# $H_\infty$-Control in a Behavioral Context

*An example*

**Paula Beukers**



**Vakgroep Wiskunde**

RuG

Master thesis

# $H_\infty$-Control in a Behavioral Context

*An example*

**Paula Beukers**

Rijksuniversiteit Groningen
Vakgroep Wiskunde
Postbus 800
9700 AV Groningen

Augustus 1996

**Abstract**

Recently it has been argued that in many cases it is more natural to view the problem of controller design as finding for a given system an additional set of 'laws' which the signals appearing in the system should obey.

In recent work the $H_\infty$-control problem has been reformulated in this behavioral framework. A complete solution of the full information version of the $H_\infty$ problem has been obtained, together with algorithms to check whether such a controller exists and how to compute it.

The purpose of this thesis is to apply these algorithms to an example, the computation of the optimal suspension of a vehicle that drives on a bumpy road. The robust controller algorithms and the simulation of the controlled behavior have been carried out as numerical algorithms in MATLAB.

# Contents

# 1 Introduction

In the standard $H_\infty$-control problem the aim is to design a feedback loop around a given system in such a way that in the closed loop system the influence of the exogenous inputs on the exogenous outputs remains within a certain a priori given tolerance. The system ($\Sigma$) under consideration has control inputs ($u$), exogenous inputs ($d$), measured outputs ($y$) and exogenous outputs ($z$). The controller ($K$) to be designed should take the measured outputs of the system as its inputs and should, on the basis of these inputs, generate control inputs for the system. These controllers should be designed in such a way that the resulting closed loop operator (mapping exogenous inputs to exogenous outputs, see figure 1.1) has norm less than or equal to some a priori given upper bound.

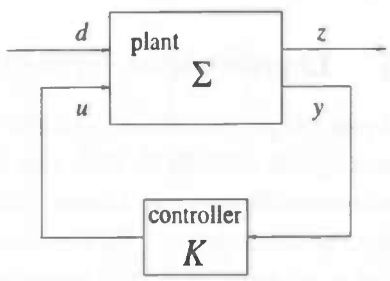


Figure 1.1

Often, it is more natural to view the problem of controller design as the problem of finding for a given system an additional set of 'laws' that the signals appearing in the system should obey. More specifically, if a system is given in terms of a certain set of 'behavioral equations' (such as $R(\frac{d}{dt})w = 0$, $w = M(\frac{d}{dt})\ell$, $R(\frac{d}{dt})w = M(\frac{d}{dt})\ell)$), then the problem of controller design is to invent an additional set of equations (such as $K(\frac{d}{dt})\ell = 0$, $c = C(\frac{d}{dt})\ell$), involving the signals appearing in the system, in such a way that the 'controlled system' (i.e. the system consisting of those signals that are compatible with both sets of equations) satisfies the a priori given control specifications.

In recent work the $H_\infty$-control problem has been reformulated in this behavioral framework. A complete solution of the full information version of the $H_\infty$ problem has been obtained, together with algorithms to check whether such a controller exists and how to compute it. The purpose of this paper is to apply these algorithms to an example, The robust controller algorithms and the simulation of the controlled behavior will be carried out as numerical algorithms in MATLAB.

We will first give a short introduction in the behavioral approach to dynamical systems and control (section 2). In this approach two-variable polynomials and linear quadratic forms play an important role. We will go into this matter in section 3. In section 4 we will explain about $H_\infty$-control in the behavioral framework. We will apply the theory to an example, the computation of the optimal suspension of a vehicle that drives on a bumpy road, in section 5. Finally, we will formulate some conclusions (section 6). In the appendices, some theory on $J$-spectral factorization (since the computation of the controller is based on this factorization), a few more graphs of the simulations, and the MATHEMATICA and MATLAB programs used for the computation and simulations can be found.

# 2 The Behavioral Approach to Dynamical Systems and Control

In this section we will first introduce some notation and basic facts from the behavioral approach to linear dynamical systems. Next we will state our view of control in this context.

## 2.1 Dynamical Systems in a Behavioral View

A dynamical system is a triple $\Sigma = (T, W, \mathfrak{B})$, with $T \subset \mathbb{R}$ the time axis, $W$ the signal space and $\mathfrak{B} \subset W^T$ the behavior. In this thesis we will be concerned with continuous-time real linear time-invariant differential dynamical systems only. Thus the time axis is $\mathbb{R}$, the signal space is $\mathbb{R}^q$ and the behavior $\mathfrak{B}$ is the solution set of a system of linear constant coefficient differential equations

$$R(\frac{d}{dt})w = 0 \tag{2.1}$$

in the real variables $w = \mathrm{col}[w_1, w_2, \ldots, w_q]$, the *manifest variables* of the system. $R$ is a real polynomial matrix, $R \in \mathbb{R}^{\bullet \times q}[\xi]$. For the behavior, i.e. for the solution set of (2.1) it is usually advisable to consider locally integrable $w$'s as candidate solutions, and to interpret the differential equation in the sense of distributions. However, to avoid mathematical technicalities, we will assume that the solution set consists of infinitely differentiable functions. Hence the behavior of (2.1) is defined as

$$\mathfrak{B} = \{w \in \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}^q) \mid R(\frac{d}{dt})w = 0\} \tag{2.2}$$

We will denote the family of dynamical systems obtained this way by $\mathfrak{L}^q$. Hence elements of $\mathfrak{L}^q$ are dynamical systems $\Sigma = (\mathbb{R}, \mathbb{R}^q, \mathfrak{B})$. Note that instead of writing $\Sigma \in \mathfrak{L}^q$ we may as well write $\mathfrak{B} \in \mathfrak{L}^q$. Often we will refer to the system $\Sigma = (T, W, \mathfrak{B})$ simply as the system $\mathfrak{B}$.

Each $R \in \mathbb{R}^{\bullet \times q}[\xi]$ defines a system $\mathfrak{B} \in \mathfrak{L}^q$, but this $R$ is not unique, there are always many more defining the same $\mathfrak{B}$. For instance, if $U$ is an unimodular polynomial matrix (e.g. $\det(U[\xi]) \in \mathbb{R} \setminus \{0\}$) such that the product $UR$ makes sense, then $R$ and $UR$ define the same element of $\mathfrak{L}^q$. Also, a system $\mathfrak{B} \in \mathfrak{L}^q$ can be represented in different ways. We will call the representation (2.1) a *kernel representation*, since it describes $\mathfrak{B}$ as $\mathfrak{B} = \ker(R(\frac{d}{dt}))$. Another way of describing $\mathfrak{B} \in \mathfrak{L}^q$ is such that $\mathfrak{B} = \mathrm{im}(M(\frac{d}{dt}))$, with the resulting representation:

$$w = M(\frac{d}{dt})\ell \tag{2.3}$$

this is called an *image representation.* The variable $\ell$ is called the *latent variable* of the system. A system only admits an image representation if it is *controllable.*

A system $\mathfrak{B} \in \mathfrak{L}^q$ is said to be *controllable* if for each $w_1, w_2 \in \mathfrak{B}$ there exists a trajectory $w \in \mathfrak{B}$ and a $t' \geq 0$ such that $w(t) = w_1(t)$ for $t < 0$ and $w(t) = w_2(t - t')$ for $t \geq t'$ (see figure 2.1). This is also how controllability in the classical approach is defined but in the behavioral approach it seems much more natural. It can be shown that $\mathfrak{B}$ is controllable if and only if its kernel representation satisfies $\text{rank}(R(\lambda)) = \text{rank}(R)$ for all $\lambda \in \mathbb{C}$, i.e. the complex matrix $R(\lambda)$ has *constant* rank for all $\lambda$.

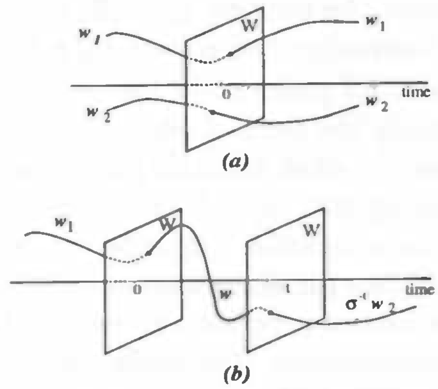


Figure 2.1

An image representation $w = M(\frac{d}{dt})\ell$ is called *observable* if $\ell$ is completely determined by $w$, i.e. if $M(\frac{d}{dt})\ell_1 = M(\frac{d}{dt})\ell_2$ implies $\ell_1 = \ell_2$. It can be shown that this notion of observability is equivalent to the condition that the complex matrix $M(\lambda)$ has *full column rank* for all $\lambda \in \mathbb{C}$. As it turns out, a controllable system always allows an observable image representation. Note that controllability is the property of a system and observability the property of a representation.

There is a third way of representing the system $\mathfrak{B}$, and that is by the so called *latent variable representation*:

$$R(\frac{d}{dt})w = M(\frac{d}{dt})\ell \tag{2.4}$$

A latent variable representation $R(\frac{d}{dt})w = M(\frac{d}{dt})\ell$ is said to be observable if $R(\frac{d}{dt})w = M(\frac{d}{dt})\ell_1$ and $R(\frac{d}{dt})w = M(\frac{d}{dt})\ell_2$ implies $\ell_1 = \ell_2$, or equivalently, the complex matrix $M(\lambda)$ has full column rank for all $\lambda \in \mathbb{C}$.

## 2.2 Control in a Behavioral View

We have just defined controllability in the behavioral context, and now we will briefly recall the view of control in the context of the behavioral approach to dynamical systems.

Let $\Sigma_1 = (T, W_1 \times C, \mathfrak{B}_1)$ and $\Sigma_2 = (T, W_2 \times C, \mathfrak{B}_2)$ be two dynamical systems with the same time axis. We hence assume that the signal spaces are Cartesian products with the factor $C$ in common. Trajectories of $\Sigma_1$ will be denoted by $(w_1, c)$ and those of $\Sigma_2$ by $(w_2, c)$. We'll define the *interconnection* of $\Sigma_1$ with $\Sigma_2$ as the dynamical system $\Sigma_1 \wedge \Sigma_2 := (T, W_1 \times W_2, \mathfrak{B})$, with $\mathfrak{B} = \{(w_1, w_2) : T \to W_1 \times W_2 |$ there exists $c$ such that$(w_1, c) \in \mathfrak{B}_1$ and $(w_2, c) \in \mathfrak{B}_2\}$. The interconnection takes place through the variable $c$, which is called the *interconnection variable.*

Assume that the *plant*, a dynamical system $\Sigma_p = (\mathbb{R}, W_1 \times C, \mathfrak{B}_p)$ is given. The second factor in the signal space denotes the space in which the interconnection variable $c$ takes its value, it is called the *interconnection* (or *control) space*. Consider now a family $\mathfrak{C}$ of dynamical systems, all with the signal space $C$ in common. An element $\Sigma_c = (\mathbb{R}, C, \mathfrak{B}_c) \in \mathfrak{C}$ will be called an *admissible controller*. The interconnected system $\Sigma_p \wedge \Sigma_c$ is called the *controlled system* (figure 2.2). The control problem for the plant $\Sigma_p$ is now to specify the set $\mathfrak{C}$ of admissible controllers, to describe what desirable properties the controlled system should have, and finally, to find an admissible controller $\Sigma_c$ such that $\Sigma_p \wedge \Sigma_c$ has the desired properties. Typically these control specifications require that certain components (the to-be-controlled variables) of the system's manifest variables need to be small as a function of the values of certain other components (the disturbances). In addition, the controlled system should be stable, in the sense that if the disturbances are zero, then the to-be-controlled variables should converge to the desired properties as time runs off to infinity. Let us be more specific. Assume that the manifest variable $w$ of the plant $\Sigma_p$ consist of three components, $w = (z, d, c)$. Here, $z$ is the to-be-controlled variable, $d$ the disturbance and $c$ the interconnection variable. Likewise, the signal space of $\Sigma_p$ is equal to the Cartesian product $Z \times D \times C$, with $Z, D$ and $C$ sets in which $z, d$ and $c$ take their values, resp. It is assumed that the disturbance is an unknown, externally imposed signal. This can be modeled by assuming that *any* function $d : \mathbb{R} \to D$ can occur, e.g., we assume that the variable $d$ is *free*. If any $\mathfrak{C}^\infty$ function can occur as the second component of the ($\mathfrak{C}^\infty$) manifest variable, then we call $d$ $\mathfrak{C}^\infty$ *free*. Also, we will require that in the controlled system $d$ is still free. If any $d$ is possible as the second component of the manifest variable $(z, d)$ of the controlled system $\Sigma_p \wedge \Sigma_c$ then we call the controller $\Sigma_c$ *admissible*.

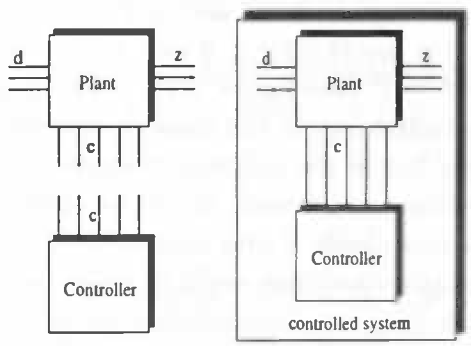

Figure 2.2

## 2.3 Dual Systems

In the existence question of stabilizing $H_\infty$-controllers an important role is played by the dual system. Suppose the controllable system $\Sigma = (\mathbb{R}, \mathbb{R}^q, \mathfrak{B})$ is given in the kernel representation

$$R(\frac{d}{dt})w = 0$$

where $R \in \mathbb{R}^{\bar{l} \times q}[\xi]$ . And let $M \in \mathbb{R}^{q \times l}[\xi]$ be such that

$$w = M(\frac{d}{dt})\ell$$

is an observable image representation of $\Sigma$. We define the *dual* of $\Sigma$ to be the system defined as $\Sigma^{\perp} := (\mathbb{R}, \mathbb{R}^q, \mathfrak{B}^{\perp}) \in \mathcal{L}^q$ with controllable kernel representation

$$M^T(-\frac{d}{dt})\bar{w} = 0$$

or observable image representation

$$\bar{w} = R^T(-\frac{d}{dt})\bar{\ell}$$

with latent variable $\bar{\ell}$. Thus the signal space of $\Sigma^{\perp}$ is equal to the signal space $\mathbb{R}^q$ of $\Sigma$.

The notation $\mathfrak{B}^{\perp}$ is used because the trajectories of $\mathfrak{B}^{\perp}$ are, in an appropriate sense, orthogonal to those of $\mathfrak{B}$, $\langle w, \bar{w} \rangle = 0$ for all $w \in \mathfrak{B} \cap \mathfrak{D}(\mathbb{R}, \mathbb{R}^q)$ and $\bar{w} \in \mathfrak{B}^{\perp} \cap \mathfrak{D}(\mathbb{R}, \mathbb{R}^q)$. More details on duality can be found in [6].

# 3 Quadratic Differential Forms

Differential equations and one-variable polynomial matrices are most suitable for describing the dynamics of linear time-invariant systems. In control in the behavioral context two-variable polynomials and linear quadratic differential forms play a similar important role.

Let $\mathbb{R}^{q_1 \times q_2}[\zeta, \eta]$ denote the set of real polynomial matrices in the (commuting) indeterminates $\zeta$ and $\eta$. An element $\Phi \in \mathbb{R}^{q_1 \times q_2}[\zeta, \eta]$ is thus given by

$$\Phi(\zeta, \eta) = \sum_{h,k=0}^{N} \Phi_{h,k} \zeta^h \eta^k \tag{3.1}$$

where $\Phi_{h,k} \in \mathbb{R}^{q_1 \times q_2}$ and $N \geq 0$ is an integer. We can associate with such a $\Phi$ a *bilinear differential form* $L_\Phi : \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}_1^q) \times \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}_2^q) \to \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R})$ defined by

$$(L_\Phi(\ell_1, \ell_2))(t) := \sum_{h,k=0}^{N} \left( \frac{d^h}{dt^h} \ell_1(t) \right)^T \Phi_{h,k} \left( \frac{d^k}{dt^k} \ell_2(t) \right)$$

If $q_1 = q_2$ $(=: q)$ then $\Phi$ induces a *quadratic differential form* $Q_\Phi : \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}^q) \to \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R})$ defined by

$$Q_\Phi(\ell) := L_\Phi(\ell, \ell)$$

The two-variable polynomial matrix (3.1) is called *symmetric* if $\Phi(\zeta, \eta) = \Phi^T(\eta, \zeta)$ $(=: \Phi^*(\zeta, \eta))$ or equivalently if $\Phi_{h,k} = \Phi_{k,h}^T$ for all $h, k$. The symmetric elements of $\mathbb{R}^{q \times q}[\zeta, \eta]$ will be denoted by $\mathbb{R}_s^{q \times q}[\zeta, \eta]$. Clearly, $Q_\Phi = Q_\Phi^* = Q_{\frac{1}{2}(\Phi + \Phi^*)}$. This shows that when we consider quadratic differential forms we can in principle restrict our attention to $\Phi$'s in $\mathbb{R}_s^{q \times q}[\zeta, \eta]$. The properties of the two-variable polynomial matrix (3.1) are completely determined by the real constant matrix

$$\tilde{\Phi} := \begin{bmatrix} \Phi_{0,0} & \Phi_{0,1} & \cdots & \Phi_{0,N} \\ \Phi_{1,0} & \Phi_{1,1} & \cdots & \Phi_{1,N} \\ \vdots & \vdots & & \vdots \\ \Phi_{N,0} & \Phi_{N,1} & \cdots & \Phi_{N,N} \end{bmatrix}$$

This matrix will be called the *coefficient matrix* associated with $\Phi(\zeta, \eta)$. We can think of $\Phi(\zeta, \eta)$ as the matrix product

$$\Phi(\zeta, \eta) = \begin{pmatrix} I & \zeta I & \ldots & \zeta^N I \end{pmatrix} \tilde{\Phi} \begin{pmatrix} I \\ \eta I \\ \vdots \\ \eta^N I \end{pmatrix}$$

Here, $I$ is the $q \times q$-identity matrix. Note that $\Phi(\zeta, \eta)$ is symmetric if and only if its coefficient matrix is a symmetric matrix.

The quadratic differential form $Q_\Phi$ is called *non-negative* if $Q_\Phi(\ell) \geq 0$ in the sense that $Q_\Phi(\ell)(t) \geq 0$ for all $t \in \mathbb{R}$. It is easily seen that $Q_\Phi$ is non-negative if and only if the coefficient matrix $\tilde{\Phi}$ is non-negative. If the system $\mathfrak{B} \in \mathcal{L}^q$ is given in an image representation $w = M(\frac{d}{dt})\ell$ and $\Phi(\zeta, \eta)$ is a two-variable $q \times q$ polynomial matrix, then $Q_\Phi$ is called *non-negative on* $\mathfrak{B}$ if $Q_\Phi(w) \geq 0$ for all $w \in \mathfrak{B} \cap \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}^q)$. This holds if and only if the quadratic differential form $Q_{\Phi_1}$ associated with

$$\Phi_1(\zeta, \eta) := M^T(\zeta)\Phi(\zeta, \eta)M(\eta)$$

is non-negative. If $\Phi(\zeta, \eta) = S$, where $S$ a constant real matrix, and if $M(\xi) = \sum_{k=0}^{N} M_k \xi^k$, then the coefficient matrix $\tilde{\Phi}_1$ associated with $\Phi_1(\zeta, \eta)$ is given by

$$\tilde{\Phi}_1 = \tilde{M}^T S \tilde{M}$$

with $\tilde{M} := (M_0 \; M_1 \; \ldots \; M_N)$ the coefficient matrix of $M(\xi)$. Hence, the quadratic differential form $w^T S w$ is non-negative on the system $\mathfrak{B}$ if and only if $\tilde{M}^T S \tilde{M}$ is non-negative.

An important role in the computation of the $H_\infty$-controller is played by a certain one-variable polynomial matrix associated with a two-variable polynomial matrix $\Phi(\zeta, \eta)$. This one-variable polynomial matrix can be obtained by means of the delta operator $\partial : \mathbb{R}^{q_1 \times q_2}[\zeta, \eta] \to \mathbb{R}^{q_1 \times q_2}[\xi]$, defined as

$$\partial\Phi(\xi) := \Phi(-\xi, \xi)$$

A polynomial matrix $M \in \mathbb{R}^{q \times q}[\xi]$ is called *para-Hermitian* if $M = M^*$, where $M^*(\xi) := M^T(-\xi)$. Note that $(\partial\Phi)^* = \partial(\Phi^*)$, hence if $\Phi \in \mathbb{R}_s^{q \times q}[\zeta, \eta]$, then $\partial\Phi$ is para-Hermitian.

## 3.1 Dissipative Systems and Storage Functions

When we want to know whether stabilizing $H_\infty$-controllers exist for a given system, an important role will be played by the notions of (strict) dissipativeness and storage functions. In this next part we will give a short introduction in these notions, which are related to the quadratic differential forms we have just explained.

Consider a system $\mathfrak{B}$ given in the observable image representation $w = M(\frac{d}{dt})\ell$, with $M \in \mathbb{R}^{q \times l}$ of full column rank for all $\lambda \in \mathbb{C}$. And let $Q_\Phi : \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}^q) \to \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}); w \mapsto Q_\Phi(w)$ be the quadratic differential form associated with a given two-variable polynomial matrix $\Phi \in \mathbb{R}_s^{q \times q}[\zeta, \eta]$. $Q_\Phi$ will be called the *supply rate*. The system will be called *dissipative with respect to the supply rate* $Q_\Phi$ if for all $w \in \mathfrak{B} \cap \mathfrak{D}(\mathbb{R}, \mathbb{R}^q)$ there holds

$$\int_{-\infty}^{\infty} Q_\Phi(w) dt \geq 0 \tag{3.2}$$

With $\mathfrak{D}(\mathbb{R},\mathbb{R}^q)$ we denote the space of all functions $f:\mathbb{R}\to\mathbb{R}^q$ with compact support. The system will be called *strictly dissipative* with respect to the supply rate $Q_\Phi$ if there exists $\varepsilon>0$ such that for all $w\in\mathfrak{B}\cap\mathfrak{D}(\mathbb{R},\mathbb{R}^q)$

$$\int_{-\infty}^{\infty} Q_\Phi(w)dt \geq \varepsilon^2 \int_{-\infty}^{\infty} \| w(t) \|^2 \, dt \tag{3.3}$$

If for all $w\in\mathfrak{B}\cap\mathfrak{D}(\mathbb{R},\mathbb{R}^q)$ there holds $\int_{-\infty}^{\infty} Q_\Phi(w)dt=0$ then $\mathfrak{B}$ is called *lossless.*

Given the image representation $w=M(\frac{d}{dt})\ell$, and the two-variable polynomial matrix $\Phi(\zeta,\eta)$ we define $\Phi'(\zeta,\eta)\in\mathbb{R}_s^{l\times l}[\zeta,\eta]$ by $\Phi'(\zeta,\eta):=M^T(\zeta)\Phi(\zeta,\eta)M(\eta)$. Now, since $w$ and $\ell$ are related by the image representation, $Q_\Phi(w)=Q_{\Phi'}(\ell)$. Therefore, we come to the following condition for (strict) dissipativeness:

The system is dissipative if and only if

$$M^T(-i\omega)\Phi(-i\omega,i\omega)M(i\omega)\geq 0 \qquad \text{for all } \omega\in\mathbb{R} \tag{3.4}$$

The system is strict dissipative if and only if there exists $\varepsilon>0$ such that

$$M^T(-i\omega)\Phi(-i\omega,i\omega)M(i\omega)\geq \varepsilon^2 M^T(-i\omega)M(i\omega) \qquad \text{for all } \omega\in\mathbb{R}$$

It is well known that if (3.4) holds then we can factorize

$$\Phi'(-\xi,\xi)=M^T(-\xi)\Phi(-\xi,\xi)M(\xi)=F^T(-\xi)F(\xi)$$

with $F\in\mathbb{R}^{l\times l}[\xi]$. Introduce now a two-variable polynomial $\Delta$ by $\Delta(\zeta,\eta):=M^T(\zeta)\Phi(\zeta,\eta)M(\eta)-F^T(\zeta)F(\eta)$. Since $\partial\Delta=0$, the two-variable polynomial $\Delta$ must contain a factor $\zeta+\eta$, and therefore we can define a new two-variable polynomial by

$$\Psi(\zeta,\eta):=(\zeta+\eta)^{-1}\Delta(\zeta,\eta). \tag{3.5}$$

Consider now the quadratic differential forms $Q_\Psi$ and $Q_\Delta$ associated with $\Psi$ and $\Delta$, respectively. We have $Q_\Delta(\ell)=Q_{\Phi'}(\ell)-\| F(\frac{d}{dt})\ell \|^2$. Furthermore, (3.5) is equivalent to: $\frac{d}{dt}Q_\Psi(\ell)=Q_\Delta(\ell)$ for all $\ell\in\mathfrak{D}(\mathbb{R},\mathbb{R}^l)$. Thus we obtain

$$\frac{d}{dt}Q_\Psi(\ell)(t)\leq Q_{\Phi'}(\ell)(t), \tag{3.6}$$

for all $\ell\in\mathfrak{C}^\infty(\mathbb{R},\mathbb{R}^l)$, for all $t\in\mathbb{R}$. (3.6) is called the *dissipation inequality.* Any quadratic differential form $Q_\Psi:\mathfrak{C}^\infty(\mathbb{R},\mathbb{R}^l)\to\mathfrak{C}^\infty(\mathbb{R},\mathbb{R})$ that satisfies this inequality is called a *storage function.* It can be shown that $\mathfrak{B}$ is dissipative if and only if there exists a symmetric two-variable polynomial matrix $\Psi(\zeta,\eta)$ such that the corresponding quadratic differential form $Q_\Psi$ satisfies (3.6). In general, storage functions are not unique. We will see in section 4.2 that the existence of a negative definite storage function for a certain dissipative system is equivalent to the existence of a stabilizing $H_\infty$-controller.

# 4 $H_\infty$ Control in the Behavioral Approach

In section 2 we have formulated the control problem in the behavioral approach. In the context of $H_\infty$-control, the main desired property of the controlled system is that the to-be-controlled variables are small (in an appropriate sense) regardless of the values that the disturbances take. And in addition the controlled system should be stable, i.e., if the disturbances are zero, then the to-be-controlled variables should converge to zero as time runs off to infinity. Small in an appropriate sense means that the $L_2$ induced norm of the system is small. The size of the signals is measured by their quadratic integrals:

$$\| z \|_2^2 = \int \| z(t) \|^2 \, dt$$

where the integrals range over $\mathbb{R}$. Now, the to-be-controlled variable $z$ being small regardless of the disturbance $d$ together with the stability, happens to be the same as the $H_\infty$-norm of (the transfer function of) the controlled system is small. This $H_\infty$-norm is defined as:

$$\| G \|_\infty := \sup_{\Re e(s) \geq 0} \| G(s) \|$$

We already mentioned that a controller $\mathfrak{B}_c$ for the system $\mathfrak{B}_p$ is *admissible* if in the controlled system $\mathfrak{B}_p \wedge \mathfrak{B}_c$ the disturbance is still free.

**Definition 4.1 :** Let $\mathfrak{B}_c$ be an admissible controller. The $H_\infty$ performance of the controlled system $\mathfrak{B}_p \wedge \mathfrak{B}_c$ is defined as

$$\begin{aligned} J(\mathfrak{B}_c) := \inf\{\gamma \geq 0 \quad | \quad & \| z \|_2 \leq \gamma \| d \|_2 \ \text{ for all} \\ & (z,d) \in (\mathfrak{B}_p \wedge \mathfrak{B}_c) \cap L_2(\mathbb{R}, Z \times D)\} \end{aligned}$$

With $L_2(\mathbb{R}, Z \times D)$ we denote the space of all functions $f : \mathbb{R} \to Z \times D$ for which $\int \| f(t) \|^2 \, dt$ is finite.

**Definition 4.2 :** Given $\gamma > 0$ (the *tolerance*), the controller $\mathfrak{B}_c$ is called

*$\gamma$-contracting* if $J(\mathfrak{B}_c) \leq \gamma$, or equivalently, if for all $(z,d) \in (\mathfrak{B}_p \wedge \mathfrak{B}_c) \cap L_2(\mathbb{R}, Z \times D)$ we have $\| z \|_2 \leq \gamma \| d \|_2$

*strictly $\gamma$-contracting* if $J(\mathfrak{B}_c) < \gamma$, equivalently, if there exists $\varepsilon > 0$ such that for all $(z,d) \in (\mathfrak{B}_p \wedge \mathfrak{B}_c) \cap L_2(\mathbb{R}, Z \times D)$ we have $\| z \|_2 \leq (\gamma - \varepsilon) \| d \|_2$

**Definition 4.3 :** An admissible controller $\mathfrak{B}_c$ is called a *stabilizing controller* if in the controlled system the signal $z$ converges to zero whenever $d = 0$, or equivalently, if $(z, 0) \in \mathfrak{B}_p \wedge \mathfrak{B}_c$ implies that $\lim_{t \to \infty} z(t) = 0$.

Now, given $\gamma > 0$, the $H_\infty$ *suboptimal* control problem is to determine all $\gamma$-contracting stabilizing controllers (if one exists). In addition the *strict $H_\infty$ suboptimal* control problem is to determine all strictly $\gamma$-contracting stabilizing controllers.

The $H_\infty$ *optimal* control problem is to minimize the $H_\infty$ performance of the controlled system over the class of all admissible stabilizing controllers, i.e. to calculate $\gamma^* := \inf\{J(\mathfrak{B}_c) | \mathfrak{B}_c$ admissible and stabilizing$\}$ and to calculate all optimal controllers $\mathfrak{B}_c^*$ with the property that $\gamma^* = J(\mathfrak{B}_c^*)$ (if one exists).

## 4.1 The Full Information $H_\infty$ Control Problem

In this paper we are only concerned with the solution of the *full information $H_\infty$-control problem.* By this we mean the interconnection variable $c$ is a *full information variable* for the system $\mathfrak{B}_p$. This means that the whole manifest variable $w = \mathrm{col}(z, d, c)$ can be determined from $c$ alone.

**Definition 4.4 :** Suppose our system $\mathfrak{B}_p$ is given in observable image representation $w = M(\frac{d}{dt})\ell$, where $w = \mathrm{col}(z, d, c)$ and $M(\frac{d}{dt}) = \mathrm{col}(Z(\frac{d}{dt}), D(\frac{d}{dt}), C(\frac{d}{dt}))$. Then the interconnection variable $c$ is a full information variable for $\mathfrak{B}_p$ if and only if $c = C(\frac{d}{dt})\ell$ is observable, equivalently if and only if $\mathrm{rank}(C(\lambda)) = l$ (full column rank) for all $\lambda \in \mathbb{C}$.

Now, if $c$ is a full information variable we call the corresponding $H_\infty$ optimal and suboptimal control problems *full information problems.*

If in the plant $\mathfrak{B}_p$ $c$ is a full information variable, then the set of controllers of the form

$$c = C(\frac{d}{dt})\ell, \; K(\frac{d}{dt})\ell = 0 \tag{4.1}$$

yields the same set of controlled systems as the set of controllers of the form

$$K'(\frac{d}{dt})c = 0$$

Without loss of generality we will restrict ourselves to polynomial matrices $K$ with full row rank.

**Lemma 4.5 :** *Consider as before the plant $\mathfrak{B}_p$ with observable image representation $w = M(\frac{d}{dt})\ell$. Assume $c$ is a full information variable. Then the controller $\mathfrak{B}_c$ given by (4.1) with $K$ of full row rank is admissible if and only if $\mathrm{col}(D, K)$ has full row rank.*

The class of all admissible controllers $\mathfrak{B}_c$ given by (4.1) will be denoted by $\mathfrak{C}$. Note that if $\mathfrak{B}_c$ is admissible and $K$ has full row rank, then $K$ has at most $l - d$ rows (number of latent variables minus the dimension of the disturbance).

Now, if the system is given by the observable image representation

$$\begin{bmatrix} z \\ d \\ c \end{bmatrix} = \begin{bmatrix} Z(\frac{d}{dt}) \\ D(\frac{d}{dt}) \\ C(\frac{d}{dt}) \end{bmatrix} \ell$$

we can easily reformulate the definitions of (strictly) $\gamma$-contracting (stabilizing) controllers in terms of the polynomial matrices.

## 4.2 Contracting Stabilizing Controllers

For now, we are only interested in controllers $\mathfrak{B}_c \in \mathfrak{C}$ which are (strictly) $\gamma$-contracting *and* stabilizing. In this section we will give conditions under which a controller is both (strictly) $\gamma$-contracting and stabilizing. We will also give conditions under which strictly $\gamma$-contracting stabilizing controllers exist for a given system.

### 4.2.1 Preliminaries

In the following we will consider the plant $\mathfrak{B}_p$ given in image representation

$$\begin{bmatrix} z \\ d \\ c \end{bmatrix} = \begin{bmatrix} Z(\frac{d}{dt}) \\ D(\frac{d}{dt}) \\ C(\frac{d}{dt}) \end{bmatrix} \ell \quad \text{or} \quad \begin{bmatrix} w \\ c \end{bmatrix} = \begin{bmatrix} M(\frac{d}{dt}) \\ C(\frac{d}{dt}) \end{bmatrix} \ell \tag{4.2}$$

We assume $c$ to be a full information variable, so $C(\lambda)$ has full column rank for all $\lambda \in \mathbb{C}$ and we assume $M(\lambda)$ to have full column rank for all $\lambda \in \mathbb{C}$. We will also consider the system $\mathfrak{B}$ given in observable image representation

$$\begin{bmatrix} z \\ d \end{bmatrix} = M(\frac{d}{dt})\ell \tag{4.3}$$

For a given $\gamma > 0$, we define the $(z + d) \times (z + d)$ real diagonal matrix $\Sigma_\gamma$ by

$$\Sigma_\gamma := \begin{bmatrix} I_z & 0 \\ 0 & -\gamma^2 I_d \end{bmatrix} \tag{4.4}$$

Associated with the plant $\mathfrak{B}_p$ and $\gamma > 0$ we will consider the symmetric two-variable $l \times l$ polynomial matrix $\Phi_\gamma(\zeta, \eta)$ defined by

$$\Phi_\gamma(\zeta, \eta) := M^T(\zeta)\Sigma_\gamma M(\eta) = Z^T(\zeta)Z(\eta) - \gamma^2 D^T(\zeta)D(\eta) \tag{4.5}$$

As mentioned before, this two-variable polynomial matrix induces a one-variable $l \times l$ polynomial matrix $\partial\Phi_\gamma(\xi) = \Phi_\gamma(-\xi, \xi)$.

### 4.2.2 Conditions

A necessary and sufficient condition for a controller to be stabilizing and strictly $\gamma$-contracting is given in the following lemma.

**Lemma 4.6 :** *Let $\gamma > 0$ and let $\mathfrak{B}_c \in \mathfrak{C}$ be represented by $c = C(\frac{d}{dt})\ell$, $K(\frac{d}{dt})\ell = 0$, with $K$ full row rank. The following statements are equivalent*

1. *$\mathfrak{B}_c$ is stabilizing and strictly $\gamma$-contracting*
2. *$\begin{bmatrix} D \\ K \end{bmatrix}$ is Hurwitz and there exists $\varepsilon > 0$ such that for all $\omega \in \mathbb{R}$ and $v \in \ker K(i\omega)$ we have $v^* M^T(-i\omega)\Sigma_\gamma M(i\omega)v \leq -\varepsilon^2 \parallel M(i\omega)v \parallel^2$*
3. *$\begin{bmatrix} D \\ K \end{bmatrix}$ is Hurwitz and the rational matrix $G := Z \begin{bmatrix} D \\ K \end{bmatrix}^{-1} \begin{bmatrix} I_d \\ 0 \end{bmatrix}$ is proper and satisfies $\parallel G \parallel_\infty < \gamma$.*

**Remark 4.7 :** We can state a similar result for $\gamma$-contracting stabilizing controllers. In that case, we get in the second statement $\varepsilon$ equal to zero and in the third $\parallel G \parallel_\infty \leq \gamma$.

**Remark 4.8 :** If $\mathfrak{B}_c$ is a strictly $\gamma$-contracting stabilizing controller, then in the controlled system $\mathfrak{B}_p \wedge \mathfrak{B}_c$ the variables $d$ and $z$ are related by the proper rational matrix $G$ with $\parallel G \parallel_\infty < \gamma$. Furthermore, $G$ has all its poles in $\Re e(\lambda) < 0$ so the $L_\infty$-norm of $G$ is equal to the $H_\infty$-norm of $G$. Thus, we see that $\mathfrak{B}_c$ is a stabilizing and strictly $\gamma$-contracting controller if and only if $\begin{bmatrix} D \\ K \end{bmatrix}$ is Hurwitz and in the controlled system the variables $d$ and $z$ are related by a proper rational matrix with $H_\infty$-norm less than $\gamma$.

### 4.2.3 Existence

Now we know under what conditions a controller is stabilizing and $\gamma$-contracting, but we haven't mentioned the existence of such controllers. It turns out that there exists a strictly $\gamma$-contracting stabilizing controller $\mathfrak{B}_c$ for the plant $\mathfrak{B}_p$ (given by (4.2)) if and only if the dual $\mathfrak{B}^\perp$ of the system $\mathfrak{B}$ (given by (4.3)) (i) is strictly dissipative with respect to the supply rate $\bar{w}^T \Sigma_{\frac{1}{\gamma}} \bar{w}$ (where $\Sigma_{\frac{1}{\gamma}}$ is defined by (4.4)) and (ii) has a *negative definite* storage function. The existence of strictly $\gamma$-contracting stabilizing controllers is also equivalent with the existence of certain regular Hurwitz factorizations (see appendix A) of the polynomial matrix $\partial\Phi_\gamma$. These factorizations will yield explicit formulas for the controllers we are looking for. The main result is the following theorem.

**Theorem 4.9** : *Let $\gamma > 0$, the following statements are equivalent:*

1. *there exists a strictly $\gamma$-contracting stabilizing controller*
2. *$\mathfrak{B}^{\perp}$ is strictly dissipative with respect to the supply rate $\bar{w}^T \Sigma_{\frac{1}{\gamma}} \bar{w}$ and there exists a negative definite storage function*
3. *there exists a polynomial matrix $F \in \mathbb{R}^{l \times l}[\xi]$ such that*
   (a) $\partial\Phi_\gamma(\xi) = F^T(-\xi) J_{l-d,d} F(\xi)$
   *(here $J_{l-d,d}$ is the signature matrix with the dimension of the positive part equal to $l - d$ and of the negative part to $d$)*
   (b) *$F$ is Hurwitz*
   (c) *$MF^{-1}$ is proper*
   (d) $\begin{bmatrix} D \\ F_+ \end{bmatrix}$ *is Hurwitz*

*Here, $F_+$ is obtained by partitioning $F$ into $\begin{bmatrix} F_+ \\ F_- \end{bmatrix}$, where $F_+$ has $l - d$ rows and $F_-$ has $d$ rows. If $F$ is a polynomial matrix such that (3) is satisfied, then $F_+$ has full row rank, and the controller $\mathfrak{B}_c$ represented by $c = C(\frac{d}{dt})\ell$, $F_+(\frac{d}{dt})\ell = 0$ is admissible, strictly $\gamma$-contracting stabilizing.*

The third statement is useful in order to compute the $\gamma$-contracting stabilizing controller, but it is not really useful for checking the existence of such controllers. The factorization in (3a) is not unique, so if a factorization is computed, it might not satisfy all the other conditions in (3) but that doesn't mean there does not exist a $\gamma$-contracting stabilizing controller. To conclude that, all other spectral factors $UF$ of $\partial\Phi_\gamma$ (where $U$ is a $J$-unitary unimodular matrix) should be checked. This is not a simple problem. So, what we would like is a simple test to check whether a $\gamma$-contracting stabilizing controller exists for a given plant. For instance, a test to decide whether a given strictly dissipative system has a negative definite storage function or not. This can be tested with the so called *Pick matrix test.*

*A Pick Matrix Test*

It will be shown that there exists a negative definite storage function if and only if a certain Pick matrix associated with the system is negative definite. We will give the definition of this Pick matrix directly applied to our $H_\infty$ problem.

Consider the plant $\mathfrak{B}_p$ given in image representation (4.2). We still assume $M(\lambda)$ and $C(\lambda)$ to have full column rank for all $\lambda \in \mathbb{C}$. Consider also the system $\mathfrak{B}$ given by the observable image representation (4.3) and its dual $\mathfrak{B}^{\perp}$ with observable image representation $\bar{w} = R^T(-\frac{d}{dt})\bar{\ell}$. Let $\gamma > 0$ be a given tolerance. Assume that $\mathfrak{B}^\perp$ is strictly dissipative with respect to the supply rate $\bar{w}^T \Sigma_{\frac{1}{\gamma}} w$. Define $\Upsilon(\zeta,\eta) := R(-\zeta)\Sigma_{\frac{1}{\gamma}} R^T(-\eta)$. Note that by strict dissipativity we have $\partial\Upsilon(i\omega) \geq \varepsilon R(i\omega)R^T(-i\omega) > 0$ for all $\omega \in \mathbb{R}$, so $\det(\partial\Upsilon)$ has no roots on the imaginary axis. We will now define the Pick matrix associated with $\Upsilon(\zeta,\eta)$ and denote it by $T_\Upsilon$. We will only give the expression for the Pick matrix in case $\partial\Upsilon$ is *semi-simple*, since in this case it is much simpler, for the general case we refer to [5]. A polynomial matrix $M \in \mathbb{R}^{q\times q}[\xi]$, $\det(M) \neq 0$, is called *semi-simple* if for all $\lambda \in \mathbb{C}$ the dimension of $\ker(M(\lambda))$ is equal to the multiplicity of $\lambda$ as a root of the polynomial $\det(M)$.

**Definition 4.10 :** (semi-simple case)
Let $\lambda_1, \lambda_2, \ldots, \lambda_k \in \mathbb{C}$ be the *distinct* roots of $\det(\partial\Upsilon)$ with $\Re e(\lambda_i) < 0 \quad i = 1, \ldots k$, and let $a_1, a_2, \ldots, a_k \in \mathbb{C}^l$ be linearly independent vectors and such that $\partial\Upsilon(\lambda_i)a_i = 0$, and such that the $a_k$'s associated with the same $\lambda_i$ span $\ker(\partial\Upsilon(\lambda_i))$. Then we define $T_\Upsilon \in \mathbb{C}^{k\times k}$ to be the Hermitian matrix $T_\Upsilon$ whose $(i,j)$-th element $(T_\Upsilon)_{i,j} \in \mathbb{C}$ is given by

$$(T_\Upsilon)_{i,j} := \frac{a_i^* \Upsilon(\bar{\lambda}_i, \lambda_j) a_j}{\bar{\lambda}_i + \lambda_j}$$

We now have the following

**Theorem 4.11 :** *The strictly dissipative system $\mathfrak{B}^\perp$ has a negative definite storage function if and only if the associated Pick matrix $T_\Upsilon < 0$*

Now the equivalence relation (given $\gamma > 0$):

*there exists a strictly $\gamma$-contracting stabilizing controller*

$$\Leftrightarrow$$

*$\mathfrak{B}^\perp$ is strictly dissipative with respect to the supply rate $\bar{w}^T \Sigma_{\frac{1}{\gamma}} \bar{w}$ and there exists a negative definite storage function*

can be reformulated as

**Theorem 4.12 :** *Let $\gamma > 0$. There exists a strictly $\gamma$-contracting stabilizing controller if and only if there exists $\varepsilon > 0$ such that $\partial\Upsilon(i\omega) \geq \varepsilon R(i\omega)R^T(-i\omega)$ for all $\omega \in \mathbb{R}$ (strict dissipativity) and $T_\Upsilon < 0$ (existence of negative definite storage function).*

## 4.3 How to Compute the $H_\infty$ Controller

We will now summarize how to compute a strictly $\gamma$-contracting stabilizing $H_\infty$ controller for a given system, represented by (4.3).

The first thing to do is to find out whether such a controller exists for the given system. It is no use to start computing a factorization if there might not exist a contracting stabilizing controller. Use the *Pick matrix test* to check whether a $\gamma$-contracting stabilizing controller exists for the given system and $\gamma$.

Next, if there exists a strictly $\gamma$-contracting stabilizing controller for the given system and $\gamma$, factorize $\partial\Phi_\gamma(\xi) = F^T(-\xi)J_{l-d,d}F(\xi)$, where $\Phi_\gamma$ is given by (4.5). The factorization can be done by different methods as mentioned in section A. If this $F \in \mathbb{R}^{l\times l}[\xi]$ satisfies

1. $F$ is Hurwitz
2. $MF^{-1}$ is proper
3. $\begin{bmatrix} D \\ F_+ \end{bmatrix}$ is Hurwitz

then $c = C(\frac{d}{dt})\ell$, $K(\frac{d}{dt})\ell = 0$ becomes an admissible strictly $\gamma$-contracting stabilizing controller for the given system, where $K = F_+$. If, however, $F$ does not satisfy all the constraints above, find a $J$-unitary unimodular matrix $U$ so that $UF$ satisfies the constraints. And then choose $K = (UF)_+$ for the controller.

# 5 Example, Car Suspension

In this section we will apply the theory explained in the previous sections to a practical situation. We will compute the (optimal) suspension of a vehicle that drives with constant speed on a bumpy road. For this example we need a model for the car on the road. We'll represent the car (chassis) as a (point)mass, the suspension simply as a force (we want to compute the suspension) and the tires as a (point)mass and spring with length $l$ and spring constant $k$. Schematic it looks like this:

$M_1$: mass of car + driver
$M_2$: mass of tire
$F_c$ : suspension force
$g$ : gravity acceleration
$z_1$ : height of driver
$z_2$ : height of tire
$d$ : road profile (disturbance)
$k$ : spring constant
$l$ : spring length

The cloud with the question mark represents the suspension device we like to model and compute.

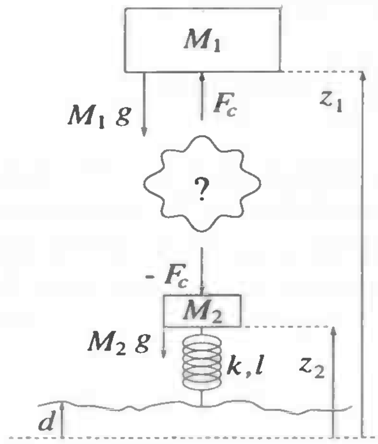


Figure 5.1

The behavioral equations follow from the laws of physics:

$$
\begin{aligned}
M_1 \frac{d^2}{dt^2} z_1 &= F_c - M_1 g \\
M_2 \frac{d^2}{dt^2} z_2 &= k(l - (z_2 - d)) - F_c - M_2 g
\end{aligned}
\tag{5.1}
$$

We will denote the equilibrium of the variable $d$ by $\bar{d}$, and likewise for the other various variables. In the case $d(t) = \bar{d}$ and $F_c(t) = \bar{F}_c$ we find the following equilibrium solution:

$$
\left\{ \begin{aligned} 0 &= \bar{F}_c - M_1 g \\ 0 &= k(l - \bar{z}_2 + \bar{d}) - \bar{F}_c - M_2 g \end{aligned} \right. \quad \Leftrightarrow \quad \left\{ \begin{aligned} \bar{F}_c &= M_1 g \\ \bar{z}_2 &= l + \bar{d} - (M_1 + M_2)\tfrac{g}{k} \end{aligned} \right.
$$

Note that these equations do not specify $\bar{z}_1$, for this will depend on the character of the suspension-device.

Now, to compute this suspension-device we need to know what kind of performance we'd like. One thing is for sure we'd like the driver to sit comfortable behind the wheel, so the so called *jerk*, $\frac{d^3}{dt^3} z_1$, needs to be small, as well as the acceleration, $\frac{d^2}{dt^2} z_1$. As for the height-difference of the driver with the equilibrium, $z_1 - \bar{z}_1$, this must be small when the car drives over a ribbed road, but when the car drives on a gradual hill we want the driver to follow the road, the higher the car goes the bigger $z_1 - \bar{z}_1$ becomes. So, we want to suppress high frequency disturbances (ribbed roads) and follow low frequency disturbances (smooth hills). Therefore we place a low-pass filter over the disturbance. To summarize, we want the following three variables to be small:

$$\frac{d^3}{dt^3} z_1$$
$$\frac{d^2}{dt^2} z_1$$
$$z_1 - \bar{z}_1 - v$$

In this last one, $v$ is the low-pass filter, a function of the disturbance, which in the frequency domain is :

$$v(i\omega) = \frac{1}{1 + \alpha i\omega}(d(i\omega) - \hat{d}(i\omega)) \tag{5.2}$$

We want $z_1 - \bar{z}_1$ to follow the disturbance $d - \bar{d}$ when it is low frequency (like smooth hills) but when it is high frequency (like bumps in the road) we want $z_1 - \bar{z}_1$ to be zero. If the disturbance is high frequency, $\omega$ is large, so $v \approx 0$ and therefore $z_1 - \bar{z}_1 \to 0$, the bumps are ignored. If the disturbance is low frequency, $\omega$ is small, so $v \approx d - \bar{d}$ and therefore $z_1 - \bar{z}_1 \to d - \bar{d}$, 'the driver follows the road'. In the time domain (5.2) becomes ($(d - \bar{d})$ as a function of $v$):

$$(d - \bar{d}) = (1 + \alpha \frac{d}{dt})v$$

We'll add this equation to the model equations (5.1).

We'll introduce the following variables

$$\begin{aligned} \Delta_1 &= z_1 - \bar{z}_1 \\ \Delta_2 &= z_2 - \bar{z}_2 \\ \Delta_d &= d - \bar{d} \\ \Delta_F &= F_c - \bar{F}_c \end{aligned}$$

and substitute them into the model equations (5.1), so they become:

$$\begin{aligned} M_1 \ddot{\Delta}_1 &= \Delta_F \\ M_2 \ddot{\Delta}_2 &= k(\Delta_d - \Delta_2) - \Delta_F \\ \Delta_d &= v + \alpha \dot{v} \end{aligned} \tag{5.3}$$

The variables we wish to remain small are

$$\dddot{\Delta}_1 \,, \; \ddot{\Delta}_1 \,, \; \Delta_1 - v$$

In order to solve this $H_\infty$-control problem using the behavioral approach we will represent the system (5.3) as an observable image representation:

$$\begin{bmatrix} z \\ d \\ c \end{bmatrix} = \begin{bmatrix} Z(\frac{d}{dt}) \\ D(\frac{d}{dt}) \\ C(\frac{d}{dt}) \end{bmatrix} \ell$$

where $z = [\,\dddot{\Delta}_1, \ddot{\Delta}_1, \Delta_1 - v]^T$ the to-be-controlled variable, $d$ the disturbance $\Delta_d$, $c$ the interconnection variable and $\ell$ the latent variable. We assume $C(\lambda)$ to have full column rank for all $\lambda \in \mathbb{C}$ (full information) and $\mathrm{col}(Z(\lambda), D(\lambda))$ also (observability).

## 5.1 Image Representation

If we choose the latent variable to be

$$\ell = \begin{bmatrix} v \\ \Delta_1 \end{bmatrix}$$

we can easily write down an image representation:

$$\begin{bmatrix} \dddot{\Delta}_1 \\ \ddot{\Delta}_1 \\ \Delta_1 - v \\ \Delta_d \\ c \end{bmatrix} = \begin{bmatrix} 0 & \frac{d^3}{dt^3} \\ 0 & \frac{d^2}{dt^2} \\ -1 & 1 \\ 1 + \alpha\frac{d}{dt} & 0 \\ \ldots C(\frac{d}{dt}) \ldots \end{bmatrix} \begin{bmatrix} v \\ \Delta_1 \end{bmatrix} \tag{5.4}$$

Let us check if the conditions for the theory developed in the previous sections are satisfied. Thus

1. the disturbance must remain free. Since $[1 + \alpha\xi \;\; 0]$ has full row rank, $\Delta_d$ is indeed completely free.

2. In order for this representation to be observable $M(\lambda)$ must be of full column rank for all $\lambda \in \mathbb{C}$.

$$M(\lambda) = \begin{bmatrix} 0 & \lambda^3 \\ 0 & \lambda^2 \\ -1 & 1 \\ 1 + \alpha\lambda & 0 \end{bmatrix}$$

and for all $\lambda \in \mathbb{C}$ $\mathrm{rank}(M(\lambda)) = 2$, full column rank. So this representation is indeed observable.

## 5.2 Existence of a Strictly $\gamma$-Contracting Stabilizing Controller

We know from section 4.2.3 that there exists a strictly $\gamma$-contracting stabilizing controller for the system $\mathfrak{B}$ if and only if the dual system $\mathfrak{B}^{\perp}$ is strictly dissipative with respect to the supply rate $\bar{w}^T \Sigma_{\frac{1}{\gamma}} \bar{w}$ and there exists a negative definite storage function. So, we first need to compute a dual system in observable image representation and then if it is strictly dissipative, we can apply the Pick matrix test to see whether it has a negative definite storage function.

### 5.2.1 The Dual System

To find an observable image representation $\bar{w} = R^T(-\frac{d}{dt})\bar{\ell}$ for the dual system $\mathfrak{B}^{\perp}$, we need to find a controllable kernel representation $R(\frac{d}{dt})w = 0$ for the system (5.4). We can obtain such $R$ by solving $RM = 0$ over all $R \in \mathbb{R}^{2\times 4}[\xi]$ such that $\mathrm{rank}(R(\lambda)) = 2$ for all $\lambda \in \mathbb{C}$.

Suppose

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} \\ r_{21} & r_{22} & r_{23} & r_{24} \end{bmatrix}$$

where $r_{ij} \in \mathbb{R}[\xi]$ for $i = 1,2;\ j = 1,2,3,4$. Then $R(\xi)M(\xi) = 0$ leads to the following systems of equations:

$$\left\{ \begin{array}{lcl} r_{13} & = & (1+\alpha\xi)\, r_{14} \\ r_{13} & = & -r_{11}\xi^3 - r_{12}\xi^2 \end{array} \right. \quad \text{and} \quad \left\{ \begin{array}{lcl} r_{23} & = & (1+\alpha\xi)\, r_{14} \\ r_{23} & = & -r_{21}\xi^3 - r_{22}\xi^2 \end{array} \right.$$

and the vectors $(r_{11}, r_{12}, r_{13}, r_{14})$ and $(r_{21}, r_{22}, r_{23}, r_{24})$ must be linearly independent for $R(\xi)$ to have full row rank. For both systems we can come up with two similar linearly independent possible solutions, so we choose one solution for the first system and the other solution for the second.

- choose $r_{13} = r_{14} = 0 \ \Rightarrow\ r_{11}\xi = -r_{12}$
- choose $r_{23} = \xi^2 + \alpha\xi^3 \ \Rightarrow\ r_{24} = \xi^2$ and $\Rightarrow\ r_{21} = -\alpha,\ r_{22} = -1$

This way we find the kernel representation of $\mathfrak{B}$, specified by the following polynomial matrix

$$R(\xi) = \begin{bmatrix} 1 & -\xi & 0 & 0 \\ -\alpha & -1 & \xi^2 + \alpha\xi^3 & \xi^2 \end{bmatrix}$$

and indeed this is controllable, so we find for the dual system the observable image representation:

$$\bar{w} = \begin{bmatrix} 1 & -\alpha \\ \frac{d}{dt} & -1 \\ 0 & \frac{d^2}{dt^2} - \alpha\frac{d^3}{dt^3} \\ 0 & \frac{d^2}{dt^2} \end{bmatrix} \bar{\ell}$$

The question we will now deal with is if the dual is strictly dissipative with respect to the supply rate $\bar{w}^T \Sigma_{\frac{1}{\gamma}} \bar{w}$, or equivalently, does there exist $\varepsilon > 0$ such that

$$R(i\omega)\Sigma_{\frac{1}{\gamma}} R^T(-i\omega) \geq \varepsilon R(i\omega) R^T(-i\omega)$$

for all $\omega \in \mathbb{R}$? Depending on $\gamma$ there exists such $\varepsilon$, for instance, if $\gamma > 1$ we might choose $\varepsilon = 1/\gamma^2$.

But, we also have to verify that $\mathfrak{B}^{\perp}$ has a negative definite storage function.

### 5.2.2 The Pick matrix Test

We will use the Pick matrix test to check whether the dual system has a negative definite storage function. We have written a package in MATHEMATICA which checks if, given $\alpha$ and $\gamma$, the Pick matrix is negative definite (see appendix C, we will explain the parameter `b`, in the text it will be mentioned as $\beta$, later). This allows us to conclude whether there exists a contracting stabilizing controller for this system. For the moment the package only works if the roots of $\det(R(\xi)\Sigma_{\frac{1}{\gamma}} R^T(-\xi))$ are simple (but we only encountered once a root with multiplicity $> 1$ and in that case it was quite easy computed directly that the Pick matrix was negative definite). In table 5.1 we have listed for some combinations of $\alpha$ and $\gamma$ the results: *yes* means there exists a contracting stabilizing controller, *no* means there does not and *?* means the algorithm was not conclusive.

| $\alpha \setminus \gamma$ | 0.7 | 0.8 | 0.9 | 1 |
|---|---|---|---|---|
| 1 | yes | yes | yes | ? |
| 0.8 | no | yes | yes | yes |
| 0.6 | no | yes | yes | yes |

Table 5.1

## 5.3 Computation of a Strictly $\gamma$-Contracting Stabilizing Controller

If we know for what $\alpha$ and $\gamma$ there exist a strictly $\gamma$-contracting controller we can start computing it. According to section 4.3 we must factorize $M^T(-\xi)\Sigma_\gamma M(\xi) = F^T(-\xi)J_{1,1}F(\xi)$ so that (i) the factor $F \in \mathbb{R}^{2\times 2}[\xi]$ is Hurwitz, (ii) $MF^{-1}$ is proper, and (iii) $\mathrm{col}(D, F_+)$ is Hurwitz.

For the computation of this factorization we used an already existing macro for MATLAB (`jsp.m`), written by H.Kwakernaak from the university of Twente (see [3]). The factorization algorithm is based on symmetric factor extraction. For this algorithm the to-be-factored para-Hermitian polynomial matrix must be *diagonally reduced* (see appendix A). Note that in this example The *diagonal leading coefficient matrix* of $M^T(-\xi)\Sigma_\gamma M(\xi)$ is non-singular and therefore $M^T(-\xi)\Sigma_\gamma M(\xi)$ is *diagonally reduced*. We have written ourselves a MATLAB macro (see appendix D.1, `controller.m`) which checks if the factorization satisfies the above three constraints and therefore if the factorization yields a suitable controller or not. In this macro the controller is transformed into state space form for simulation purposes. Although the major advantage of looking to systems and controllers in the behavioral view is that you don't have to point out input, state and output variables.

We use for a given $\alpha$ the smallest $\gamma$ for which a $\gamma$-contracting stabilizing controller still exists (thus the Pick matrix test), to compute the controller in MATLAB. However, it is not at all necessary that the factorization algorithm yields immediately the correct factor. What is done if the factor $F$ doesn't satisfy all the constraints, is to find a $J$-unitary unimodular matrix $U$ such that $UF$ satisfies all the constraints. There are however no complete algorithms, to find that unimodular matrix $U$ such that $UF$ is a correct factor. Therefore we simply adjust $\gamma$ a little bit until the macro `jsp` comes up with the right one. It is worth mentioning that during our simulations we mainly encountered the problem that the factor $F$ did not satisfy $\mathrm{col}(D, F_+)$ being Hurwitz. The algorithmic question is a matter of further research.

The controller we want is $c = C(\frac{d}{dt})\ell$, $K(\frac{d}{dt}) = 0$, where $K(\xi) = F_+(\xi) = [a(\xi) \;\; b(\xi)]$. In all our computations we came up with controllers of the form $K(\xi) = [a \;\; b(\xi)]$, where $\deg(b(\xi))$ is 3 and, of course, $a$ is a constant.

## 5.4 Simulations

In order to evaluate the performance of the controlled system, we simulate the controlled system in MATLAB. We compute Bode plots, plots of the step responses, and we also simulate the reaction of the car when he's driving over a specific kind of road profile. For instance, a ribbed road, modeled by a high frequency sine wave with small amplitude, or a gradual hill, modeled by a low frequency sine wave with large amplitude.

The first results were not very good (see appendix B.1). If we looked at the step response of the system, we noticed, although $\Delta_d = 1$, there did not hold $\Delta_1 \to 1$. This is somewhat strange, the car goes up a step in the road ($\Delta_d = 1$) but the chassis (and therefore the driver) goes up less ($\Delta_1 \not\to 1$), the suspension pulls it down. How does this happen?
The controller was computed so that $\dddot{\Delta}_1$, $\ddot{\Delta}_1$ and $\Delta_1 - v$ become small (in an appropriate sense). But we didn't say anything about the importancy of these three variables to become small. It seems that the first two variables are much more important to the controller than $\Delta_1 - v$, they dominate $\Delta_1 - v$. Therefore we decide to put in a weighting, instead of just wanting $\Delta_1 - v$ to be small we now want $\beta(\Delta_1 - v)$ to be small, where $\beta > 1$. In the model, we can simply change the diagonal matrix $\Sigma_\gamma$ into $\mathrm{diag}(1, 1, \beta^2, -\gamma^2)$. The algorithms (etc.) won't change by this adjustment.

To make the simulations easier we have written some MATLAB functions (see appendix D.2).

After suitably adjusting the parameters we got rather nice results for $\alpha = 4$, $\beta = 5$ and $\gamma = 0.90$. Increasing $\alpha$ meant the step got better but it took more time, increasing $\beta$ also meant better results but it also meant that $\gamma$ increased. What we preferred was rather good results in not to long a time and $\gamma$ small if possible (see appendix B.2).

### 5.4.1 The Bode Plot

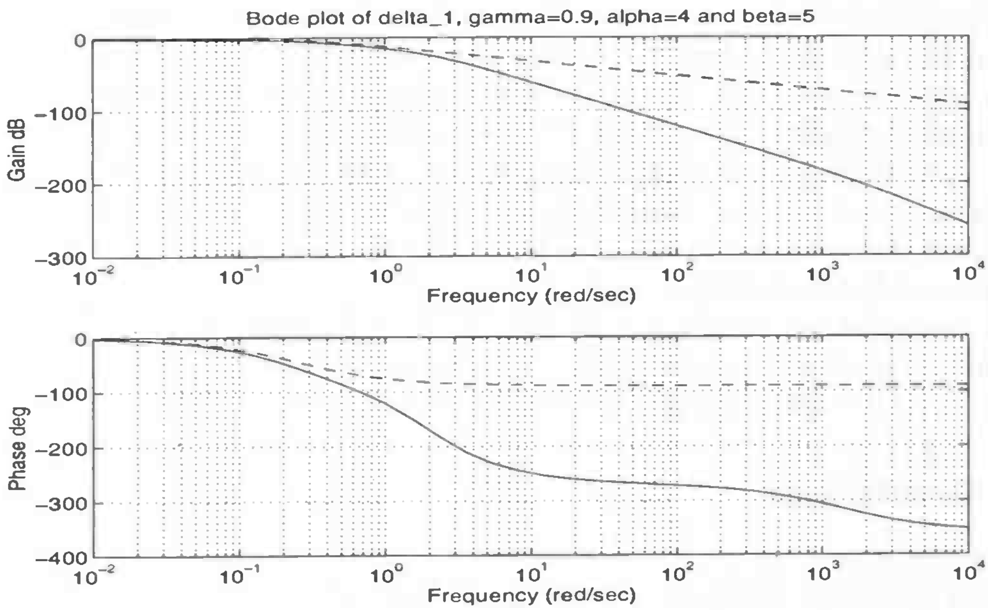


Figure 5.2

The dashed line is the frequency response of $\Delta_d \to v$.
What we see here (figure 5.2), is that indeed low frequency disturbances are let through and high frequency disturbances are suppressed.

### 5.4.2 The Step Response

We place a step (=1) on the input (in this case the input is the disturbance) and make a plot of the response of the car ($\Delta_1$). We can interpret the step as going up one level, the road now is parallel to the ground level but at height 1. We expect the car to do the same, go 'drive at height 1'.

As we can see (figure 5.3) the car is trying but doesn't come to the exact height. So, the chassis is a bit closer to the road now, then it was driving on ground level. The suspension pulls it down. This means that the controller is not really good.

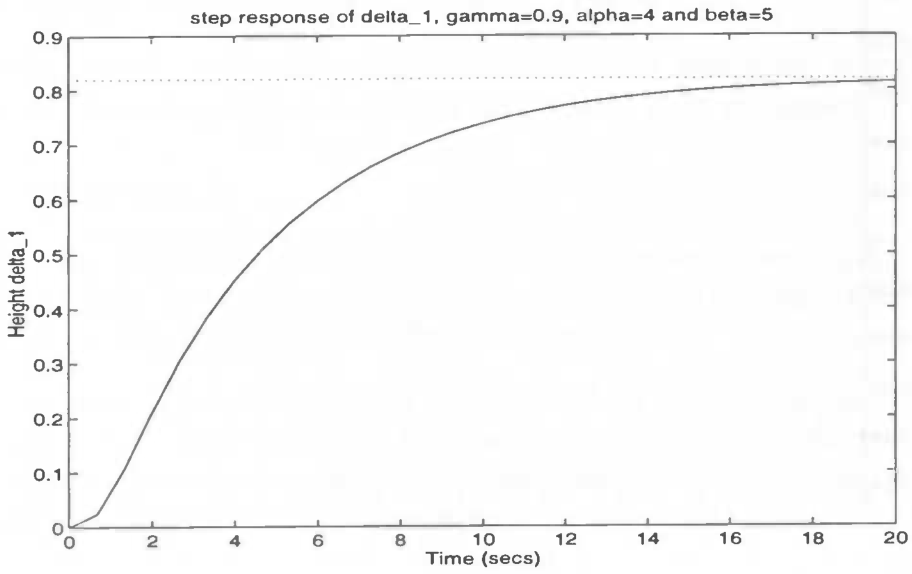


Figure 5.3

### 5.4.3 The Response to Arbitrary Inputs

*A Ribbed Road*

First we'd like to see the response of $\Delta_1$ when the car is driving on a ribbed road. We can model this by assuming that the disturbance is a high frequency sine wave with low amplitude. We chose the function $0.1\sin(20t)$. What we want is that this disturbance is 'ignored', that the car simply drives through not trying to follow this road profile. In the controlled system high frequency disturbances must be suppressed.

The dotted line is the road profile, the solid line is the car response (figure 5.4). As we can see, the car is doing what we expected.

*A Smooth Hill*

Now, what does the car when it goes up a (smooth) hill? Will it simply follow this road, as we expect? Low frequency disturbances must be let through. We model this type of road by a low frequency sine wave with 'high' amplitude, $\sin(0.1t)$.

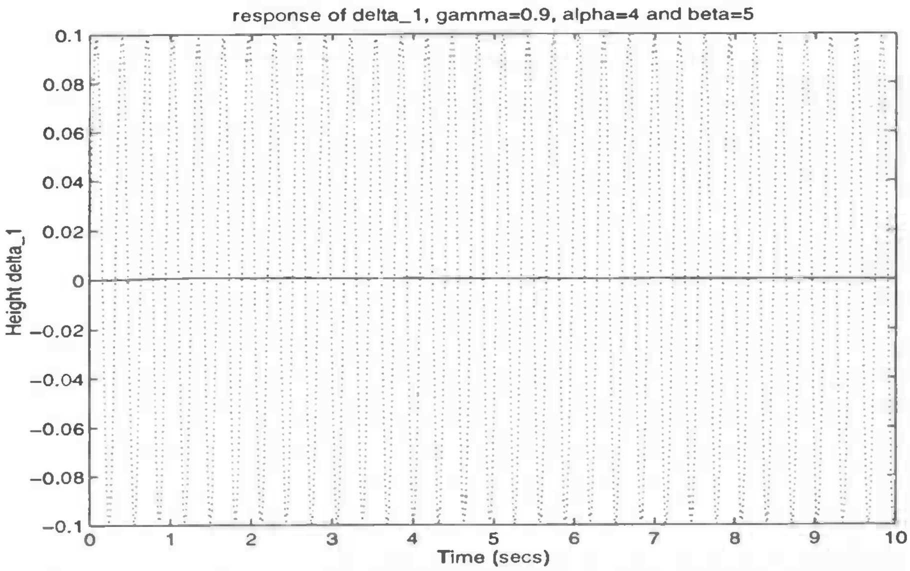


Figure 5.4

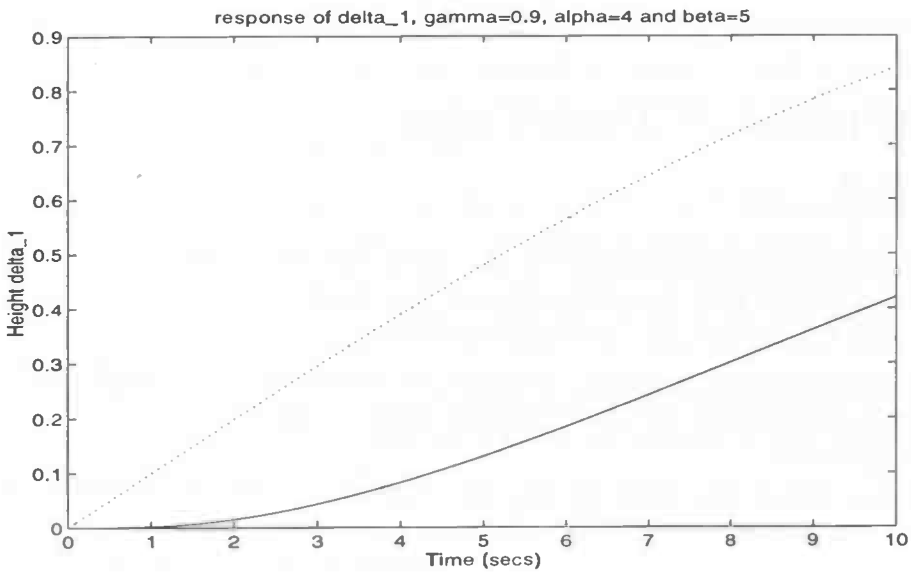


Figure 5.5

The dotted and solid lines are as mentioned above. As we can see (figure 5.5), the car is following the road but not all the way up. He stays behind a little. It's like the suspension is pulling the car down. This is what we have already seen at the step response.

# 6 Conclusions and Future Work

We can now formulate some conclusions on what we've seen in the example. We will also mention some areas of research that remain to be developed.

## 6.1 Conclusions

- For high frequency disturbances the controlled system does what it is supposed to do, it suppresses the disturbance and drives through. This is what the filter in the system laws tells the system to do.
- For low frequency disturbances the system does not respond exactly as we expect. The filter lets these frequencies through so that the system can follow them, but the controlled system does not. The results might get better if the weighting $\beta$ depends om $\omega$ (or rather $\omega^{-1}$).

## 6.2 Future Work

We have seen in section 5 that there are still fields that remain to be explored. Especially regarding computational algorithms.

- We can compute a factorization, there are numerous algorithms already developed, but how can we compute a factorization that immediately satisfies the constraints for a controller to be strictly $\gamma$-contracting and stabilizing (section 4.3)?
- We have not yet considered the implementation of the controller. Is it possible to design the controller in such a way that it can be implemented with passive elements (spring, damper,...)?
- And finally, what will the car in the example do if the weighting depends on $\omega$?

# A $J$-Spectral Factorization

The computation of the $H_\infty$ controller is based on $J$-spectral factorization. We only consider $J$-spectral factorization of para-Hermitian polynomial matrices, because the matrix $Z$ we'll have to factorize always is, $Z(\xi) = M^T(-\xi)\Sigma_\gamma M(\xi)$, where $\Sigma_\gamma$ is a constant diagonal matrix.

A factorization $Z(\xi) = L_2(\xi)L_1(\xi)$ is called a *spectral factorization* if the factors $L_1$ and $L_2$ have disjoint spectra, so if the set of eigenvalues of $L_1$ and $L_2$ are disjoint. Then the factors $L_1$ and $L_2$ are called *spectral factors.*

A factorization $Z(\xi) = P^T(-\xi)JP(\xi)$ is called a *$J$-spectral factorization* if $J$ is a signature matrix and $P$ a square matrix with real coefficients such that det $P$ is Hurwitz. Note that $P$ and $P^*$ have disjoint spectra, since $P$ is Hurwitz. A factorization of the form $Z(\xi) = P(\xi)JP^T(-\xi)$, with $J$ and $P$ as above, is called a *$J$-spectral cofactorization.*

A sufficient condition for the existence of a $J$-spectral factorization is that det $Z$ has no roots on the imaginary axis. A $J$-spectral factorization is *not* unique.

**Theorem A.1 :** *Let the polynomial matrix $P$ be a spectral factor of the para-Hermitian polynomial matrix $Z$ with corresponding signature matrix $J$. All other spectral factors of $Z$ are of the form $UP$, with $U$ unimodular s.t. $U^*JU = J$, $U$ is said to be a $J$-unitary unimodular matrix*

The factorization $Z(\xi) = M^T(-\xi)\Sigma_\gamma M(\xi) = F^T(-\xi)JF(\xi)$ is called a *regular factorization* if the McMillan degree of $F$ is equal to that of col$(M, F)$. In that case the factor $F$ is called a *regular factor.* Note that if $F$ is square and det$F \neq 0$, then the McMillan degree of $F$ is equal to that of col$(M, F)$ if and only if $MF^{-1}$ is a proper rational matrix.

There are several algorithms to compute a $J$-spectral factorization. For instance based on diagonalization, successive factor extraction or the solution of an algebraic Riccati equation (see [4]).

## A.1 Diagonal Reducedness

Often, to apply factorization algorithms on a polynomial (para-Hermitian) matrix $Z(\xi)$, the matrix $Z(\xi)$ must be diagonally reduced.

**Definition A.2 :** Suppose that half the degrees of the diagonal entries of the $n \times n$ para-Hermitian polynomial matrix $Z(\xi)$ are $\delta_1, \delta_2, \ldots, \delta_n$, and define the *diagonal leading coefficient matrix* $Z_D$ of $Z$, if it exists, as

$$Z_D = \lim_{|\xi|\to\infty} E^*(\xi)Z(\xi)E(\xi)$$

where $E$ is the polynomial matrix defined by $E(\xi) =$ diag $(\xi^{-\delta_1}, \xi^{-\delta_2}, \ldots, \xi^{-\delta_n})$. $Z$ is *diagonally reduced* if $Z_D$ is nonsingular.

# B Graphs for Different Values of the Parameters

## B.1 First Results

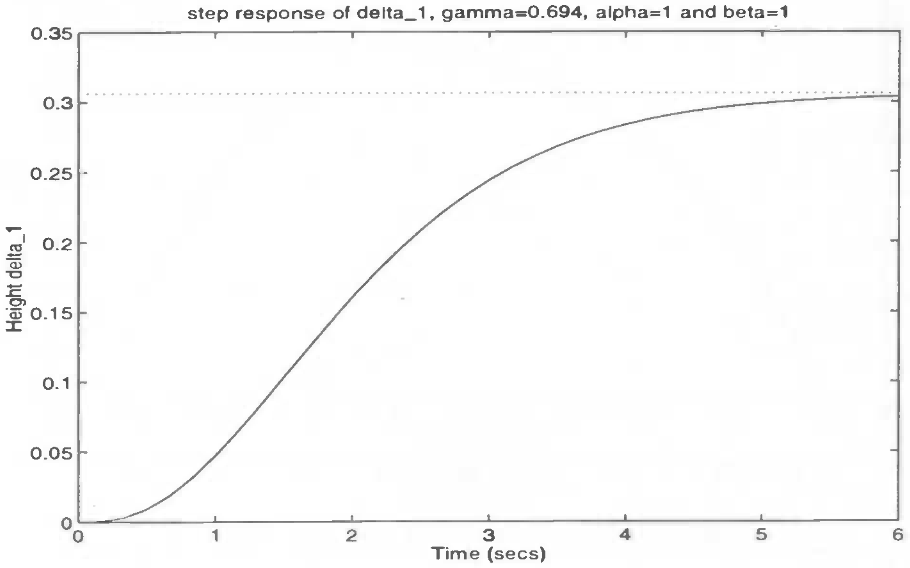


Figure B.1

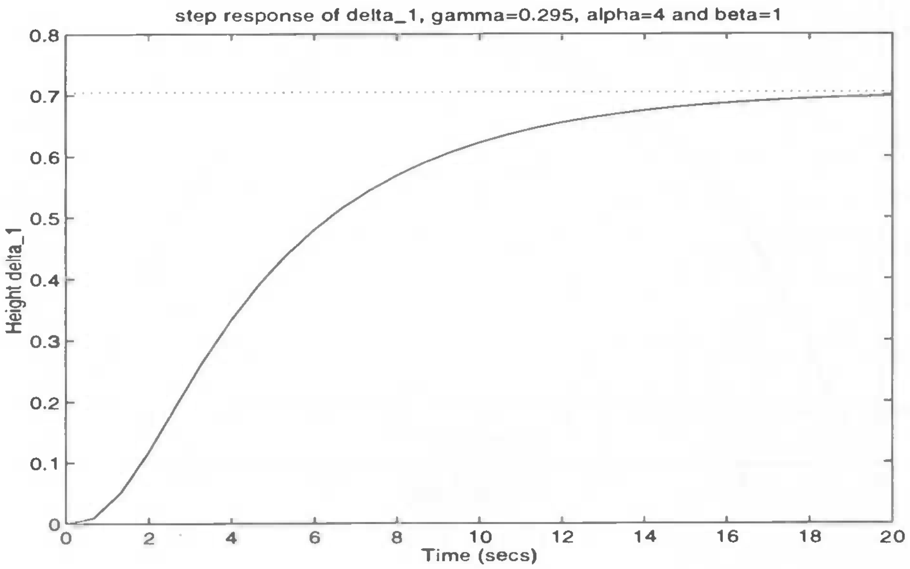


Figure B.2

## B.2 More Step Responses

Increasing $\alpha$ meant better results but longer settling time.

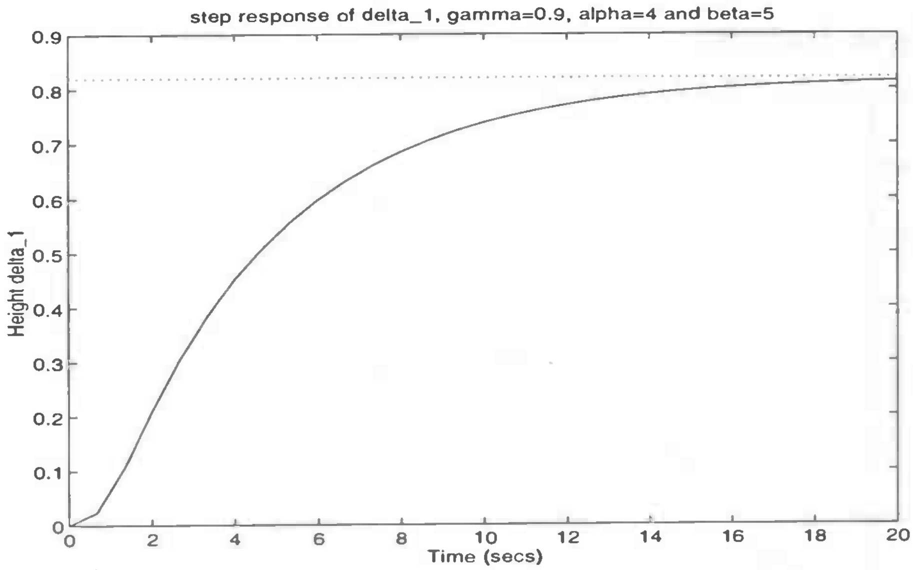


Figure B.3

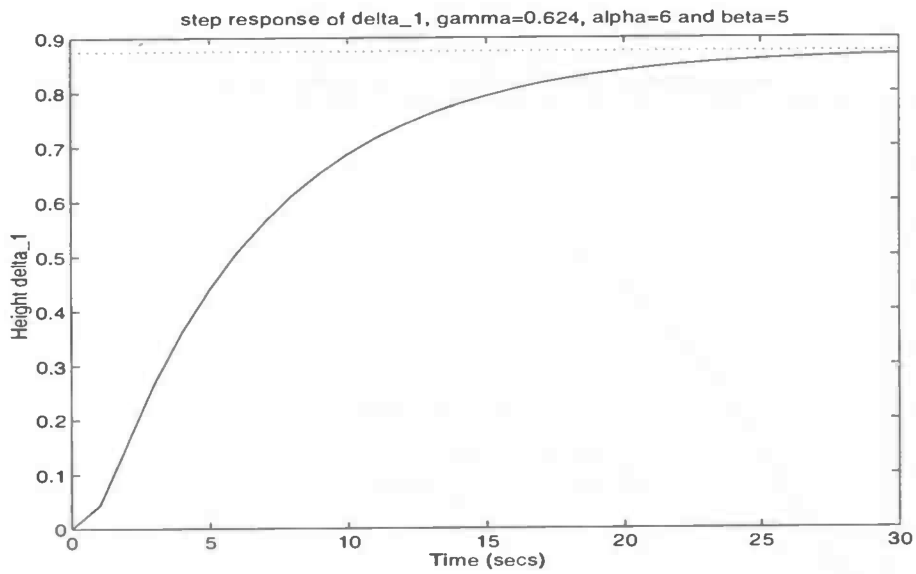


Figure B.4

Increasing $\beta$ also gave better results but also meant increasing $\gamma$.

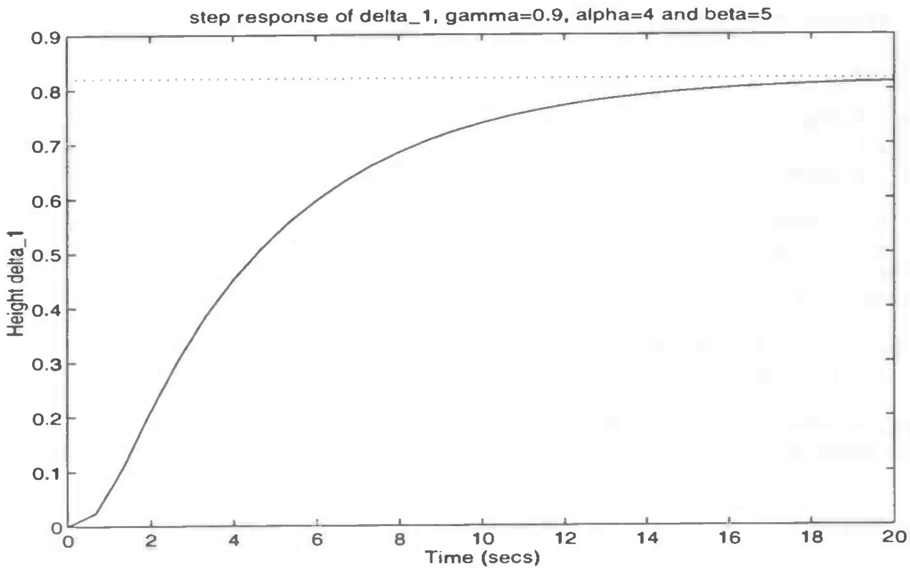


Figure B.5

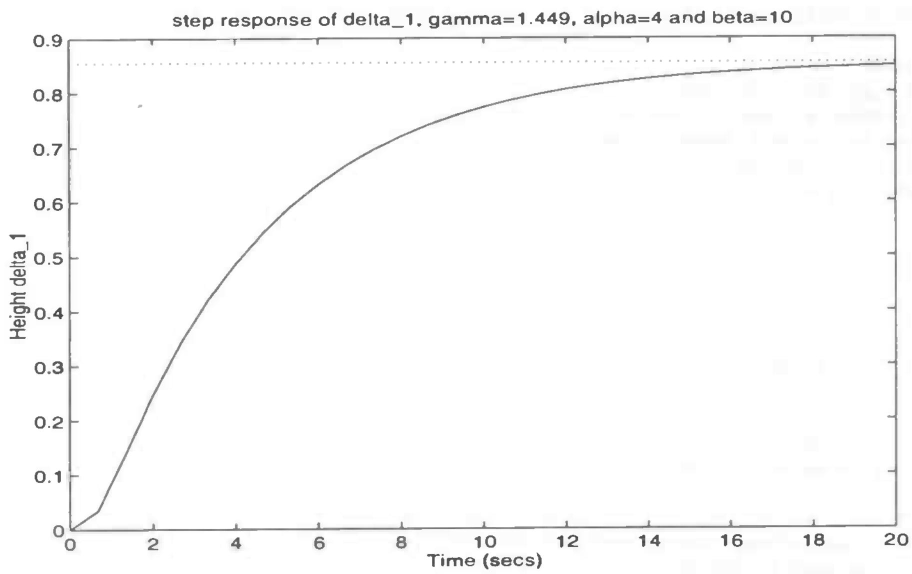


Figure B.6

# C Mathematica Package

```
(* Wed Aug  7 14:42:26 METDST 1996

 The file pick.ma

 model: | z |     | M(d/dt) |
        | d |  =  |    | 1
        | c |     | C(d/dt) |

 M(xi) = |    0      xi^3 |
         |    0      xi^2 |
         |   -1       1   |
         | 1+a*xi     0   |

 Z(xi) = M^T(-xi).| I_3  0   |.M(xi)
                  |  0  -g^2 |

This file computes whether for the given gamma and alpha the
Pick matrix is negative definite.
*)

BeginPackage["Pick`"]

Pick::usage =
"Pick[a,b,g], where a,b,g>0.\n
It computes the Pick matrix associated with R[xi].S[1/b,1/g].Transpose[R[-xi]].
It only works if multiplicity of roots of R[xi].S[1/b,1/g].Transpose[R[-xi]] is 1."

Existence::usage =
"Existence[a,b,g], where a,b,g>0.\n
It computes whether a gamma contracting stabilizing H_inf controller\n
exists for the system with image representation w=M(xi)l,\n
M[xi,a]={ {0, xi^3}, {0, xi^2}, {-1, 1}, {1+a xi, 0} }.\n
Existence makes use of Pick."


Begin["`Private`"]


M[xi_, a_] := {
              {0, xi^3},
              {0, xi^2},
              {-1, 1},
              {1+a xi, 0}
             } (* we want to vary the parameter a *)

R[xi_, a_] := {{1, -xi, 0, 0}, {-a, -1, xi^2 + a*xi^3, xi^2}}
           (* a controllable kernel representation of the system,
              now dual system is x = R'(-xi) y *)

S[b_,g_] := DiagonalMatrix[{1,1,b^2,-g^2}]

Z[xi_,a_,b_,g_] := Transpose[M[-xi,a]].S[b,g].M[xi,a]

G[z_,h_,a_,b_,g_] := R[-z,a].S[1/b,1/g].Transpose[R[-h,a]]

detg[xi_,a_,b_,g_] := Det[ G[-xi,xi,a,b,g] ]

soln[a_,b_,g_] := N[ Solve[detg[xi,a,b,g]==0, xi] ]

X[a_,b_,g_] := xi /. soln[a,b,g]

negeig[a_,b_,g_] := Select[X[a,b,g], (Re[#]<0)&]
```

```
enkelv[a_,b_,g_] := Union[ negeig[a,b,g] ]

multipl[a_,b_,g_] := Map[Count[negeig[a,b,g], #]&, enkelv[a,b,g] ]

fi[z_,h_,a_,b_,g_,r_,s_] := D[G[z,h,a,b,g],{z,r},{h,s}]

phi[z_,h_,a_,b_,g_,mz_,mh_] :=
  Module[{cz=Conjugate[z], F=Table[0,{i,1,mz},{j,1,mh}]},
          For[r=1, r<=mz, r++,
              For[s=1, s<=mh, s++,
                  f = fi[p,q,a,b,g,r,s];
                  F[[r,s]] = f /. {p->cz, q->h}
                 ]
             ];
          Flatten[F,2] (* in het test geval waar F[r,s]={1,1} werkt dit *)
        ]

psi[a_,b_,g_] :=
  Module[{neig,m,l,n,P,eindP},
          neig = enkelv[a,b,g];
          m=multipl[a,b,g];
          l=Length[neig];
          n=Dimensions[G[z,h,a,b,g]][[2]];
          P=Table[0,{i,1,l},{j,1,l}];
          eindP=Table[0, {i,1, n*Sum[m[[i]],{i,1,Length[m]}] }];
          For[i=1, i<=l, i++,
              For[j=1, j<=l, j++,
                  P[[i,j]] = phi[neig[[i]],neig[[j]],a,b,g,m[[i]],m[[j]]]
                 ]
             ];
          P
        ]

Pick[a_,b_,g_] :=
  Module[{neig,m,n,T,P,A,dg,head,index },
          neig = enkelv[a,b,g];
          m = multipl[a,b,g];
          n = Sum[m[[i]],{i,1,Length[m]}];
          T = Table[0,{i,1,n},{j,1,n}];
          P = psi[a,b,g];
          If[Union[m]!={1},
             Return["This problem is not solved by this procedure yet\n
                     Roots = '1' with multiplicity = '2'", neig, m
                   ]
            ];
          A = Table[0,{i,1,Length[neig]}];
          For[i=1, i<=Length[neig],i++,
              A[[i]] = NullSpace[G[-neig[[i]],neig[[i]],a,b,g]];
              If[A[[i]]=={},
                 dg=Chop[G[-neig[[i]],neig[[i]],a,b,g]];
                 A[[i]]=Chop[{-dg[[1,2]],dg[[1,1]]}];
                 If[Union[Chop[dg.A[[i]] ]]!={0},
                    Return["The trick doesn't work here."]
                   ],
                 A[[i]]=First[A[[i]]]
                ]
             ];
          For[i=1, i<=n, i++,
              For[j=1, j<=n, j++,
                  T[[i,j]] = 1/(Conjugate[neig[[i]]]+neig[[j]]) *
                                 Conjugate[A[[i]]].P[[i,j]].A[[j]]
                 ]
             ];
          T
```

```
        ];


Existence[a_,b_,g_] :=
  Module[{T},
         T = Pick[a,b,g];
         If[StringQ[T],
            Return["The problem is not solved with this procedure.\n",T],
            If[Union[Sign[Chop[Eigenvalues[T]]]]=={-1},
               Print["There exists a gamma contracting stabilizing H_inf controller for"];
               Print["gamma = ",g,", alpha = ",a," and beta = ",b],
               Print["There does not exist a gamma contracting stabilizing H_inf controller for"];
               Print["gamma = ",g,", alpha = ",a," and beta = ",b]
              ]
           ]
         ]


End[ ] (* of Private *)

EndPackage[ ]
```

# D MATLAB Macros

## D.1 Computation of the Controller

```
% Tue Aug  6 12:44:29 METDST 1996
%
% The file controller.m, K = controller(a, b, g)

%  model: | z |    | M(d/dt) |
%         | d | =  |         | 1
%         | c |    | C(d/dt) |

%  M(xi) = |    0      xi^3 |
%          |    0      xi^2 |
%          |   -1       1   |
%          | 1+a*xi     0   |

%  Z(xi) = M^T(-xi).| I_2  0      |.M(xi)
%                   |  0  b^2     |
%                   |  0   0  -g^2 |

% Z(xi) = M^T(-xi).S_b,g.M(xi) = | b^2-g^2+g^2*a^2*xi^2       -b^2        |
%                                |          -b^2         -xi^6+xi^4+b^2 |
%

function K = controller(a, b, g)

DegZ=[2 6];
Z=[b^2-g^2 -1*b^2  0 0  g^2*a^2 0  0 0  0 0  0 0  0 0;
   -1*b^2 b^2  0 0  0 0  0 0  0 1  0 0  0 -1 ];

[F,degF,J,degJ,muu,n] = jsp(Z,DegZ);

% First test whether the first row of the spectral factor makes a
% g-contracting stabilizing H_inf controller
if ~correct(F, degF, J, degJ, a)
  disp(' ');
  error('This factorization does not lead to a g-contracting stabilizing controller.')
end

K = F(1,:);

if (degF == 3 & K(8)~=0)
  K = K/K(8);
  if (K(3)~=0 | K(5)~=0 | K(7)~=0)
    error('This factorization is not suitable as a controller.');
  else
    K(3)=K(4);
    K(4)=K(6);
  end;
else
  error('This factorization is not suitable as a controller.');
end

K = K(1:4);
% the controller now is u=-Kx

----------------------------------------------------------------------------

% Tue Aug  6 13:47:50 METDST 1996

% The file correct.m
% It tests whether the first row of the spectral factor makes a
% g-contracting stabilizing H_inf controller
```

```
function corr=correct(F, degF, J, degJ, a)

corr = 1;              % initialisation

% I the factorization M'(-xi).S_g.M(xi)=F'(-xi).J.F(xi) exists
% and will be correct if J=diag([1,-1])
if (J~=diag([1,-1]) | degJ~=0)
  corr = 0;
  disp(' ');
  disp('J~=diag([1,-1]), so this factorization is not suitable');
end

%II F(xi) must be Hurwitz
if ~hurwitz(F, degF)
  corr = 0;
  disp(' ');
  disp('The factorization is not Hurwitz');
end

%       M
%III n( ) = n(F)  <=>  M.inv(F) proper  <=>   deg(M)<=deg(F)
%       F
degM=3;
if degF<degM
  corr = 0;
  disp(' ');
  disp('The factorization is not suitable,M.inv(F) is not proper');
end

%IV [D;K]=[D;F+] must be Hurwitz
K = F(1,:); degK = round( max(find(K))/2 ) -1;
D = [1 0  a 0  zeros( size(K(:, 5:length(K) )) )]; degD = 1;
if ~hurwitz([D;K], max(degD,degK))
  corr = 0;
  disp(' ');
  disp('[D;K] is not Hurwitz');
end
----------------------------------------------------------------------------

% Tue Aug  6 13:24:39 METDST 1996

% The file hurwitz.m

% This function computes the eigenvalues of the polynomial matrix A
% and tests whether they lie in the open left half plane or not.
% it returns TRUE or 1 if they do, so if the matrix is Hurwitz.

function hurw=hurwitz(A, degA)

hurw = 1;      %initialisatie

[V,eig] = poleig(A, degA);
eig=diag(eig);
for j = 1:length(eig)
  if ( isinf(eig(j)) | isnan(eig(j)) )
    eig(j)=0;
  end
end

eig=eig( min(find(eig)):max(find(eig)) );
for j = 1:length(eig)
  if sign(real(eig(j)))~=-1
    hurw = 0;
```

```
  end
end
```

## D.2 Simulation

```
% Sat Tue  6 12:44:29 METDST 1996
%
% The file state.m

%  model: | z |    | M(d/dt) |
%         | d | =  |         | l
%         | c |    | C(d/dt) |

%  M(xi) = |     0      xi^3 |
%          |     0      xi^2 |
%          |    -1       1   |
%          | 1+a*xi      0   |

%  Z(xi) = M^T(-xi).| I_3  0  |.M(xi)
%                   |  0 -g^2 |

% Z(xi) = M^T(-xi).S_g.M(xi) = | 1-g^2+g^2*a^2*xi^2       -1        |
%                              |          -1         -xi^6+xi^4+1 |
%

% in state space:
% d/dt x = A x + B u + E d
%      y = C x + D u + F d    where F = 0
% or   y = C1 x
% where
% x = [v, z1, d/dt z1, d2/dt2 z1]
% u = d3/dt3 z1

function [A,B,C,E,F] = state(a)

path(path,'/home/civil/users/paula/afstuderen/kwaker/version2');

A = [-1/a 0 0 0;
      0 0 1 0;
      0 0 0 1;
      0 0 0 0 ];

B = [ 0 0 0 1 ]';

C = [ 0 1 0 0 ];

E = [ 1/a 0 0 0 ]';

F = [0];
-----------------------------------------------------------------------

% Wed Aug  7 09:55:46 METDST 1996
%
% The file bodes.m, bodes(a,b,g,w).
% It computes the bode plots of the system,
% for the system variable delta_1
% The argument w is a frequency vector which
% is optional and can be omitted.

function bodes(a,b,g,w)
```

```
met = 0;                  % default there is no frequency vector
if nargin < 3
  disp(' ');
  error('Not enough input arguments.')
elseif nargin == 4
  met = 1;
elseif nargin > 4
  disp(' ');
  error('Too many input arguments.')
end

[A,B,C,E,F] = state(a);
K = controller(a,b,g);

% now, the closed loop system is
% d/dt x = (A - B K) x + E d
%       y = C x
%
% where
% x = [v, z1, d/dt z1, d2/dt2 z1]
% u = d3/dt3 z1


if met
  [mag,phase] = bode(A-B*K,E,C,F,1,w);
else
  [mag,phase,w] = bode(A-B*K,E,C,F,1);
end

[m,p] = bode(1,[a 1],w);

% As a first result we are interested in the bode plot of z1
% together with the transfer function of d->v

subplot(211), semilogx(w, 20*log10(mag)), grid, hold
subplot(211), semilogx(w, 20*log10(m),'r--')
xlabel('Frequency (red/sec)'), ylabel('Gain dB')
title(['Bode plot of delta_1, gamma=',num2str(g),', alpha=',num2str(a),' and beta=',num2str(b)])

subplot(212), semilogx(w, phase), grid, hold
subplot(212), semilogx(w, p,'r--'),
xlabel('Frequency (red/sec)'), ylabel('Phase deg')

-----------------------------------------------------------------------------

% Tue Aug  6 15:13:23 METDST 1996
%
% The file stap.m, stap(a,b,g,T)
% It simulates the step response of the system,
% a step is placed on the input d (the disturbance).
% The argument T is a time vector which is optional
% and can be omitted.

function stap(a,b,g,T)

met = 0;                  % default there is no time vector
if nargin < 3
  disp(' ');
  error('Not enough input arguments.')
elseif nargin == 4
  met = 1;
elseif nargin > 4
  disp(' ');
  error('Too many input arguments.')
```

```
end

[A,B,C,E,F] = state(a);
K = controller(a,b,g);

% now, the closed loop system is
% d/dt x = (A - B K) x + E d
%      y = C x
%
% where
% x = [v, z1, d/dt z1, d2/dt2 z1]
% u = d3/dt3 z1

if met
  [Y, X] = step(A-B*K, E, C, F, 1, T);
else
  [Y, X, T] = step(A-B*K, E, C, F, 1);
end

% As a first result we are interested in the step response of z1
% plot(T, Y)
if met
  step(A-B*K, E, C, F, 1, T);
else
  step(A-B*K, E, C, F, 1);
end

xlabel('Time (secs)'), ylabel('Height delta_1')
title(['step response of delta_1, gamma=',num2str(g),', alpha=',num2str(a),' and beta=',num2str(b)])

-------------------------------------------------------------------------------

% Fri Aug  9 14:09:26 METDST 1996
%
% The file sim.m, sim(a,b,g,U,T,X0)
% It simulates the time response of the system to the input U
% (the disturbance d), a 1-column vector with length(T) rows.
% The time vector must be regularly spaced.
% The initial condition X0 is optional and can be omitted.

function sim(a,b,g,U,T,X0)

met = 0;                  % default there is no initial condition
if nargin < 5
  disp(' ');
  error('Not enough input arguments.')
elseif nargin == 6
  met = 1;
elseif nargin > 6
  disp(' ');
  error('Too many input arguments.')
end

[A,B,C,E,F] = state(a);
K = controller(a,b,g);

% now, the closed loop system is
% d/dt x = (A - B K) x + E d
%      y = C x
%
% where
% x = [v, z1, d/dt z1, d2/dt2 z1]
% u = d3/dt3 z1

if met
```

```
  [Y, X] = lsim(A-B*K, E, C, F, U, T, X0);
else
  [Y, X] = lsim(A-B*K, E, C, F, U, T);
end

% As a first result we are interested in the response of z1
plot(T, Y, T, U,':')

xlabel('Time (secs)'), ylabel('Height delta_1')
title(['response of delta_1, gamma=',num2str(g),', alpha=',num2str(a),' and beta=',num2str(b)])
```

---

```
% Thu Aug  8 10:26:17 METDST 1996
%
% The file meerdere.m, meerdere(alpha,beta,gamma,opt,w),
% it produces for the combinations of alpha(i), beta(i) and gamma(i),
% if possible the graphs of the step response (opt=0) or the bode plot
% (opt=1). If you don't give an option the bode plot will be given
% (default opt=1). Also, it is optional to give in a time or frequency
% vector (resp. for step response or bode plot).
% It is best not to use to many parameters at once, the procedure will
% stop as soon as one combination does not give a correct controller.

function meerdere(alpha,beta,gamma,opt,w)

met = 1;                % initialisatie, with met we check whether there
                        % is a time or frequncy vector
if nargin < 3
  disp(' ');
  error('Not enough input arguments.')
elseif nargin == 3
  opt=1;                % default setting
  met = 0;              % there is no time or frequency vector
  elseif nargin == 4
    if length(opt) > 1
      w = opt;
      opt =  1;         % there is a time or frequency vector
    else
      met = 0;          % there is no time or frequency vector
    end
  elseif nargin > 5
    disp(' ');
    error('Too many input arguments.')
end

if ( length(alpha)~=length(beta) | length(alpha)~=length(gamma) )
  disp(' ');
  error('The parameters should be vectors of the same length. If you want the bode / step response plots
       at all combinations you should use: combi(alpha,beta,gamma,opt)')
end

N = length(alpha);      % to number the figures

for k=1:N               % all existing figures 1..N will be overwritten
  figure(k)
  clf
end

if ~met
  for k=1:N
    a = alpha(k);
    b = beta(k);
    g = gamma(k);
    figure(k);
```

```
    if opt
      bodes(a,b,g);
    else
      stap(a,b,g);
    end
  end
else
  for k=1:N
    a = alpha(k);
    b = beta(k);
    g = gamma(k);
    figure(k);
    if opt
      bodes(a,b,g,w);
    else
      stap(a,b,g,w);
    end
  end
end
```

---------------------------------------------------------------------------

```
% Thu Aug  8 10:26:17 METDST 1996
%
% The file combis.m
% it produces for all the combinations of alpha, beta and gamma, if
% possible the graphs of the step response (opt=0) or the bode plot
% (opt=1). If you don't give an option the bode plot will be given
% (default opt=1).
% It is best not to use to many parameters at once, the procedure will
% stop as soon as one combination does not give a correct controller.

function combis(alpha,beta,gamma,opt)

N = length(alpha)*length(beta)*length(gamma);  % to number the figures
n = 1;                                           %initialisatie

if nargin==3
  opt=1;               % default setting
end

for k=1:N              % all existing figures 1..N will be overwritten
  figure(k)
  clf
end

for k=1:length(alpha)
  a = alpha(k);
  for l=1:length(beta)
    b=beta(l);
    for m=1:length(gamma)
      g=gamma(m);
      figure(n);
      n=n+1;
      if opt
        bodes(a,b,g);
      else
        stap(a,b,g);
      end
    end
  end
end
```

# References


[1] A.C. Antoulas and J.C. Willems, *Mathematical Models of Systems.* DISC Course Notes, part 1, 1995.

[2] I. Gohberg, P. Lancaster and L. Rodman, *Matrix Polynomials*, New York: Academic Press, inc. 1982

[3] H. Kwakernaak, "MATLAB Macros for Polynomial $H_\infty$ Control System Optimization", *Report*, Dept. of Applied Mathematics, University of Twente

[4] H. Kwakernaak and M. Šebek, "Polynomial $J$-Spectral Factorization." In: *IEEE Trans. on Automatic Control*, Vol. 3, no. 2, February 1994

[5] H.L. Trentelman and J.C. Willems, "$H_\infty$ Control in a Behavioral Context, part 1: the full information case." preprint July, 1996.

[6] J.C. Willems and H.L. Trentelman, "On Quadratic Differential Forms." preprint April 25, 1996.