Controller Design by Minimax Optimization

A Youla Approach

Maartje Nillesen

SIGNAAL

Department of
Mathematics

RuG

Master's thesis

Controller Design by Minimax Optimization

A Youla Approach

Maartje Nillesen

University of Groningen
Department of Mathematics
P.O. Box 800
9700 AV Groningen

May 1999

Preface

In September 1998 I started my research project at Hollandse Signaalapparaten B.V. in Hengelo. During this period I did research in developing a method for designing and optimizing low-order, digital controllers.
For me this was a great opportunity to experience the possibilities and problems of designing feedback systems in practice.

Special thanks go to my supervisors.
To Ruth Curtain (R*u*G) for all her help and careful reading. Her creative ideas and guidance in how to do mathematical research were truly helpful.
To Wim Vaassen (Signaal) for his continuous support. His critical, but always very practical way of using mathematics taught me much about the advantages and limitations of mathematics in real life.

I thank Rienk Bakker (Signaal) for his encouragement and inspiring interest in my research. Our conversations helped me a lot to understand more of the complicated problems in control theory.
Finally, I would like to mention that the fine proportion of serious work and humour made my time at Signaal above all a very pleasant one.

Groningen, May 1999.

Maartje Nillesen.

Contents

Chapter 1

Introduction

Hollandse Signaalapparaten B.V. in Hengelo is a company specialized in designing and producing advanced communication and defense systems.
Most of the work is focused on naval combat systems, for instance the design of command and control systems, sensors and radar.
In the development of these systems, feedback control theory plays an important role.
For example, a camera on a sailing ship, subject to roll motions should be stabilized at a certain fixed position.
Another example can be found in the design of radar systems. These systems use the technique of so called servo systems to track an airplane.

In general, the objective in designing a control system is to make a certain output y behave in a desired way by manipulating the input u.
The resulting feedback system should be stable and should meet certain design specifications. These design specifications partly consist of performance criteria, which specify the desired behaviour of the system. They also consist of certain stability requirements which guarantee robust stability.

The design of such a feedback system could be done by solving the well known *mixed sensitivity problem*.
This problem deals with finding a stabilizing controller which minimizes the weighted H_∞-norm of the sensitivity function S and the complementary sensitivity function T. The functions S and T represent the 'performance' of the system.
Weighting functions are used to shape S and T such that the design specifications are met.

There exist standard H_∞ methods to solve the mixed sensitivity problem.
The problem with these methods is that continuous, stable, rational weighting functions are required, whereas in practice these weighting functions are not known. Usually the weights are given by frequency dependent bounds derived from the design specifications.
As will be explained in this report, this fact makes the classical H_∞ methods very impractical for the design of *low*-order controllers.
It must be noticed that there is not much literature available about the mixed sensitivity problem when the weights are not given by continuous, stable, rational functions.

At Hollandse Signaalapparaten B.V. a *Minimax* optimization method was developed to be able to perform *direct*, *low*-order controller design.
Although this direct low-order controller design seems to be a very practical and interesting method, the resulting optimization problem is a very complex, non-convex problem.

In this report the possibility of using the Youla parameterization is investigated to simplify this minimax optimization problem.
The Youla parameterization is a parameterization of all stabilizing controllers, which uses the theory of coprime factorizations. The motivation for using this parameterization is that the original minimax optimization can now be converted to an easier, better posed optimization problem.

Chapter 2 describes the mixed sensitivity problem and the minimax approach (K-optimization) used by Hollandse Signaalapparaten.

After some theory about coprime factorizations and the introduction of the famous Youla parameterization, chapter 3 presents the new Youla minimax optimization method.

Finally, chapter 4 gives the obtained numerical results of this method when implemented in MATLAB.

Chapter 2

The Mixed Sensitivity Problem

In control system design one of the most discussed problems is loop shaping. This concerns important issues such as stabilization, tracking and disturbance attenuation.
In general, in designing a control system, the aim is to manipulate the control action u such that the output y behaves in a desired way. For example, a control objective in the servo problem is that the output tracks a reference input as closely as possible.

The question whether these and other design specifications can be met, using the technique of loop shaping, is discussed in this chapter.

2.1 Loop Shaping

Consider the following feedback configuration, where $G(z)$ is a real rational proper transfer function of a plant, and $K(z)$ is the transfer function of the controller.

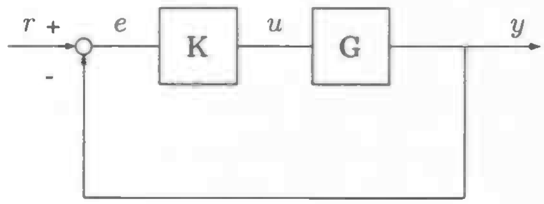

Figure 2.1: *Standard feedback configuration.*

In this figure y is the observed output, u is the control input and r is the reference input. The tracking error e is defined as $r - y$.
In radar systems for example, one can think of this tracking error as being the error between

the estimated position and the actual position of an airplane. It is obvious that for good tracking, this error should be as small as possible. The problem of keeping this error small is called the *servo problem.*

In this report we only consider single-input/single-output (SISO) systems, which means that all signals (u, r, y, e) are scalars.
The closed loop transfer function T (from r to y), is equal to $GK(1+GK)^{-1}$:

$$y = GKe = GK(r-y) \qquad \frac{y}{r} = \frac{GK}{1+GK}$$

This transfer function T represents the output behaviour of the system. Keeping the servo error e close to zero in a certain frequency range corresponds to T being nearly one in that range.

The basic problem we consider in feedback system design is the following:

Find a controller K such that the closed loop system

(1) is stable and,
(2) acquires some additional desired properties.

The first part of this problem (*stability*) is of course essential. A controller which does not stabilize our plant is in fact useless.
The latter part concerns an equally important issue in feedback system design: *Loop shaping.*

We now concentrate on stability. Before this can be defined we need some other definitions. We first introduce the function space H^1_∞ and the H_∞-norm for discrete-time systems.

Let $\mathcal{D}$ denote the closed unit disc

$$\mathcal{D} = \{z \in \mathbb{C} \quad | \quad |z| \leq 1\}$$

The complement of $\mathcal{D}$ in $\mathbb{C}$ is denoted by :

$$\mathcal{D}^C = \{z \in \mathbb{C} \quad | \quad |z| > 1\}$$

A function $F : \mathcal{D}^C \rightarrow \mathbb{C}$ is bounded on $\mathcal{D}^C$ if

$$\exists\, r > 0 \ \text{ such that } \ |F(z)| \leq r \quad \forall z \in \mathcal{D}^C$$

Definition 2.1
A function $F : \mathcal{D}^C \rightarrow \mathbb{C}$ is in the function space H^1_∞ if F is analytic and bounded on $\mathcal{D}^C$, i.e.,

$$\sup_{z \in \mathcal{D}^C} |F(z)| < \infty \tag{2.1}$$

If $F \in H^1_\infty$ we can use the maximum modulus theorem to replace $\mathcal{D}^C$ by $\partial\mathcal{D}$ in (2.1), where $\partial\mathcal{D}$ denotes the unit circle:

$$\partial\mathcal{D} = \{z \in \mathbb{C} \quad | \quad |z| = 1\}$$

and therefore

$$\| F \|_\infty = \sup_{z \in \mathcal{D}^C} |F(z)| = \sup_{z \in \partial\mathcal{D}} |F(z)|$$

For single-input/single-output (SISO) systems this H_∞-norm is simply the peak value of the magnitude plot of $F(z)$.

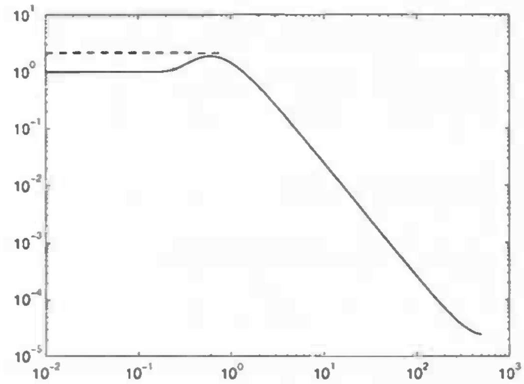

Figure 2.2: H_∞*-norm.*

Definition 2.2
A transfer function F is called **stable** if $F \in H^1_\infty$.

The fact that F is in H^1_∞ implies that F is proper ($|F(\infty)|$ is finite) and F has all its poles inside the unit disc D.

Definition 2.3
Suppose M is a transfer matrix whose components are transfer functions. Then this transfer matrix M is called stable if all components are in H^1_∞.
With $\mathcal{M}H^1_\infty$ we denote the class of matrices with components in H^1_∞.

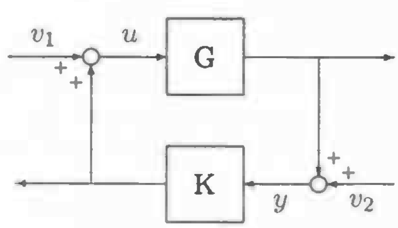

Figure 2.3: *Stability diagram.*

From now on, suppose that $G(z)$ is a linear, time-invariant, discrete-time SISO system. Consider the feedback system given in figure 2.3.
In terms of transfer matrices this system is described by:

$$\begin{bmatrix} u \\ y \end{bmatrix} = \begin{bmatrix} (1-GK)^{-1} & K(1-GK)^{-1} \\ G(1-GK)^{-1} & (1-GK)^{-1} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

Definition 2.4
The closed loop system shown in figure 2.3 is said to be *stable* if the transfer matrix

$$\begin{bmatrix} (1-GK)^{-1} & K(1-GK)^{-1} \\ G(1-GK)^{-1} & (1-GK)^{-1} \end{bmatrix}$$

from $(v_1\ v_2)^T$ to $(u\ y)^T$ belongs to $\mathcal{M}H^1_\infty$.

If the system is stable, then for all bounded inputs $(v_1\ v_2)^T$, the outputs u and y are also bounded.

Under the constraint of K being stabilizing, the main issue in the design is *loop shaping*.
As mentioned above, one of the design objectives is to make the tracking error as small as possible. Another design specification is to minimize the influence of the measurement noise.
To meet these (and other) design objectives loop shaping is a useful tool.
In general, the idea in loop shaping is to put conditions on certain closed loop functions by choosing appropriate weighting functions.
We will first specify which closed loop functions we consider and explain why we do this.

2.1.1 Sensitivity Functions

In the process of loop shaping, two functions play an important role: The *sensitivity function* and the *complementary sensitivity function.*

The sensitivity function S is defined as $(1+GK)^{-1}$ and is equal to the transfer function from r to e:

$$e = r - y = r - GKe \qquad \frac{e}{r} = \frac{1}{1+GK}$$

This function characterizes the sensitivity of the control system output to external disturbances. More explicitly it gives you information about how sensitive the closed loop transfer function T is to an infinitesimal perturbation in G. [1]
To see this, take the limiting ratio of a relative perturbation in T ($\frac{\Delta T}{T}$) to a relative perturbation in G ($\frac{\Delta G}{G}$):

$$lim_{\Delta G \to 0} \frac{\Delta T/T}{\Delta G/G} = \frac{dT}{dG}\frac{G}{T} = \frac{1}{1+GK} = S$$

As we see, $T = 1 - S$, and this is the reason that T is called the complementary sensitivity function.

Both S and T have certain desired characteristics which we will discuss later on, but the fact that

$$S + T = 1$$

already gives us some idea about the problems we meet when we wish to shape S and T simultaneously.
For optimal disturbance attenuation (i.e., the closed loop system is not sensitive for output disturbances), S should be equal to 0 at all frequencies.
In contrast, to make the output insensitive for measurement noise, T should be small at high frequencies. Evidently, as S and T are complementary functions, the above requirements are conflicting.

To shape S and T we introduce weighting functions W_1 and W_2 such that if

$$|S| < {W_1}^{-1}$$

$$|T| < {W_2}^{-1}$$

S and T have the desired shape and the design specifications are met.
For instance, to shape S we use certain performance criteria, which are a measure for 'goodness of tracking'.

Suppose that the following specifications guarantee good performance :
(Continuous-time system)

$$\begin{cases} |S(j\omega)| \leq \delta & 0 < \omega \leq \omega_\beta \\ |S(j\omega)| \leq \gamma & \forall\omega > \omega_\beta \end{cases}$$

where ω_β is some fixed frequency point.
As S is a real rational transfer function, we have $|S(-j\omega)| = |S(j\omega)|$. We therefore only consider positive frequencies ω. δ and γ are constants with $0 < \delta < 1$, $\gamma > 1$.
We can rewrite these criteria by choosing a frequency dependent weighting function W_1:

$$|W_1(j\omega)S(j\omega)| \leq 1 \qquad \forall\omega > 0$$

with

$$W_1(j\omega) = \begin{cases} \delta^{-1} & 0 < \omega \leq \omega_\beta \\ \gamma^{-1} & \forall\omega > \omega_\beta \end{cases}$$

Choosing the weights W_1 and W_2 is quite an important issue as by designing W_1 and W_2 we actually design S and T.

2.1.2 Robust Stability and Performance

Assume that we have constructed a controller K that stabilizes G.
In control theory we always work with models of, for example, a physical system. In practice, we never have an exact description of the system, we never know the precise value of the parameters of the model.
Because of this parametric uncertainty we do not only want our controller to stabilize the

original system G, but also a slightly perturbed system $\tilde{G}$.
This property is what we call *robust stability*.

Consider the following set Γ of perturbed plant transfer functions:

$$\Gamma = \{\tilde{G} : \tilde{G} = (1 + \Delta W_2)G\}$$

where G is the nominal plant transfer function, W_2 is a fixed stable weighting function, and Δ is a perturbation, a variable stable transfer function with $\| \Delta \|_\infty \leq 1$.
This uncertainty model (Multiplicative perturbation) is a disc-like uncertainty model and ΔW_2 is the normalized plant perturbation away from 1:

$$\frac{\tilde{G}}{G} - 1 = \Delta W_2$$

A controller K stabilizes G **robustly** if it stabilizes every $\tilde{G}$ in Γ.
The next theorem states the robust stability condition[1].

Theorem 2.5
K stabilizes G robustly if and only if

$$\| W_2 T \|_\infty < 1 \tag{2.2}$$

Another very important feature is *performance*.
The performance of, for example, a tracking system could be measured by the size of the tracking error signal. In general, the performance of a system is measured by the sensitivity function S, and is considered satisfactory if

$$\| W_1 S \|_\infty < 1 \tag{2.3}$$

where W_1 is a suitable weighting function.

Combining (2.2) and (2.3) gives us a condition for simultaneously achieving performance and robust stability: (See [1])

$$\| max(|W_1 S|, |W_2 T|) \|_\infty < 1 \tag{2.4}$$

This last inequality which consists of conditions on both S and T leads us to a well-known problem in H_∞-control theory, *the mixed sensitivity problem*, which we discuss in the next section.

[1] For explanation and proof see for example [1]

2.2 The Mixed Sensitivity Problem

The mixed sensitivity problem is defined as:

Find a controller K that stabilizes G and minimizes the H_∞-norm of

$$\begin{bmatrix} W_1 S \\ W_2 T \end{bmatrix}$$

This is a special case of the standard H_∞ problem and deals with minimizing the weighted H_∞-norm of both S and T.

Although $\left\| \begin{bmatrix} W_1 S \\ W_2 T \end{bmatrix} \right\|_\infty$ is equal to

$$\sup_{\omega \in \mathbb{R}} \sqrt{|W_1(j\omega)S(j\omega)|^2 + |W_2(j\omega)T(j\omega)|^2} \tag{2.5}$$

and this is not exactly the condition for performance and robust stability we stated in the last section, we have the following inequality (compare (2.4)):

$$\frac{1}{\sqrt{2}}(|W_1S|^2 + |W_2T|^2)^{\frac{1}{2}} \leq \max(|W_1S|, |W_2T|) \leq (|W_1S|^2 + |W_2T|^2)^{\frac{1}{2}} \tag{2.6}$$

Clearly these norms are not the same, but they are equivalent. Moreover, the actual solutions we obtain solving the optimization problem using different norms have the same qualitative properties. [1]

By solving the mixed sensitivity problem, we basically use the technique of loop shaping. We choose appropriate frequency dependent weighting functions $\tilde{W_1}$ and $\tilde{W_2}$ such that if $|S| < \tilde{W_1}$, $|T| < \tilde{W_2}$ the design specifications are met. We now solve the mixed sensitivity problem with $W_1 = \tilde{W_1}^{-1}$, $W_2 = \tilde{W_2}^{-1}$.

If we find a stabilizing K such that

$$\left\| \begin{bmatrix} W_1 S \\ W_2 T \end{bmatrix} \right\|_\infty < 1$$

the functions $|S|$ and $|T|$ will have the desired characteristics.

One of the factors that makes the problem of simultaneously minimizing the (weighted) S and T complicated is the following.
If we make S nearly zero (for good disturbance rejection), then T will be nearly one, which means measurement noise is not filtered out. Conversely, making T small results in S being nearly one. This means there always is a trade-off between these conflicting requirements.
This problem is partially solved by keeping S and T small in a certain frequency range. This is illustrated in figure 2.4.

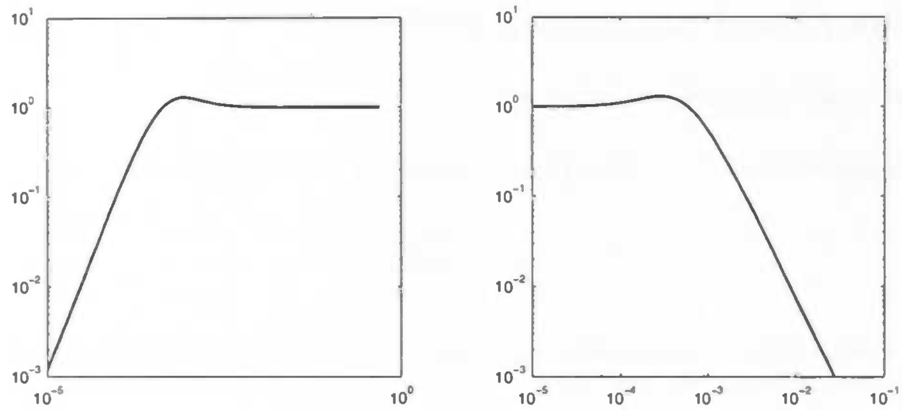

Figure 2.4: *Typical graphs of S and T respectively.*

2.3 Standard H_∞ Methods

Solving the mixed sensitivity problem standard H_∞ methods could be used. There are good numerical methods to design an optimal H_∞ controller (See for example MATLAB : *robust control toolbox*).
We now briefly discuss these standard methods.

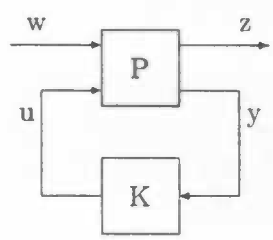

Figure 2.5: *Standard feedback configuration.*

Suppose that P (see figure 2.5) is partitioned as

$$\begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}$$

Figure 2.5 represents the following algebraic equations:

$$z = P_{11}w + P_{12}u$$

$$y = P_{21}w + P_{22}u$$

$$u = Ky$$

The transfer function of the actual process or plant is the sub matrix P_{22}.
By eliminating y and u and using $u = Ky$ it is easily seen that

$$z = [P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21}]w \tag{2.7}$$

If H is defined as $[P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21}]$, the *standard H_∞ problem* is defined as:

$$\text{minimize} \parallel \text{H} \parallel_\infty \tag{2.8}$$

for all stabilizing K.

Defining G and K as in figure 2.1 and comparing (2.7) with the mixed sensitivity problem defined in section 2.2, it is easy to deduce we should choose

$$P_{11} = \begin{bmatrix} W_1 \\ 0 \end{bmatrix}, \quad P_{12} = \begin{bmatrix} -W_1G \\ W_2G \end{bmatrix}, \quad P_{21} = I, \quad P_{22} = -G$$

To use standard H_∞ methods it is necessary to design appropriate continuous, rational,stable weighting functions $\tilde{W}_1$ and $\tilde{W}_2$. The plant augmented with these weighting functions forms the generalized plant.
Now an optimal H_∞ controller is computed for this generalized plant. The order of the controller obtained by this method is equal to the order of the generalized plant.

Suppose the question is now to solve the mixed sensitivity problem described above, but the order of the controller is **fixed**:

Find a compensator K of order k, that stabilizes G and minimizes the H_∞-norm of

$$\begin{bmatrix} W_1S \\ W_2T \end{bmatrix} \tag{2.9}$$

where k is some fixed constant.

This is not an unreasonable assumption, as in practice (i.e., we actually design a controller for a given physical system), low controllers are preferable, so we **fix** the order at some low constant k.
In the remainder of this report we will consider this fixed order mixed sensitivity problem.

We also make the assumption that the weighting functions are defined by frequency dependent bounds derived from the design specifications.
In general these functions are *not* described by continuous, stable, rational transfer functions. For example, a design objective for T might be that [2]

$$|T(e^{2\pi i\omega/f_s})| \leq \alpha \qquad 0 < \omega \leq \omega_\beta$$

where $\omega \in \Omega = [0, \ldots, \frac{1}{2}f_s]$, a frequency vector.
f_s denotes the sample frequency and $0 < \omega_\beta < \frac{1}{2}f_s$. [3]

[2] Discrete-time specifications, for more information about discrete-time systems see [2]
[3] $\frac{1}{2}$ f_s is the Nyquist frequency, see [2]

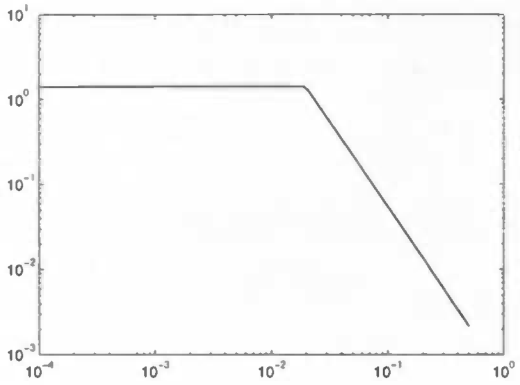

Figure 2.6: *Bounds on T.*

To filter out measurement noise T is required to roll off at 2 decade/decade for $\omega \geq \omega_\beta$. This means the slope of the line is -2 on a log-log scale. (See figure 2.6)

To be able to apply standard H_∞ methods to this modified design problem, we first have to fit a continuous, stable, rational transfer function on the bounds on T described above. As this function is not differentiable at ω_β, it is hard to make a good continuous stable rational approximation of this function.

Generally speaking, to get a reasonable fit on specification bounds, the order of the fitted transfer function will be high. As a result, due to these high order weighting functions, the H_∞ optimization results in a very high order controller, approximately of the order of the plant G plus the order of the weighting functions.
As we required a **low** order controller, the H_∞ design is not practical.
It is of course possible to reduce the order of this high order controller using model reduction techniques. Obviously, this reduced controller is no longer optimal, and if the desired order is much smaller than the order of obtained high-order controller, it is even doubtful if this low order controller still meets the specifications.

The question arises whether there is a more efficient, reliable way of low-order controller design.

2.4 The Minimax Optimization Method

A possible solution for the problems described in the preceding section is to rewrite the mixed sensitivity problem as described in (2.9) as a *minimax* optimization problem.
This is the method currently used at Hollandse Signaalapparaten. One of the advantages is that it is now possible to do **direct** low order design.

In general, a minimax problem has the following structure:

$$\min_x \max_y \; F(x,y) \quad \text{such that} \quad G(x) \leq 0 \tag{2.10}$$

In this problem $F(x)$ is the *goal function*, whose maximum, with respect to y, should be minimized, with respect to x. $G(x)$ is a *constraint function*.

Let us first try to reformulate the mixed sensitivity problem as a minimax problem. By comparing (2.9) and (2.10) we obtain the following minimax problem:

$$\min_{K\in\Gamma} \max_{\omega\in\mathbb{R}} \sqrt{|W_1(j\omega)S(j\omega)|^2 + |W_2(j\omega)T(j\omega)|^2} \tag{2.11}$$

such that K is a stabilizing controller.

Γ consists of all controllers K of a certain fixed order k. In this minimax problem the goal function is the weighted H_∞-norm of S and T and the constraint function is that the closed loop system is stable.
In section 2.2 we discussed different, but equivalent norms, and we have seen from equation (2.6) that the following is is an equivalent problem:

$$\min_{K\in\Gamma} \max_{\omega\in\mathbb{R}} \{\max\left(|W_1(j\omega)S(j\omega)|, |W_2(j\omega)T(j\omega)|\ \right)\} \tag{2.12}$$

such that K is a stabilizing controller.

From now on this optimization criterion will be used to design a sub-optimal controller.

The goal function is:

$$\max\ \{|W_1S|, |W_2T|\} = \max\ \left\{ \left|W_1\frac{1}{1-GK}\right|, \left|W_2\frac{GK}{1-GK}\right| \right\}$$

The optimization criterion is now exactly the criterion for simultaneously achieving performance and robust stability described in (2.4) instead of (2.5).

The idea of this minimax approach, which we will call the **K-optimization method** from now on, is to use the parameters of the controller K as tuning parameters. By varying these parameters the value of the goal function $\max(|W_1S|, |W_2T|)$ changes.

As in practice the question is to design a digital controller for a *discrete* time system we now apply the minimax approach to discrete time systems.
The optimization constraint is that K stabilizes G, so for discrete time systems the constraint function is that all closed loop eigenvalues have modulus less than one:

$$|\lambda(A_{cl})| < 1$$

where A_{cl} is the A-matrix of the closed loop system.

Since numerically we can only maximize over a certain grid, we replace $\omega \in \mathbb{R}$ by $\omega \in \Omega$, where Ω is a discrete frequency grid.

An essential difference between the K-optimization method used at Signaal and the classical H_∞ approach described in section 2.3 lies in the description of the weighting functions.

The H_∞ methods required continuous, stable, rational weighting functions. Since in practice we do not know the weighting functions W_1 and W_2, the weights first have to be constructed. This fact is, as explained before, a major disadvantage in the standard H_∞ algorithms.

In the K-optimization method the weights are represented by a set of discrete frequency points which determine the upper bounds on S and T. Instead of using continuous weighting functions of the complex variable $z = e^{2\pi i\omega/f_s}$, the weighting functions are now designed *graphically* on a frequency grid Ω of the real variable ω.
The idea is to 'draw' the weights directly on the screen, by specifying their magnitude $|W_1(z)|$ and $|W_2(z)|$. This is illustrated by figure 2.7.

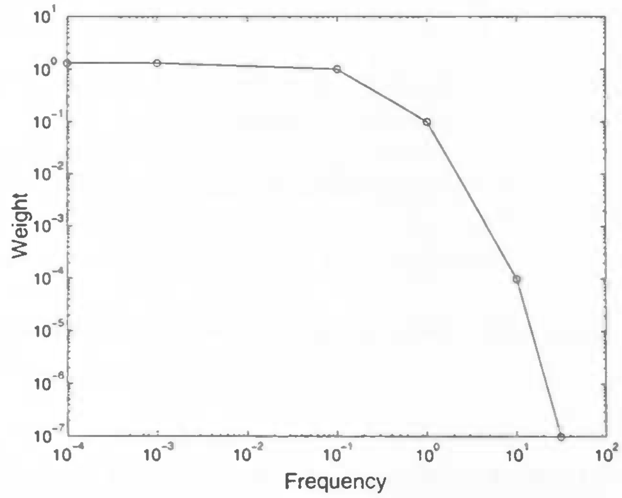

Figure 2.7: *Graphically designed weighting function.*

An advantage of this approach is that design specifications can now be directly converted into weighting functions. So instead of defining weighting functions W_1 and W_2 of the complex variable z and then taking the absolute value we now directly define (graphically) the magnitude functions

$$|W_1(z)| := \mathcal{W}_1(\omega)$$
$$|W_2(z)| := \mathcal{W}_2(\omega)$$

According to these new weighting functions it makes sense to solve the following:

$$\min_{K\in\Gamma} \max_{\omega\in\Omega} \{\max\left(\mathcal{W}_1(\omega)|S(z)|, \mathcal{W}_2(\omega)|T(z)| \right) \} \quad \text{such that} \quad |\lambda(A_{cl})| < 1 \qquad (2.13)$$

As a consequence of the description of the weighting functions, there is no need for a continuous stable approximation of the weights and direct low order control design is possible.

The minimax problem described in (2.13) is implemented in MATLAB.
After having fixed a starting estimate of a stabilizing controller of order k, the function MINIMAX from the MATLAB *Optimization Toolbox* is used to optimize the goal function over this grid of frequency points.

Although the advantage of doing direct low-order design is evident, we also face some difficult problems.
The set of all stabilizing controllers has quite a complicated structure. It is possible that varying the parameters of a stabilizing controller slightly may result in a destabilizing controller.
Another problem is that we are optimizing over a non-convex subspace.
Due to the non-convexity of this problem, the algorithm may converge to a local optimum. Different starting estimations converge to different optima, so we can never tell whether we found the optimal controller or not.

Because of the complexity of this problem, in the next chapter the possibility of using a parameterization of all stabilizing controllers is investigated to simplify the optimization criterion.

Chapter 3

The Youla Parameterization Approach

In this chapter we investigate the possibility of using a parameterization of all stabilizing controllers to solve the minimax optimization problem described in chapter 2. The parameterization we use is the famous Youla parameterization. After we have done this we can rewrite (2.13) as an easier, linear optimization problem.

The idea behind this parameterization is to parameterize all controllers K which stabilize G, via a parameter matrix Q in $\mathcal{M}H_\infty^1$.

To do this we need some theory about coprime factorizations. By using state space formulas for doubly coprime factorizations of stabilizable systems we find an observer-based controller which, together with the parameter, Q leads to the Youla parameterization.

3.1 Coprime Factorizations

Let G be a linear, time-invariant, discrete-time SISO system described by:

$$x_{k+1} = Ax_k + Bu_k \tag{3.1}$$
$$y_k = Cx_k + Du_k \tag{3.2}$$

I.e. $G(z) = C(zI - A)^{-1}B + D$

Suppose $G(z)$ is *stabilizable* and *detectable*.
The fact that G is stabilizable means that there exists a matrix F such that $A + BF$ is stable.
Detectability implies that there exists a matrix H such that $A + HC$ is stable.
For discrete-time systems, a matrix X is stable if and only if all its eigenvalues are inside the open unit disc $\mathcal{D}$. ($|\lambda(X)| < 1$)

We now formulate some important definitions and lemmas, without proof.
Proofs and more information about coprime factorizations can be found in for instance [3] and [4]
All matrices denote matrices whose components are transfer functions.

Definition 3.1
Two matrices F, G in $\mathcal{M}H^1_\infty$ are *right coprime* if they have equal number of columns and there exist matrices X, Y in $\mathcal{M}H^1_\infty$ such that

$$XF + YG = I$$

Such an equation is called a *Bezout* identity.
Two matrices F,G in $\mathcal{M}H^1_\infty$ are *left coprime* if they have equal number of rows and there exist matrices X, Y in $\mathcal{M}H^1_\infty$ such that

$$FX + GY = I$$

Lemma 3.2
For each proper real-rational matrix G we can find eight matrices in $\mathcal{M}H^1_\infty$ satisfying the following equations:

$$G = NM^{-1} = \tilde{M}^{-1}\tilde{N} \tag{3.3}$$

$$\begin{bmatrix} \tilde{X} & -\tilde{Y} \\ -\tilde{N} & \tilde{M} \end{bmatrix} \begin{bmatrix} M & Y \\ N & X \end{bmatrix} = I \tag{3.4}$$

The above equations constitute a **doubly coprime factorization** of G.

Now define eight functions in $\mathcal{M}H^1_\infty$ by:

$$\begin{bmatrix} M & Y \\ N & X \end{bmatrix}(z) = \begin{bmatrix} I & 0 \\ D & I \end{bmatrix} + \begin{bmatrix} F \\ C_F \end{bmatrix}(zI - A_F)^{-1} \begin{bmatrix} B & -H \end{bmatrix} \tag{3.5}$$

$$\begin{bmatrix} \tilde{X} & -\tilde{Y} \\ -\tilde{N} & \tilde{M} \end{bmatrix}(z) = \begin{bmatrix} I & 0 \\ -D & I \end{bmatrix} + \begin{bmatrix} F \\ C \end{bmatrix}(zI - A_H)^{-1} \begin{bmatrix} -B_H & H \end{bmatrix} \tag{3.6}$$

where $A_F = A + BF$
$A_H = A + HC$
$C_F = C + DF$
$B_H = B + HD.$

As mentioned before, A_F and A_H are assumed to be stable matrices.

Lemma 3.3
The eight functions defined as in (3.5) and (3.6) satisfy (3.3) and (3.4) and therefore constitute a doubly-coprime factorization.

Although we will not prove this lemma, it can be easily shown that
$G = NM^{-1}$:
Define a vector $v_k := u_k - Fx_k$. Then from (3.1) and (3.2) we get:

$$\begin{aligned} x_{k+1} &= A_F x_k + B v_k \\ u_k &= F x_k + v_k \\ y_k &= C_F x_k + D v_k \end{aligned}$$

Evidently the transfer functions from v to y, resp. from v to u are given by:

$$\begin{aligned} N &= C_F(zI - A_F)^{-1}B + D \qquad & y &= Nv \\ M &= F(zI - A_F)^{-1}B + I \qquad & u &= Mv \end{aligned}$$

Therefore

$$y = NM^{-1}u \quad \Longleftrightarrow \quad G = NM^{-1}$$

Theorem 3.4 (Youla Parameterization)
Let G have the doubly-coprime factorization given by (3.5) and (3.6).
Assume that $det(X - NQ)$ and $det(\tilde{X} - Q\tilde{N})$ are not identically zero, which guarantees $(X - NQ)$ and $(\tilde{X} - Q\tilde{N})$ have well-defined inverses.

Then the set of all proper, real rational controllers K stabilizing G is given by

$$K = (Y - MQ)(X - NQ)^{-1} = (\tilde{X} - Q\tilde{N})^{-1}(\tilde{Y} - Q\tilde{M}) \tag{3.7}$$

where Q is an arbitrary parameter in $\mathcal{M}H^1_\infty$

For $Q = 0$, we simply have an observer-based controller. So basically this parameterization consists of an observer-based stabilizing controller $\tilde{K}$ and a free design parameter Q. Figure 3.1 represents the observer-based controller $\tilde{K}$.

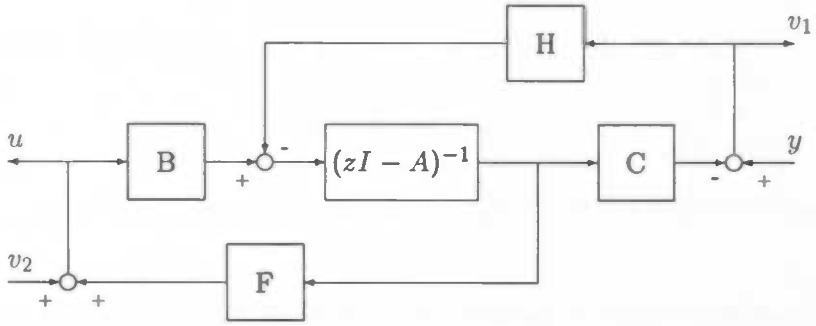

Figure 3.1 *The observer-based controller $\tilde{K}$.*

Now an arbitrary system $Q \in \mathcal{M}H^1_\infty$ is connected from the additional output v_1 to the additional input v_2, where Q is the Youla parameter.

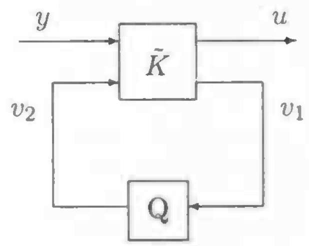

Figure 3.2

The block diagram in figure 3.2 shows the resulting stabilizing controller K obtained from the Youla parameterization.
Figure 3.3 represents the resulting closed loop system.

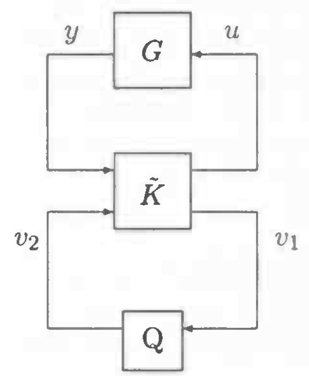

Figure 3.3 *Plant G with stabilizing controller.*

3.2 Closed Loop Maps

From now on we use a positive feedback convention as shown in figure 3.4.

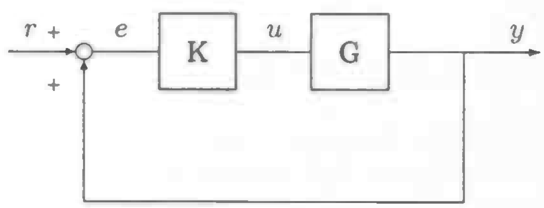

Figure 3.4

The beauty of the Youla parameterization lies in the fact that all admissible closed loop maps are affine in the parameter Q.
This can be verified by direct substitution, for example

$$\begin{aligned} S &= (I-GK)^{-1} = ((I-NM^{-1}(\tilde{X}-Q\tilde{N})^{-1}(\tilde{Y}-Q\tilde{M}))^{-1} \\ &= (I-(M\tilde{X}-MQ\tilde{N})^{-1}(N\tilde{Y}-NQ\tilde{M}))^{-1} \end{aligned}$$

$$= \quad ((M\tilde{X} - MQ\tilde{N})^{-1}(M\tilde{X} - MQ\tilde{N} - N\tilde{Y} + NQ\tilde{M}))^{-1} \tag{3.8}$$

From lemma 3.3 we have $M\tilde{X} - N\tilde{Y} = I$ and $MQ\tilde{N} = NQ\tilde{M}$.
So (3.8) reduces to

$$S = M(\tilde{X} - Q\tilde{N}) \tag{3.9}$$

Similarly, we can derive

$$T = N(\tilde{Y} - Q\tilde{M}) \tag{3.10}$$

We now substitute (3.9) and (3.10) in our original problem formulated in section 2.4

$$\min_{Q \in \mathcal{M}H^1_\infty} \max_{\omega \in \Omega} \{\max (\mathcal{W}_1|M(\tilde{X} - Q\tilde{N})|, \ \mathcal{W}_2|N(\tilde{Y} - Q\tilde{M})|)\} \tag{3.11}$$

The motivation for using the Youla parameterization is that our goal function is now affine in Q, and therefore the optimization problem is linear.
Another advantage is that to ensure K stabilizes G, we only need to choose $Q \in \mathcal{M}H^1_\infty$, so we are optimizing over a convex subspace.
In other words, this is a better posed optimization problem than the original problem formulated in section 2.4.

3.3 Design Parameter Q

As stated before the idea of the Youla parameterization is to stabilize the plant first by choosing appropriate matrices F and H.
A first attempt was to keep F and H fixed and to modify the stabilized plant by varying the parameter Q, where $Q \in \mathcal{M}H^1_\infty$.
Implementation of this principle at the minimax optimization problem turned out not to give satisfactory results. The reason why this did not work well is given below.

Lemma 3.5
Suppose G has order g and Q has order q.
Then

$$order(K) = g + q$$

Proof
This can be easily seen from $K = (Y - MQ)(X - NQ)^{-1}$

The converse of this lemma is *not* true:
If K has order k then the corresponding Q does not necessarily have order $k - g$.
Consequently, varying $Q \in \mathcal{M}H^1_\infty$, but keeping the order of Q fixed (as required by the minimax algorithm) does not capture all stabilizing controllers of order $g + q$.
So once we fix the order of K by fixing the order of Q, we limit the set of kth order stabilizing controllers.
This is illustrated by counting the tuning parameters of respectively K and Q.
Suppose G(z) is a first order plant and the question is to design a second order proper controller:

$$K = \frac{a_2z^2 + a_1z + a_0}{b_2z^2 + b_1z + 1}$$

As $g + q = k = 2$ we choose a first order proper Q:

$$Q = \frac{\alpha z + \beta}{\gamma z + 1}$$

In this case the number of tuning parameters n_k of the controller K is 5, while Q has only three tuning parameters ($n_q = 3$).
In general,

$$n_q = 2q + 1 \qquad n_k = 2k + 1 = 2(g + q) + 1$$

$$n_k - n_q = 2g$$

Therefore we may conclude that Q has too few optimization parameters and does not give you a full parameterization of all kth order stabilizing controllers.
Obviously this method fails to find the optimal controller, so we tried a second approach.

3.4 Design Parameters Q, F and H

To compensate the fact that, when varying Q alone, the number of optimization parameters is too few, F and H are also considered as optimization parameters.
F and H each consist of g parameters where g is the order of $G(z)$. The total number of parameters is now

$$n_q + 2g = n_k$$

The number of optimization parameters is now equal to n_k, the number of parameters when optimizing over K. So this could give us a full parameterization of all kth order stabilizing controllers.
Whether or not this is the case remains an open question. Our numerical results indicate that it does provide us with a sufficiently rich class of kth order controllers for our applications.
This concept will be implemented in MATLAB to perform the Minimax optimization.

Implementation

In this section an outline of the program[1] using *Minimax* from the MATLAB *Optimization Toolbox* is given.

Using Q, F and H as design parameters, The minimax problem is the following:

$$\min_{Q,F,H} \max_{\omega \in \Omega} \{\max\ (\ \mathcal{W}_1|M(\tilde{X} - Q\tilde{N})|,\ \mathcal{W}_2|N(\tilde{Y} - Q\tilde{M})|\ )\} \qquad (3.12)$$

where $Q \in \mathcal{M}H^1_\infty$ and F, H are such that respectively A_F and A_H are stable.

• The goal function

The **goal function** calculates the frequency responses of

$$(\mathcal{W}_1|M(\tilde{X} - Q\tilde{N})|, \mathcal{W}_2|N(\tilde{Y} - Q\tilde{M})|) \quad \text{for} \quad \omega \in \Omega$$

[1]See also Appendix B

for a given vector x where x contains the parameters of respectively F, H and Q in state space representation.
For example if G is a second order plant, we have $F = [f_1\ f_2]$ and $H = [h_1\ h_2]^T$.
Suppose that the algorithm is started with a second order Q:

$$Q = \left[\begin{array}{c|c} A_q & B_q \\ \hline C_q & D_q \end{array}\right] = \left[\begin{array}{cc|c} a_1 & a_2 & b_1 \\ a_3 & a_4 & b_2 \\ \hline c_1 & c_2 & d \end{array}\right] \tag{3.13}$$

This representation of Q is an overparameterization of Q represented as a transfer function. This can be seen by counting the number of parameters:
In state space representation this second order Q has nine optimization parameters, whereas in transfer function representation a second order Q has only five parameters. (See section 3.3)
Why we use this state space representation is explained below in the section parameter reduction.
The vector x is defined as $[f_1\ f_2\ h_1\ h_2\ a_1\ a_3\ c_1\ a_2\ a_4\ c_2\ b_1\ b_2\ d\ ]$.

Ω is a discrete frequency grid, which should be chosen dense enough to prevent S and T from having a peak value between two frequency points.

The goal function also computes the **constraint function**.
To guarantee stability of the closed loop system Q has to be in $\mathcal{M}H^1_\infty$, i.e., $|\lambda(A_q)| < 1$, where A_q is the A matrix of the parameter Q.
We also have (from the Youla parameterization)

$$|\lambda(A_F)| < 1 \qquad |\lambda(A_H)| < 1$$

To construct the necessary coprime factorizations the formulas given in (3.5) and (3.6) are implemented.
To get a reasonable starting estimate of F and H such that A_F and A_H are stable, the function *'dlqr'* from the *Control Toolbox* is used. This function performs linear-quadratic regulator design for discrete-time systems and calculates the optimal gain matrix F such that the state-feedback law $u_k = -Fx_k$ minimizes the cost function $J = \sum(x^TQx + u^TRu)$.

• The gradient function

Minimax uses line- and gradient search to find the optimum. In general, gradient methods use information about the slope of the function to determine the correct search direction. If no gradient function is provided, the calculation of these gradients is performed numerically by finite difference approximation.
To improve accurateness and efficiency an analytically computed gradient function can be supplied by the user.
This gradient function computes the gradients of the objective function with respect to the optimization parameters.
In this case, the gradient function calculates the partial derivatives of $\mathcal{W}_1|S|$ and $\mathcal{W}_2|T|$ with respect to the elements of F, H and Q.

For example

$$\frac{\partial|S|}{\partial Q_i} = \frac{\partial(\bar{S}S)^{\frac{1}{2}}}{\partial Q_i} = \frac{1}{2|S|}(\frac{\partial \bar{S}}{\partial Q_i}S + \frac{\partial S}{\partial Q_i}\bar{S})$$

$$= \frac{1}{2|S|}(\overline{\frac{\partial S}{\partial Q_i}\bar{S}} + \frac{\partial S}{\partial Q_i}\bar{S}) = \frac{1}{|S|}\Re(\frac{\partial S}{\partial Q_i}\bar{S})$$

And

$$\nabla S = \begin{bmatrix} \frac{\partial S}{\partial f_i} \\ \frac{\partial S}{\partial h_i} \\ \frac{\partial S}{\partial Q_i} \end{bmatrix} = \begin{bmatrix} \frac{\partial S}{\partial F}\frac{\partial F}{\partial f_i} \\ \frac{\partial S}{\partial H}\frac{\partial H}{\partial h_i} \\ \frac{\partial S}{\partial Q}\frac{\partial Q}{\partial Q_i} \end{bmatrix}$$

It also calculates the gradient of the constraint. Because the singular values of a matrix are continuous and eigenvalues are not, we use the following:

$$\lim_{p\to\infty} \sqrt[p]{\bar{\sigma}(A^p)} = \max|\lambda(A)|$$

where $\bar{\sigma}$ denotes the largest singular value.
So if $\sqrt[p]{\bar{\sigma}(A^p)} < 1$ for p large enough, A is stable.

• Parameter reduction

As described in the preceding section, the number of parameters of a qth order Q represented as a transfer function is equal to $2q+1$. The system is completely determined by these parameters.
Using MATLAB it is not recommended to use transfer function representations as they have poor numerical properties.
Therefore the vector x contains Q in *state space* representation. In state space representation, Q contains $(q+1)^2$ parameters. (see for instance (3.13) for $q=2$)
This means there are $(q+1)^2 - (2q+1) = q^2$ parameters too many when Q is given in state space form.

The idea is now to reduce the original set of parameters given by the vector x to a smaller, reduced set of parameters. The resulting reduced vector x_{red} contains all important search directions.
The non reduced vector x contains $(q+1)^2 + 2g$ parameters where g denotes the order of the plant G.
To reduce the set of parameters a singular value decomposition[2] of the gradient of the goal function is used:

$$\nabla f(x_0) = U\ S\ V^T = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} S_1 & 0 \\ 0 & S_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} \quad (3.14)$$

S is partitioned in two blocks S_1 and S_2 where S_1 has dimension $2q+1+2g$ and S_2 has dimension q^2.
From this singular value decomposition we can derive the principal directions of the gradient.

[2]See Appendix A

It turns out that S_2 is equal to zero.
This means that there are only $2q + 1 + 2g$ important directions, exactly the number of parameters which we expected from the transfer function representation. Varying x in the other directions has no influence on the value of the goal function.
Suppose we start the optimization routine with $x = x_0$.
Then from (3.14) we have

$$\nabla f(x_0) = U_1 S_1 V_1^T$$

Suppose now

$$x = x_0 + V_1 S_1^{-1} x_{red}$$

where x_{red} has dimension $2q + 1 + 2g$, then

$$f(x) = f(x_0 + V_1 S_1^{-1} x_{red}) \approx f(x_0) + \nabla f(x_0) V_1 S_1^{-1} x_{red} = f(x_0) + U_1 x_{red}$$

$f(x_0)$ is a constant vector and $f(x)$ is now a function of the reduced vector x_{red}. The optimization procedure is now performed for the reduced set of parameters given by x_{red}.

As the $2q + 1 + 2g$ directions we found depend on x_0, the singular value decomposition of the gradient is done again whenever $max|x_{opt} - x_{red}| > \beta$ where β is a bound and x_{opt} is the optimized x_{red} obtained from the minimax algorithm.
The bound β is determined by numerical experiment. If β is chosen too large, the singular value decomposition is no longer valid. On the other hand, if β is too small, the algorithm converges very slowly.
The transformation $V_1 S_1^{-1}$ is used to find the resulting (unreduced) vector x.

Chapter 4

Sub-optimal Design

In this chapter the application of the method described in section 3.4 is illustrated by solving the minimax optimization problem for a concrete system. By studying a design example all the aspects treated in the preceding chapters, such as loop shaping and performance are used to find a sub-optimal low order controller.

Finally the Youla approach is compared with the already existing K-optimization method at Signaal discussed in section 2.4.

4.1 A design example: The double integrator

The double integrator is described by the following differential equation:

$$\frac{\mathrm{d}^2 y}{\mathrm{d}t^2} = u \tag{4.1}$$

Physically, this equation represents Newtons second law:

$$F = ma$$

where F represents the force, a is the acceleration and m is a mass, which is set to one in equation (4.1).
Using the Laplace transform, (4.1) , with zero initial conditions ($\dot{y}(0) = y(0) = 0$), we obtain the transfer function:

$$\hat{G}(s) = \frac{\hat{y}(s)}{\hat{u}(s)} = \frac{1}{s^2}$$

If we define the state vector x as

$$x := \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} y \\ \dot{y} \end{bmatrix}$$

where y is the position and $\dot{y}$ is the velocity, (4.1) is equivalent to the following state space representation:

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \tag{4.2}$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} x \tag{4.3}$$

As we are considering discrete time systems, we use the z-transform to rewrite (4.3) and (4.4) as a discrete time system, with sampling time 1.

$$x_{k+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0.5 \\ 1 \end{bmatrix} u_k \tag{4.4}$$

$$y_k = \begin{bmatrix} 1 & 0 \end{bmatrix} x_k \tag{4.5}$$

As the eigenvalues of the A-matrix of this system are equal to 1, the system has two poles on the unity circle, which means the system is unstable.

4.1.1 Design of a third order controller

Suppose we have the following problem:

- *Find a sub-optimal, proper, third order controller which stabilizes the system described above, sampled at a sample frequency of 1000 Hz.*

Our first aim is to find appropriate initial values for the parameters Q, F and H which form an initial stabilizing *'controller'* K_0. After this is done the optimization algorithm described in section 3.4 is used to design a sub-optimal stabilizing controller.

We have performed many designs of a controller with the algorithm using different initial estimates. We first give the results obtained when using an (arbitrary) initial estimate of the parameters Q, F and H.

As the order of the controller is equal to the order of the plant(= 2) plus the order of Q, the optimization algorithm is started with a stable Q of order one.
We choose[1]

$$Q_0 = [0.1 \ 9 \ 8 \ -2].$$

Since the A-matrix of this Q_0 is equal to 0.1, and $|0.1| < 1$, Q_0 belongs to H^1_∞.
To determine F_0 and H_0 we have to choose values for r_1 and r_2. These variables are used by the function *'dlqr'* from the *Control Toolbox.*
For example if we choose $r_1 = 10000$, *'dlqr'* calculates the optimal gain matrix F_0 such that the state-feedback law $u_k = -F_0 x_k$ minimizes the cost function $J = \sum(x^T Q x + u^T r_1 u)$, where Q is set to 1.
This results in

$$F_0 = [-0.0093 \ \ -0.1368]\,, \qquad |\lambda(A + BF_0)| = \begin{bmatrix} 0.9316 \\ 0.9316 \end{bmatrix}$$

[1]for explanation see section 3.4 and Appendix B.

In a similar way, the variable r_2 determines H_0. Choosing $r_2 = 10000$ corresponds with

$$H_0 = \begin{bmatrix} -0.1415 \\ -0.0093 \end{bmatrix}, \qquad |\lambda(A + H_0C)| = \begin{bmatrix} 0.9315 \\ 0.9315 \end{bmatrix}$$

We now design the weighting functions.

As described in section 3.4, the design is done graphically. Of course there are many possible designs, depending on the choice of the specification bounds. In this example the following specifications are used:
For good tracking performance we choose a closed loop bandwidth ω_β of 20 Hz. Up to this frequency the system will respond well to a reference signal.
Above this frequency ω_β we let the weight for T roll off at 2 decade/decade to reduce the influence of measurement noise and to guarantee robust stability.

The algorithm converges, and a sub-optimal third order, stable controller is found:

$$K_1 = \frac{-0.03963z^3 + 0.07451z^2 - 0.03173z - 0.003173}{z^3 - 0.9847z^2 - 0.5546z + 0.5429} \tag{4.6}$$

The corresponding complementary sensitivity function is:

$$T_1 = \frac{0.01981z^4 - 0.01744z^3 - 0.02139z^2 + 0.01745z + 0.001587}{z^5 - 2.965z^4 + 2.397z^3 + 0.646z^2 - 1.623z + 0.5445} \tag{4.7}$$

The resulting sub-optimal controller K_1 is now substituted in the optimization criterion $\max\{\mathcal{W}_1|S|, \mathcal{W}_2|T|\}$. We define the corresponding μ**-value** as the maximum of this optimization criterion evaluated with the obtained sub-optimal controller on the grid Ω.
In this case we have, for K_1:

$$\mu_1 := \max_{\omega\in\Omega}\{\max(\mathcal{W}_1|S_1|, \mathcal{W}_2|T_1|)\} = 0.9220 \tag{4.8}$$

Furthermore, as the value of μ_1 is smaller than one, we have:

$$|S_1| < \mathcal{W}_1, \qquad |T_1| < \mathcal{W}_2,$$

so the design specifications are met.

The peak value of the sensitivity function $|S_1|$ is equal to 1.2821. The peak value of the complementary sensitivity function $|T_1|$ is 1.3069, which represents the maximum gain of a signal.
In figure 4.1 the weighting functions and the optimized S and T are shown.

In this figure, the straight lines are the weighting functions.
The sensitivity function S is the curved line, increasing to one. The complementary sensitivity T is the curved line which starts at one for low frequencies and rolls off for frequencies $\omega \geq \omega_\beta = 20$.

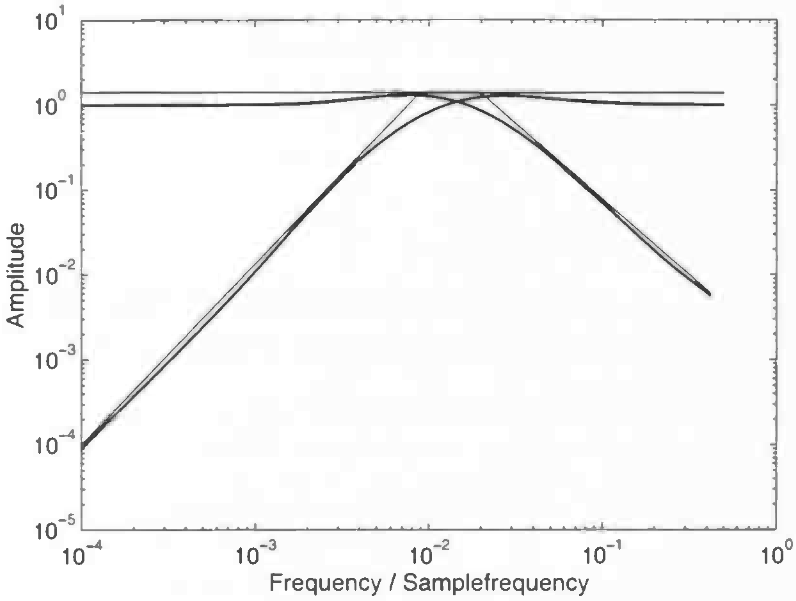

Figure 4.1: *S and T after optimization.*

The step response of the optimized closed loop system is drawn in figure 4.2.
The overshoot is small and the settling time is only 0.1 seconds.(i.e., the output of the system settles down to its steady state value 1 in 0.1 sec.)

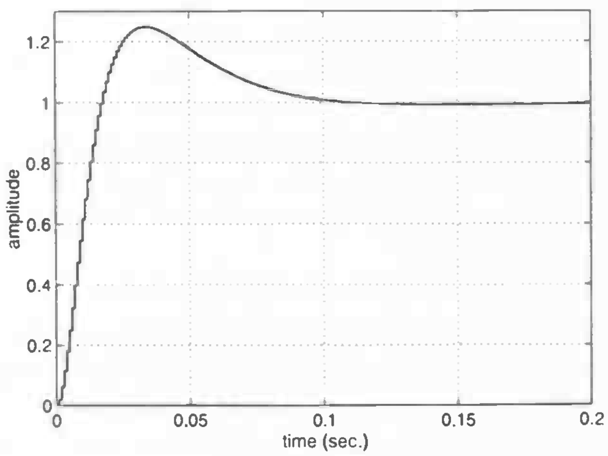

Figure 4.2: *Step response of the optimal system, sampled at 1000 Hz.*

The value of μ_1 given above is in fact the smallest value for μ we found using the algorithm for this design example.

When we start with different initial estimates the algorithm may converge to different, local optima. This illustrated by table 4.1. This table gives the obtained results with respect to μ when the algorithm was started with six different initial estimates for Q_0, r_1 and r_2.

Q_0	r_1	r_2	μ
[0.1 9 8 −2]	10000	10000	0.9220
[−0.9 9 8 −2]	1000	1000	0.9223
[0.1 9 8 −2]	1000	1000	0.9788
[0.5 5 10 6]	1000	1000	0.9222
[0.5 5 10 6]	1000	10000	0.9304
[0.8 3 4 5]	10000	10000	0.9222

Table 4.1

We see that, except for one case, the variations in μ are small.
Although these variations are small, due to the non-convexity of this optimization problem, the possibility of converging to a local optimum always exists.
It is therefore recommendable to perform more than one design using different initial estimates.

4.1.2 Designing higher order controllers

After comparing different third order controllers, higher order controllers are compared with the third order controller K_1 found in (4.6).
Increasing the order k of the controller by increasing the order q of the Q-parameter results in slightly smaller values for μ:

K	k	q	μ	$\max(\|S\|)$	$\max(\|T\|)$
K_1	3	1	0.9220	1.2821	1.3069
K_2	4	2	0.9184	1.2768	1.3015
K_3	5	3	0.9180	1.2761	1.3008

Table 4.2
k , q : *order of resp.* K *and* Q.
μ : *μ-value after Youla-optimization.*

The resulting fourth (K_2) and fifth (K_3) order sub-optimal controllers are:

$$K_2 = \frac{-0.04411z^4 + 0.07856z^3 - 0.02225z^2 - 0.01651z + 0.004285}{z^4 - 0.7393z^3 - 0.9968z^2 + 0.7384z + 0.001201}$$

$$K_3 = \frac{-0.04389z^5 + 0.1201z^4 - 0.1025z^3 + 0.01526z^2 + 0.01583z - 0.004763}{z^5 - 1.703z^4 - 0.1452z^3 + 1.594z^2 - 0.8414z + 0.09616}$$

K_2 was found by using the initial estimate: $Q_0 = [0.1\ 0\ 2\ 3\ 0.2\ 3\ 4\ 5\ 6]$ and $r_1 = r_2 = 100000$. For K_3 we used $Q_0 = [-0.8\ 0\ 0\ 5\ 9\ 0.1\ 0\ 6\ 9\ 7\ 0.9\ 9\ 4\ 3\ 2\ 1]$, $r_1 = r_2 = 1000$.
Table 4.2 only gives the best results for μ. As described above, different initial values for Q_0, r_1 and r_2 may converge to different optima.

4.2 Results: Youla compared with K-optimization

In this section the results obtained in table 4.2 are compared with the results obtained when using minimax optimization for the original problem as described in section 2.4, the K-optimization method used at Signaal.

As both algorithms may find local optima and in some case do not even converge, the algorithms were tested with several starting estimates.
In the original K-optimization problem, used at Signaal, the algorithm was tested with several estimates for K_0. The starting estimates Q_0, F_0 and H_0 of the Youla optimization method were also varied to find a sub-optimal controller.
The best results, with respect to the value for μ are given in table 4.3.

k	q	μ_{youla}	μ_k
3	1	0.9220	0.9303
4	2	0.9184	0.9222
5	3	0.9180	0.9222

Table 4.3
k , q : *order of resp. K and Q.*
μ_k , μ_{youla} : *μ-values after resp. K and Youla-optimization.*

Table 4.3 shows that the values for μ using the Youla parameterization are slightly smaller than the values for μ_k, which means the sub-optimal controllers we obtained from the Youla optimization are slightly better then the sub-optimal controllers resulting from K-optimization. The variations in μ_{youla} are small, for $k = 4$ and $k = 5$ we always have $\mu_{youla} < 0.93$.

One of the advantages of the method using Youla is the following fact:
The algorithm appears to be less sensitive to variations in the initial estimate. Even if the initial values of the sensitivity function S_0 and complementary sensitivity function T_0 are far above the the desired $\mathcal{W}_1^{-1}$ and $\mathcal{W}_2^{-1}$ (i.e., μ is large), the algorithm does converge quickly to the optimal controller.

The algorithm used at Signaal requires a good starting guess for the initial estimate K_0. This K_0 is designed by defining the poles and zeros of S_0 and T_0.
If μ_k is large for K_0, the algorithm converges slowly and the optimum found is often a local optimum.
This is illustrated by the following example.

First an arbitrary *stable* (required for K-optimization) initial K_0 was designed. Then both algorithms were started with this K_0.
Table 4.4 shows the results.

μ_0	μ_k	N_k	μ_{youla}	N_{youla}
7.3475	0.9842	90	0.9222	54
5.7194	0.9841	154	0.9222	31

Table 4.4

μ_0 : *initial value of* μ.

μ_k , μ_{youla} : μ*-values after resp. K and Youla-optimization.*

N_k, N_{youla} : *Number of iterations used in the optimization procedure for resp. K and Youla-optimization.*

Obviously the optimum μ_k is a local optimum, as the optimum μ_{youla} is smaller than μ_k, whereas the number of iterations N_k to find this local optimum is larger than N_{youla}.

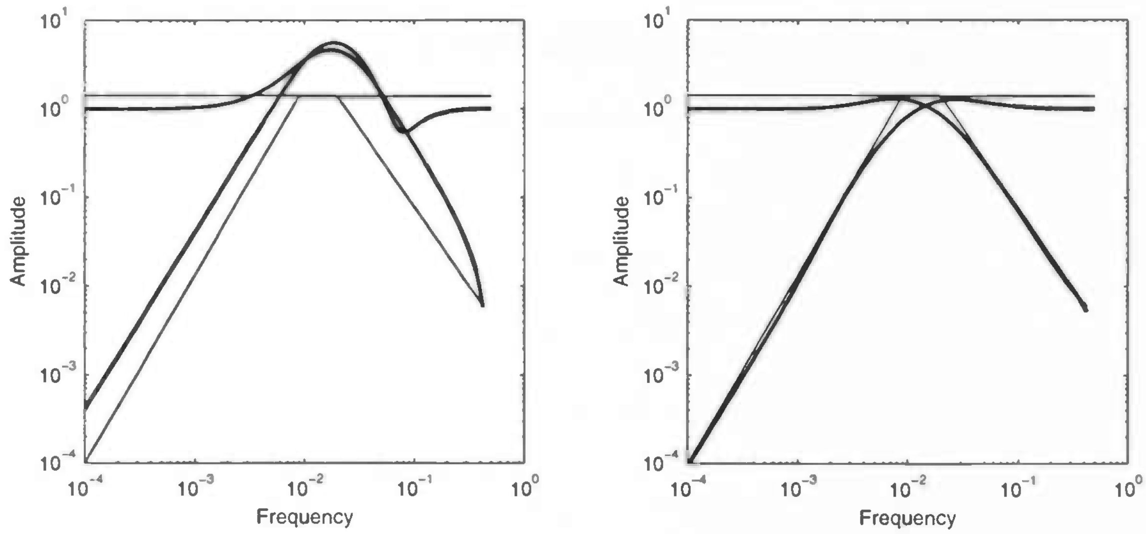

Figure 4.3: *S and T before and after optimization.*

In figure 4.3 S_0 and T_0 *(left)* and the resulting S_{opt} and T_{opt} using Youla, *(right)* are shown. Figure 4.4 demonstrates the difference between the weighted S and T obtained from respectively K- and Youla-optimization.

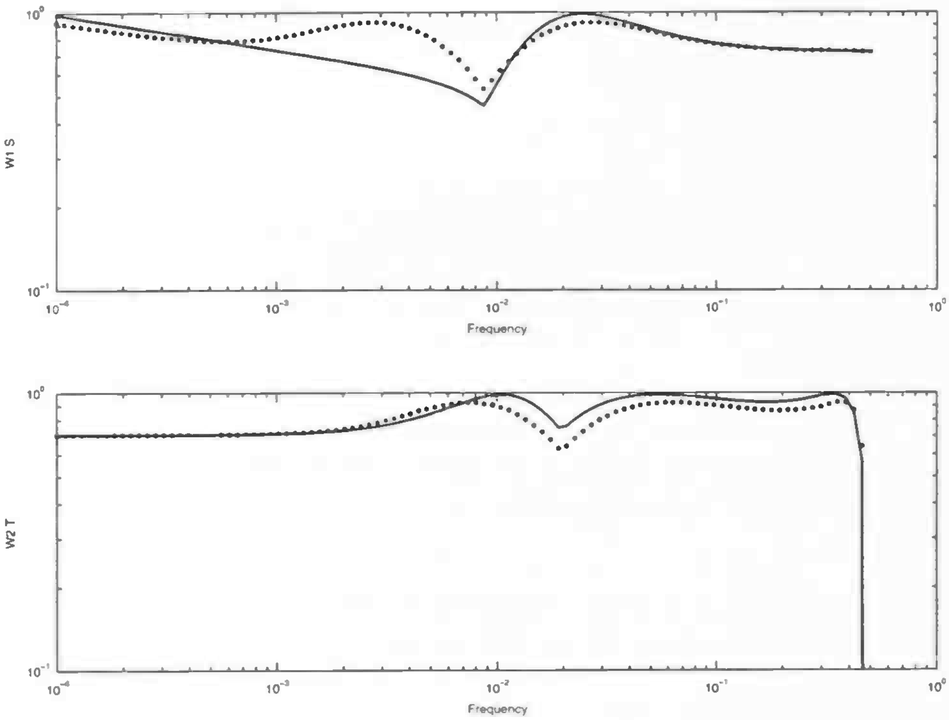

Figure 4.4: *Weighted S_k, T_k (solid) and S_{youla}, T_{youla} (dotted)*

4.3 Conclusions

Although the direct low-order design by the minimax optimization method evidently has some important advantages, the resulting K-optimization problem is a non-convex, complex problem.

By using the Youla parameterization, the problem was rewritten as a better posed optimization problem.
Considering Q as the only optimization parameter did result in a convex optimization problem. Unfortunately, fixing the order of Q, reduced the set of stabilizing controllers and did not provide us with a full parameterization of all stabilizing controllers of a given order.
We therefore introduced the feedback matrices F and H as additional optimization parameters.
Whether or not this actually gives us a full parameterization of all stabilizing controllers of a fixed order k would be an interesting issue for further research.
The numerical results do in fact indicate that the obtained class of kth order stabilizing controllers is sufficiently rich for our purposes.

Compared with the original K-optimization method, the Youla optimization method results in slightly smaller (i.e. better) values for the optimization criterion

$$\max_{\omega \in \Gamma}\{\max(\mathcal{W}_1|S|, \mathcal{W}_2|T|)\}$$

Another advantage is that the algorithm seems to be less sensitive for variations in the initial estimate of a stabilizing controller.

Appendix A

Singular Value Decomposition

A useful tool in matrix analysis is **singular value decomposition**. The following theorem states that every matrix A in $\mathbb{R}^{m\times n}$ has a singular value decomposition.

Theorem
Let $A \in \mathbb{R}^{m\times n}$. There exist unitary[1] matrices

$$U = [u_1, u_2, \ldots.., u_m] \in \mathbb{R}^{m\times m}$$

$$V = [v_1, v_2, \ldots.., v_m] \in \mathbb{R}^{n\times n}$$

such that

$$A = U\Sigma V^T, \quad \Sigma = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix}$$

where

$$\Sigma_1 = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_p \end{bmatrix}$$

and

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0, \quad p = \min\{m, n\}$$

Proof
See for example [5]

The largest singular value of A is often denoted as

$$\bar{\sigma}(A) = \sigma_{max} = \sigma_1$$

From a geometrical point of view, the singular values of a matrix A are precisely the lengths of the semi-axes of the hyperellipsoid $\mathcal{E}$ defined by $\mathcal{E} = \{y : y = Ax,\ x \in \mathbb{C}^n,\ \|x\| = 1\}$. This means v_1 is the direction in which $\|y\|$ is largest for all $\|x\|$. Conversely, v_n is the direction in which $\|y\|$ is smallest for all $\|x\|$.

This principle is applied in the parameter reduction method (section 3.4) to find the principal directions of the gradient of the goal function.

[1] $U^T U = I = UU^T$

Appendix B

Program Description

In section 3.4 a description of the implementation of the minimax method has been given. This method is implemented in MATLAB. The optimization procedure is performed in the program `Optimize`. The structure and subroutines of `Optimize` are discussed in this appendix.

OPTIMIZE

Given a system `[A,B,C,D]`, a starting estimate for the parameters `Q`,`F` and `H` and weighting functions `w1` and `w2` on a discrete frequency grid `z`, this function calculates a sub-optimal low-order controller for the mixed sensitivity problem. The main routine used in this program is `minimax` from the MATLAB *Optimization Toolbox*.

MINIMAX

The MATLAB function `minimax` is a numerical search algorithm which uses `goalfun_red` and `gradfun_red` to calculate the optimal vector `FHQ_opt` which contains `F_opt`, `H_opt` and `Q_opt`.
Syntax: `FHQ_opt = minimax('goalfun_red', FHQ_red, 'gradfun_red')`
The vector `FHQ_red` contains the reduced parameter set for `F`, `H` and `Q`.

GOALFUN_RED

Goal function for `minimax` optimization which calculates frequency response of the weighted sensitivity function S and complementary sensitivity function T for the reduced parameter set `FHQ_red` on a given discrete frequency grid `z`, where

$$S = M(\tilde{X} - \tilde{N}Q), \qquad T = N(\tilde{Y} - Q\tilde{M})$$

This function calls `coprimefactor` to calculate the necessary coprime factors.

GRADFUN_RED

The function `gradfun_red` computes the derivatives of the weighted sensitivity function S

and the weighted complementary sensitivity function T with respect to the reduced set of optimization parameters FHQ_red.

COPRIMEFACTOR

This function calculates the frequency responses of a doubly coprime factorization for a given discrete-time system $G = (A, B, C, D)$ and matrices F and H such that

$$G(z) = N(z)M(z)^{-1} = \tilde{M}(z)^{-1}\tilde{N}(z)$$

and

$$\begin{bmatrix} \tilde{X} & -\tilde{Y} \\ -\tilde{N} & \tilde{M} \end{bmatrix} \begin{bmatrix} M & Y \\ N & X \end{bmatrix} = I.$$

Coprimefactor uses the functions described in equations (3.5) and (3.6).

TRANSFORMATION

The function transformation computes the transformation matrix trans $= V_1 S_1^{-1}$ needed for the parameter reduction method as described in section 3.4.
This transformation matrix is used to calculate the reduced vector FHQ_red from the original full parameter set FHQ before the optimization procedure is started. Minimax then computes the *reduced* optimal vector FHQ_opt which is then transformed back to the non-reduced optimal vector.
This function calls gradfunction which computes derivatives of the weighted sensitivity function S and the weighted complementary sensitivity function T with respect to the full set of optimization parameters FHQ.

MAKECONTR

Makecontr calculates the transfer function of the optimal controller contr_opt from the parameters F_opt, H_opt and Q_opt resulting from the minimax optimization.
To do this we use

$$K = (Y - MQ)(X - NQ)^{-1}$$

(See theorem 3.4: The Youla parameterization)
It also computes the resulting closed loop system sysclosed_opt (T), and the sensitivity function S_opt.

GRAPH

Given a system $G = (A, B, C, D)$ and a vector FHQ containing F, H and Q, this function plots the graphs of S and T together with the weighting functions w1 and w2 on a log-log scale.
The function coprimefactor is used to calculate the necessary coprime factors.

Bibliography

[1] J. C. Doyle, B. A. Francis, and A. R. Tannenbaum. *Feedback Control Theory.* Macmillan, 1992.

[2] G.F. Franklin, J.D.Powell, and M.L.Workman. *Digital Control of Dynamic Systems.* Addison Wesley, 1980.

[3] B.A.Francis. *Lecture Notes in Control and Information Sciences, a Course in H_∞ Control Theory.* Springer-Verlag, 1987.

[4] J.M. Maciejowski. *Multivariable Feedback Design.* Addison-Wesley, 1989.

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

[6] M. A. Dahleh and I.J. Diaz-Bobillo. *Control of uncertain systems. A linear programming approach.* Prentice-Hall, Englewood Cliffs, NJ, 1995.