufficient stability condition for systems with structured uncertainties: The major principal direction alignment principle. *International Journal of Control*, 42:575–598, 1985.

[58] B. Kouvaritakis and H. Latchman. Singular-value and eigenvalue techniques in the analysis of systems with structured perturbations. *International Journal of Control*, 41:1381–1412, 1985.

[59] J. M. Krause and P. P. Khargonekar. Parameter identification in the presence of nonparametric uncertainty. *Automatica*, 26:113–123, 1990.

[60] S. Kumar and J. H. Seinfeld. Optimal location of measurements in tubular reactors. *Chem. Eng. Sci.*, 33:1507–1516, 1978.

[61] D. L. Laughlin, M. Morari, and R. D. Braatz. Robust performance of cross-directional basis-weight control in paper machines. *Automatica*, 1992. in press.

[62] D. K. Le, O. D. I. Nwokah, and A. E. Frazho. Multivariable root loci and robustness analysis for soft failures. In *Proceedings of the 11th IFAC World Conference*, 1990.

[63] D. K. Le, O. D. I. Nwokah, and A. E. Frazho. Multivariable decentralized integral controllability. *International Journal of Control*, 54:481–496, 1991.

[64] J. H. Lee, R. D. Braatz, M. Morari, and A. Packard. Screening tools for robust control structure selection. Presented at the *AIChE Annual Meeting*, Paper 152p, 1991.

[65] J. H. Lee, R. D. Braatz, M. Morari, and A. Packard. Screening tools for robust control structure selection. *Automatica*, 1992. submitted.

[66] J. H. Lee, R. D. Braatz, M. Morari, and A. Packard. Robust control structure selection. CDS Technical Memo CIT-CDS 93-0XX, California Institute of Technology, Pasadena, CA 91125, 1993.

[67] J. H. Lee and M. Morari. Robust measurement selection. *Automatica*, 27:519–527, 1991.

[68] L. Ljung. *System Identification: Theory for the User.* Prentice-Hall, Englewood Cliffs, New Jersey, 1987.

[69] C. McCarthy and G. Strang. Optimal conditioning of matrices. *SIAM J. Numer. Anal.*, 10:370–388, 1973.

[70] M. Milanese and A. Vicino. Optimal estimation theory for dynamics systems with set membership uncertainty: An overview. *Automatica*, 27:997–1009, 1991.

[71] M. Morari. Robust stability of systems with integral control. *IEEE Trans. on Auto. Control*, AC-30:574–577, 1985.

[72] M. Morari and J. C. Doyle. A unifying framework for control system design under uncertainty and its implications for chemical process control. In M. Morari and T. J. McAvoy, editors, *Third International Conference on Chemical Process Control (CPC III)*. Elsevier, Amsterdam, 1986.

[73] M. Morari, C. E. Garcia, J. H. Lee, and D. M. Prett. *Model Predictive Control.* Prentice-Hall, Inc., Englewood Cliffs,N.J., 1992. in preparation.

[74] M. Morari and J. H. Lee. Model predictive control: The good, the bad, and the ugly. In W. H. Ray and Y. Arkun, editors, *Fourth International Conference on Chemical Process Control (CPC IV)*, pages 419–444. Elsevier, Amsterdam, 1991.

[75] M. Morari, N. L. Ricker, D. B. Raven, Y. Arkun, N. Bekiaris, R. D. Braatz, M. S. Gelormino, T. R. Holcomb, S. M. Jalnapurkar, J. H. Lee, Y. Liu, S. L. Oliveira, A. R. Secchi, S.-Y. Yang, and Z. Q. Zheng. Model predictive control toolbox (MPC-tools): Matlab functions for the analysis and design of model predictive control systems, 1992.

[76] M. Morari and E. Zafiriou. *Robust Process Control.* Prentice-Hall, Inc., Englewood Cliffs, N.J., 1989.

[77] K. G. Murty and S. N. Kabadi. Some NP-complete problems in quadratic and nonlinear programming. *Mathematical Programming*, 39:117–129, 1987.

[78] C. N. Nett. Decentralized control system design for a variable-cycle gas turbine engine. In *Proceedings of the 1990 American Control Conference*, 1990.

[79] C. N. Nett and K. D. Minto. A quantitative approach to the selection and partitioning of measurements and manipulations for the control of complex systems. In *Proceedings of the 1989 American Control Conference*, 1989.

[80] C. N. Nett and J. A. Uthgenannt. An explicit formula and an optimal weight for the 2-block structured singular value interaction measure. In *Proceedings of the 1987 American Control Conference*, pages 506–511, 1987.

[81] O. D. I. Nwokah, A. E. Frazho, and D. K. Le. A note on decentralized integral controllability. *International Journal of Control*, 57:485–494, 1993.

[82] O. D. I. Nwokah and R. Perez. On multivariable stability in the gain space. In *Proceedings of the 1990 IEEE Conference on Decision and Control*, pages 328–333, 1990.

[83] O. D. I. Nwokah and R. Perez. On multivariable stability in the gain space. *Automatica*, 27:975–983, 1991.

[84] E. E. Osborne. On pre-conditioning of matrices. *Journal of Assoc. Comp. Mach.*, 7:338–345, 1960.

[85] A. Packard and J. C. Doyle. The complex structured singular value. *Automatica*, 29:71–109, 1993.

[86] A. Packard and P. Pandey. Continuity properties of the real/complex structured singular value. *IEEE Trans. on Auto. Control*, 1993. in press.

[87] A. K. Packard. *What's New With μ: Structured Uncertainty in Multivariable Control.* PhD thesis, University of California, Berkeley, 1988.

[88] A. K. Packard. LFT optimal control. In *IMA Workshop*, Minneapolis, September 1992.

[89] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity.* Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[90] J. Partington. Robust identification and interpolation in H_∞. *International Journal of Control*, 54:1281–1290, 1991.

[91] D. E. Reeves, C. N. Nett, and Y. Arkun. Control configuration design for complex systems: A practical theory. Technical report, Georgia Tech, 1992.

[92] R. T. Reichert. Robust autopilot design using μ-synthesis. In *Proceedings of the 1990 American Control Conference*, pages 2368–2373, 1991.

[93] J. Rohn and S. Poljak. Checking robust nonsingularity is NP-hard. *Mathematics of Control, Signals, and Systems*, 1992. in press.

[94] M. G. Safonov. Imaginary-axis zeros in multivariable H_∞-optimal control. In R. F. Curtain, editor, *Modeling, Robustness and Sensitivity Reduction in Control Systems.* Springer-Verlag, New York, 1987.

[95] L. Sartor. *Slot Coating: Fluid Mechanics and Die Design.* PhD thesis, University of Minnesota, 1990.

[96] L. E. Scriven and W. J. Suszynski. Take a close look at coating problems. *Chem. Eng. Prog.*, 86(9):24–29, 1990.

[97] R. Sezginer and M. Overton. The largest singular value of $e^X A e^{-X}$ is convex on convex sets of commuting matrices. *IEEE Trans. on Auto. Control*, 35:229–230, 1990.

[98] J. S. Shamma. Necessity of the small-gain theorem for time-varying and nonlinear systems. *IEEE Trans. on Auto. Control*, 36:1138–1147, 1991.

[99] A. Shapiro. Weighted minimum trace factor analysis. *Psychometrika*, 47:243–264, 1982.

[100] A. Shapiro. Optimal block diagonal l_2 scaling of matrices. *SIAM J. Numer. Anal.*, 22:81–94, 1985.

[101] A. Shapiro. Upper bounds for nearly optimal diagonal scaling of matrices. *Lin. and Mult. Alg.*, 29:145–147, 1991.

[102] S. Skogestad and M. Morari. Design of resilent processing plants—IX. Effect of model uncertainty on dynamic resilience. *Chem. Eng. Sci.*, 42:1765–1780, 1987.

[103] S. Skogestad and M. Morari. Implications of large RGA elements on control performance. *Ind. Eng. Chem. Res.*, 26:2323–2330, 1987.

[104] S. Skogestad and M. Morari. Some new properties of the structured singular value. *IEEE Trans. on Auto. Control*, AC-33:1151–1154, 1988.

[105] S. Skogestad and M. Morari. Robust performance of decentralized control systems by independent designs. *Automatica*, 29:119–125, 1989.

[106] S. Skogestad, M. Morari, and J. C. Doyle. Robust control of ill-conditioned plants: High purity distillation. *IEEE Trans. on Auto. Control*, AC-33:1092–1105, 1988.

[107] R. S. R. Smith. *Model Validation for Uncertain Systems.* PhD thesis, California Institute of Technology, Pasadena, 1990.

[108] G. W. Stewart and J. G. Sun. *Matrix Perturbation Theory.* Academic Press, Boston, Massachusetts, 1990.

[109] J. Stoer and C. Witzgall. Transformations by diagonal matrices in a normed space. *Numer. Math.*, 4:158–171, 1962.

[110] A. Van Der Sluis. Condition numbers and equilibration of matrices. *Numer. Math.*, 14:14–23, 1969.

[111] A. Van Der Sluis. Condition, equilibration and pivoting in linear algebraic systems. *Numer. Math.*, 15:74–86, 1970.

[112] S. A. Vavasis. Quadratic programming is in NP. *Information Processing Letters*, 36:73–77, 1990.

[113] R. J. Veillette, J. V. Medanić, and W. R. Perkins. Design of reliable control systems. *IEEE Trans. on Auto. Control*, 37:290–304, 1992.

[114] M. Vidyasagar. *Control System Synthesis: A Factorization Approach.* MIT Press, Cambridge, Massachusetts, 1985.

[115] G. A. Watson. An algorithm for optimal l_2 scaling of matrices. *IMA Journal of Numerical Analysis*, 11:481–492, 1991.

[116] C. J. Webb. *Robust Control Strategies for a Fixed Bed Chemical Reactor.* PhD thesis, California Institute of Technology, Pasadena, 1990.

[117] R. Weber and C. Brosilow. The use of secondary measurements to improve control. *AIChE Journal*, 18:614–623, 1972.

[118] P. M. Young. *Robustness With Parametric and Dynamic Uncertainties.* PhD thesis, California Institute of Technology, Pasadena, 1993.

[119] P. M. Young, M. P. Newlin, and J. C. Doyle. μ analysis with real parametric uncertainty. In *Proceedings of the 30th IEEE Conference on Decision and Control*, pages 1251–1256, 1991.