ISSN 0280–5316
ISRN LUTFD2/TFRT--5550--SE

Integrated Control and Diagnosis Using 4 DOF Controllers

Michael Templin

Department of Automatic Control
Lund Institute of Technology
February 1996

Department of Automatic Control **Lund Institute of Technology** P.O. Box 118 S-221 00 Lund Sweden	*Document name* MASTER THESIS
	Date of issue February 1996
	Document Number ISRN LUTFD2/TFRT--5550--SE
Author(s) Michael Templin	*Supervisor* Karl-Erik Årzén, Bo Bernhardsson
	Sponsoring organisation TFR project ”Integrated Control and Diagnosis”

Title and subtitle
Integrated Control and Diagnosis Using 4 DOF Controllers

Abstract

The four degree of freedom (4 DOF) controller provides an approach for design of integrated control and diagnosis systems. The problem is formulated on standard interconnection form and can be attacked with different control design methods, e.g., H_∞ or H_2 control.

This thesis has investigated the 4 DOF controller in combination with H_∞ and μ-design. Two problems have been studies, a literature example and diagnosis of a simulated inverted pendulum. Special focus has been put on the selection of weighting filters.

Key words

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title 0280–5316	*ISBN*

Language English	*Number of pages* 59	*Recipient's notes*
Security classification		

Acknowledgments

I would like to thank my supervisors *Karl-Erik Årzén* and *Bo Bernhardsson* for all help and guidance they have provided. A special thank to *Mats Åkesson* for nice cooperation.
Finally I would like to thank the staff at the department for making my time here very pleasant.

Lund, February 1996

Michael Templin

1 Introduction

Diagnosis is becoming a more and more important part of control systems. One way of designing the diagnosis system is to use the four degree of freedom controller. It is a controller with an extra output - the alarm output. If the process to be controlled is uncertain, i.e. there is a mismatch between the model and reality, the controller has to be designed to be robust against this uncertainty. One design method that incorporates uncertainties is H_∞-synthesis.

In the design of a control and diagnostic module some tradeoffs have to be done. For example diagnostics and control performance have to be traded off against each other. The goal of this master thesis is to:

- Get an understanding of how H_∞-synthesis works.
- Explain the steps of a design of a control and diagnostic module using the four degree of freedom controller and H_∞-synthesis.
- Elucidate the tradeoffs taken.

The designs in this thesis have been made in MatLab and the toolbox μ-tools. All simulations have been made in Simulink.

Chapter 2 contains a short introduction to on-line monitoring and diagnosis. In Chapter 3 the four degree of freedom controller and a way of setting up the control and diagnosis problem as a standard problem are described. Chapter 4 contains a short presentation of H_∞- and μ-synthesis. A design example is thoroughly discussed in Chapter 5. In Chapter 6 some designs on the inverted pendulum are presented. Appendix A is a short description of the μ-tools commands used in this thesis, and an example of a complete design is shown in Appendix B.

2 On-Line Monitoring and Diagnosis

On-line monitoring and diagnosis is becoming an increasingly important part of modern control systems. One reason for this is that the demands on reliability and safety are getting higher.
A number of different approaches have been suggested and research within the area is performed in several different research communities, for example control engineering, artificial intelligence, signal processing, chemical engineering and mathematical statistics. Monitoring and diagnosis methods are either *symptom-* or *model-based.*

2.1 Symptom-based diagnosis

The symptom-based approaches are based on direct associations between faults and their symptoms. The associations can be represented as rules, tables, or decision trees. The knowledge underlying the associations is often of a purely heuristic nature, gathered from the experience of a human expert in the problem domain, for example the process operator or the process designer.

Symptom-based diagnosis has several drawbacks. The knowledge acquisition process is not easy. It is difficult to get a complete system without knowledge gaps. The diagnosis system will thus inherit the misconceptions of the human expert and suffer from the same shortcomings. Also, a complete system can contain a large number of rules, often up to several thousands. The maintenance of a system like this is quite difficult. A major reason for this is that the process structure - the process components and their interconnections - not is explicitly represented in the diagnosis module. Instead, information about a specific process component and its connections may be spread out over a number of rules. Small changes in the process can therefore influence very large parts of the module.

2.2 Model-based diagnosis

Model-based diagnosis is based on a model of the process. The basic idea is to use the model to predict the normal, fault-free behaviour (the nominal behaviour) of the process. A residual is created by comparing the signals from the process model with the observed signals. Deviations in the residuals indicate faults. The situation is shown in Figure 1.

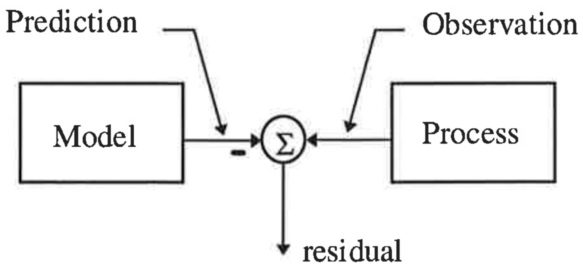


Figure 1: The general idea behind model based diagnosis

The model-based diagnosis techniques have their origin in control engineering and Artificial Intelligence. Within control engineering, on-line diagnosis goes under the name of Fault Detection and Isolation (FDI). The methods used are based on the analytical redundancy inherent in dynamic process models on for example state-space form.

The task of a model-based diagnosis method is to be sensitive to faults at the same time as it is insensitive - robust - to some different non-ideal circumstances in the actual process. These circumstances include modeling errors between the actual system and the mathematical model, disturbances and system and measurement noise. The situation is depicted in Figure 2. The modeling errors and the noise are often combined into a vector of unknown inputs according to Figure 3.

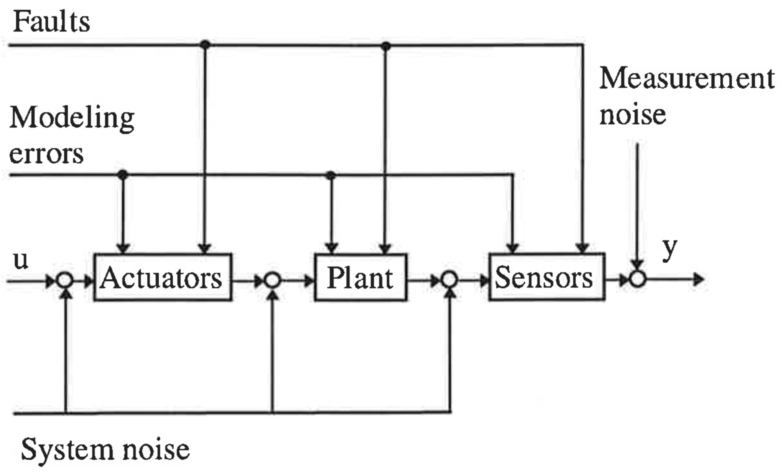


Figure 2: Representation of a system with noise, faults and modeling errors.

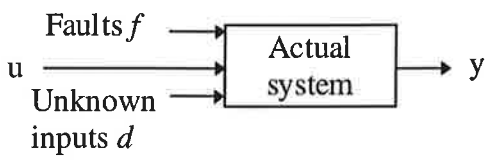


Figure 3: Simplified system representation

The three main approaches that are used in FDI are

- observers
- parity space equations, and
- parameter identification

The basic idea behind the observer approach is to reconstruct the output of the system from the measurements using observers or Kalman filters, where the estimation error or innovation is used for the detection and isolation of the faults. The basic configuration is shown in Figure 4.

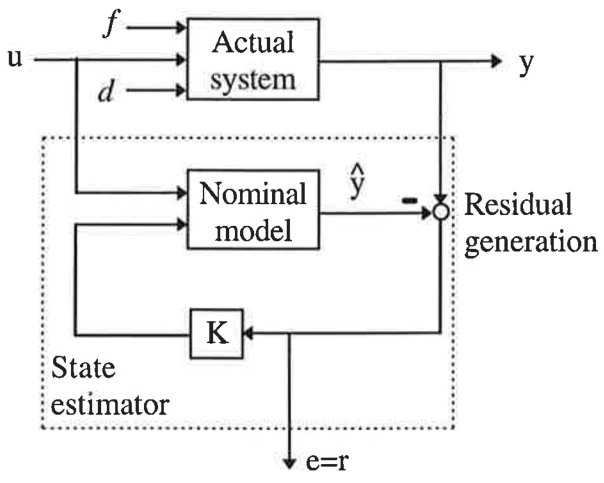


Figure 4: Residual generation through state estimation

Several alternative approaches exist. It is possible to use linear or non-linear, full or reduced-order observers in the deterministic setting or Kalman filters in the stochastic setting. The approaches also differ with respect to the number of process outputs driving the estimator. One can use only one output (the most reliable) to drive the estimator, or the full output vector. It is also possible to use a set of estimators, an estimator bank, where the different estimators are driven by different subsets of the output vector.

The parity space approach is based on calculation of the so called parity equations. These are formed from the equations in the dynamic state space models. It has been shown that the parity space approach is equivalent to the observer approach when the appropriate filters are used.

The parameter identification methods are based on on-line estimation of process parameters through recursive identification. It is suitable for detection of faults that are seen as changes in physical parameters such as friction, mass, resistance, capacitance, etc.

All three approaches mentioned above can be summarized as in Figure 5. Based on a priori knowledge and experience of the real process the following models are used:

- a model of the nominal process
- a model of the observed process
- models of the faulty process.

Since all methods rely on the detection of changes in comparison to the normal status the nominal model must be known and tracked with high precision. The models of the faulty process show the effects of the faults on the analyzed quantities. These effects are called *fault signatures*.

Basically, there are two different ways of generating fault-accentuated signals using analytical redundancy. One way is to use parity checks or observer schemes. These two methods all use state estimation techniques. The other way is parameter estimation. The resulting signals are used to design decision functions as norms or likelihood functions.

The basis for the decision on if there is a fault or not is the fault signature. In most applications the FDI process is completed when the fault location and time are identified. Sometimes it may be important to gain more insight about the situation than is given by just fault location and time. For this purpose deeper knowledge about the nature of the process, for example degree of aging, operational environment, used tools, previous faults and history of operation and maintenance. This task is therefore often tackled with the aid of an expert system.

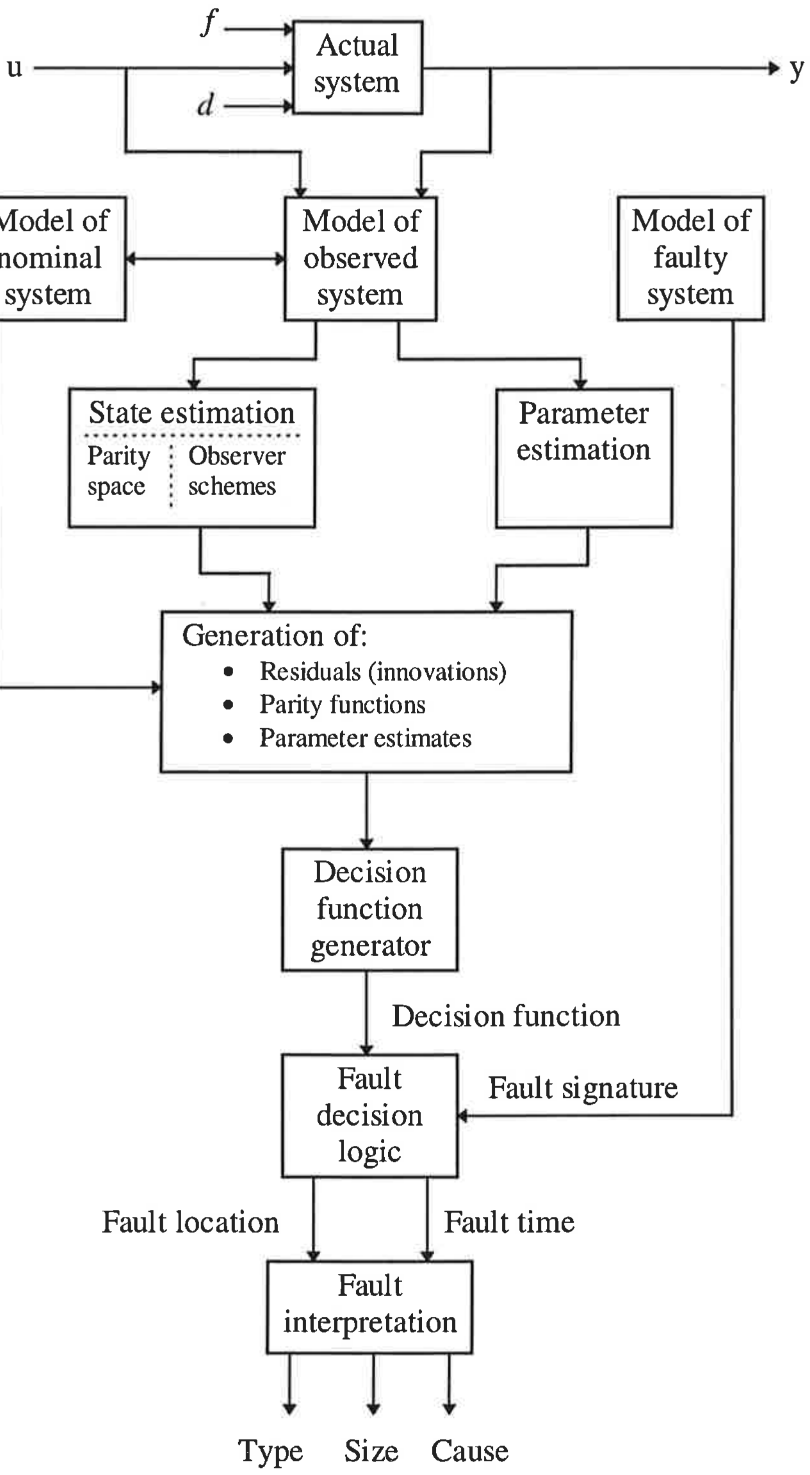


Figure 5: General architecture of a diagnosis system.

The approaches also differ with respect to which types of faults they are best suited for. Faults differ with respect to where in the process they occur. For example, a fault may occur in the actuators, in the process components or in the sensors. Faults can also be of different type. An additive fault is a simple addition of an offset to a sensor or actuator signal. In a multiplicative fault some process signal is multiplied by a fault signal. Parameter identification methods are claimed to be especially suitable for multiplicative faults and additive faults on the input and output signal. It can also detect very small changes which makes it possible to handle slowly developing faults. Observer based methods are claimed to be best suited for additive faults.

A drawback with most of the work in FDI is that it does not consider the interaction between the feedback control algorithm and the diagnosis algorithm. The approach taken in the 4 DOF (4 degree of freedom) controller method is to view the design of the controller and the design of the diagnosis method simultaneously. This is done by translating the entire problem into a multi-variable control problem. Using this formalism modern control design methods developed for design of robust controllers can be used for an integrated design of both controller and diagnosis system. This is the topic of the remaining chapters of the report.

3 Four degree of freedom controllers.

The last twenty years a lot of work has been done on designing filters for monitoring and detecting system faults. Most of the papers presented have been focused on how to design three different modules: a control module, a diagnostic module and a reconfiguration module. When the diagnostic module detects a fault, the reconfiguration module computes a strategy for the control module, in order to achieve reliable control performance under changing conditions. One significant problem with the methods presented in the papers is that they all design the control and design modules independently, neglecting the interaction between them. In addition, the methods are most often not applicable to plants with uncertainty, since they are unable to incorporate norm bounded uncertainties, which are common in control synthesis.
To avoid the first problem Jacobson and Nett introduced the four degree of freedom, or four parameter controller, in [7]. This controller makes simultaneous design of the two modules possible. In [8] Tyler and Morari showed by rewriting the system that the four degree of freedom controller is a special case of the general interconnection structure used by most modern control synthesis methods. Thus it is possible to use results from H_∞-, H_2- and μ-synthesis methods to design the integrated control and diagnostic module.
In the same paper Tyler and Morari showed that designing an optimal control module, and then designing an optimal diagnostic module, is equivalent to designing the two modules simultaneously, if a nominal model (without uncertainty) is considered. If, however, the model contains uncertainty, the two models should be designed simultaneously, in order to take the interaction between the modules into consideration.

3.1 The general idea

A general multi-input, multi-output system **G** can be written as:

$$\begin{bmatrix} z \\ y \end{bmatrix} = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix} \begin{bmatrix} d \\ u \end{bmatrix}$$

where d represent uncontrolled inputs, u controlled inputs, z unused outputs (to be minimized, for example control errors) and y the outputs used by the control algorithm. The uncontrolled input d is

$$d = \left[f^T, \quad n^T\right]^T$$

where f is process faults, such as errors in actuators or sensors, or in the process itself and n is process and measurement noise and different disturbances (for example load disturbances).

In controllers as we are used to see them, we have two degrees of freedom (thus called two degree of freedom or two parameter controller, 2DOF). One is used to specify the feedforward of reference signals, and the other to specify the feedback from the system outputs. The result is a control signal to stabilize the system. In a four degree of freedom (4DOF) controller another output is added: the alarm or diagnostic output a. This means that we get two new parameters. These parameters say how much of and how the system measurements and reference signals go into the alarm signal. The resulting controller K can be written as:

$$\begin{bmatrix} a \\ u \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} \begin{bmatrix} r \\ y \end{bmatrix}$$

If K_{11} and K_{12} of the 4DOF controller are equal to zero, so is the alarm signal a, and the controller is turned into a 2DOF controller. This shows that the 4DOF is a direct extension of the 2DOF.
The system and controller then look like

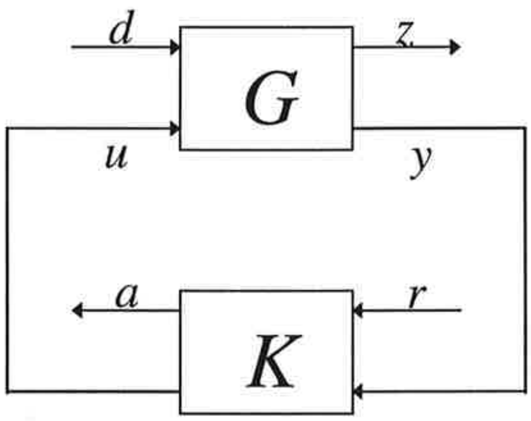


Figure 6: Process and four degree of freedom controller.

3.2 Stability properties

The diagnostic signal a is not part of the closed loop, and does not affect the closed loop stability of the system. Of course K_{11} and K_{12} should be chosen stable. The feedforward term K_{21} also has no effect on the closed loop stability. Hence it is K_{22} that determines the stability.

If a nominal plant G is stabilized by a 4DOF controller K, the perturbed plant G_Δ is stable if and only if

$$\begin{bmatrix} 0 & 0 \\ 0 & K_{22} \end{bmatrix}$$

is a stabilizing controller for G_Δ. This means that stability robustness with the four parameter controller can be analyzed using standard stability robustness tests.

3.3 Tradeoffs

The signal a is used to detect and isolate different faults that can occur in the process. In Figure 7 input faults (actuator faults) are modeled by f_i and output (sensor) faults by f_o. Except for the faults there are input and output noises (n_i and n_o respectively) and a disturbance w.

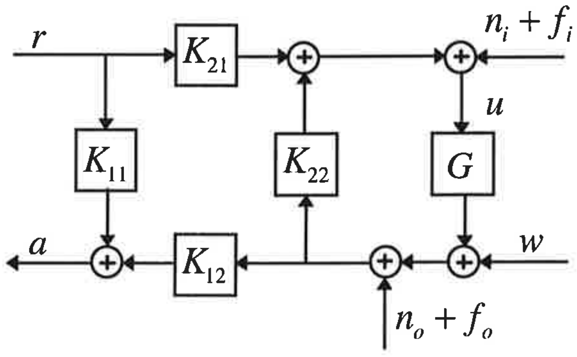


Figure 7: Plant consisting of the process G and a four degree of freedom controller and noise, disturbances and faults.

The diagnostic objective is to track or identify the faults with the diagnostic signal a. This also means that all other system inputs shall be rejected at a. With the transfer function to the signal x from the signal y written as T_{xy}, the control and diagnostic objectives can be summarized as :

- Achieve closed loop stability, considering plant uncertainties.
- Make $T_{yr} - I$, T_{yn}, T_{yf} and T_{yw} small in some sense.
- Make $T_{af} - I$, T_{an}, T_{aw} and T_{ar} small in some sense.

These objectives have to be satisfied simultaneously, and are constrained by the size of T_{ur}, T_{un} and T_{uw}.

3.3.1 C-parametrization

All stabilizing four parameter controllers K for the plant G can be parametrized in a simple way. For a stable plant, one can do as in Figure 8. The left part of the figure is the controller, and the right part the plant. All the C_{ij} are stable transfer matrices and can be called "free parameters". The stable transfer matrix C is defined as:

$$C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

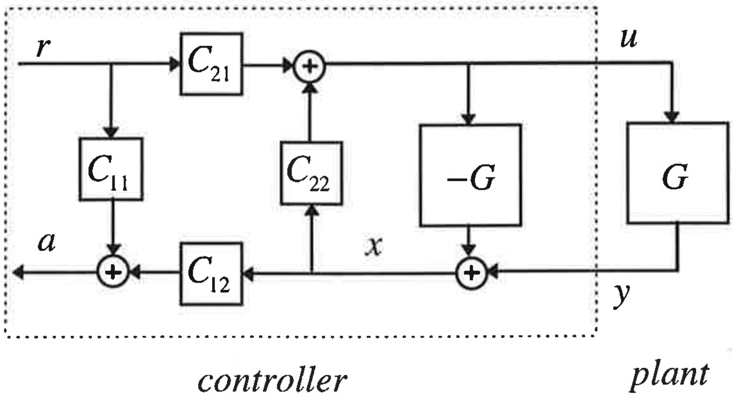


Figure 8: The C-parametrized controller and the (stable) plant it is designed for.

For every stable C there is a corresponding K for G and vice versa. Formulas that relate K and C are given by:

$$K = \begin{bmatrix} C_{11} - C_{12}G(I + C_{22}G)^{-1}C_{21} & C_{12}(I + GC_{22})^{-1} \\ (I + C_{22}G)^{-1}C_{21} & (I + C_{22}G)^{-1}C_{22} \end{bmatrix}$$

$$C = \begin{bmatrix} K_{11} + K_{12}(I - GK_{22})^{-1}GK_{21} & K_{12}(I - GK_{22})^{-1} \\ (I - K_{22}G)^{-1}K_{21} & (I - K_{22}G)^{-1}K_{22} \end{bmatrix}$$

In the C-parametrization it is clearly seen that any stabilizing four parameter controller contains an implicit internal model of the plant it is stabilizing (the block -G in Figure 8). The model is implicit in that it need not be realized, and run in parallel with the plant, but is contained in the four parameter controller.
When there is no uncertainty (i.e. no modeling errors, noise, disturbances or faults) the residual x in Figure 8 is zero, and there is no feedback or feedforward from x.

In the presence of uncertainty, x is no longer zero, and there is both feedforward and feedback. This can be viewed as a means of deducing uncertainties.

3.3.2 The nominal case

Using the definitions

$$S_o = (I - GK_{22})^{-1} \quad H_o = -GK_{22}S_o = -S_oGK_{22}$$
$$S_i = (I - K_{22}G)^{-1} \quad H_i = -K_{22}GS_i = -S_iK_{22}G$$

where S_o and H_o are the output and complementary output sensitivity transfer matrices, and S_i and H_i the input and complementary input sensitivity transfer matrices, the notation will be simpler. These definitions lead to the relations

$$H_o + S_o = I, \; H_i + S_i = I$$

which indicate that the transfer matrices are not independent.
Table 1 shows expressions for the nominal closed loop transfer matrices in terms of K and C.

Table 1: Nominal closed loop transfer matrices

T_{xy}	K-form	C-form
T_{yr}	S_oGK_{21}	GC_{21}
T_{yn_o}	$-H_o$	GC_{22}
T_{yn_i}	S_oG, GS_i	$G(C_{22}G + I)$
T_{yw}	S_o	$GC_{22} + I$
T_{ar}	$K_{11} + K_{12}T_{yn_i}K_{21}$	C_{11}
T_{an_i}	$K_{12}GS_i$	$C_{12}G$
T_{an_o}, T_{aw}	$K_{12}S_o$	C_{12}

- Since $T_{af_o} = T_{an_o} = T_{aw}$, sensor fault diagnosis has to be done on differences in frequency content of the fault, noise and disturbance. This means that sensor diagnostic performance and noise and disturbance rejection have to be traded off against one another.
- Since $T_{af_i} = T_{an_i}$ the limitations on the sensor fault diagnosis also go for the actuator fault diagnosis.

- Since $T_{af_i} = K_{12}GS_i = K_{12}S_oG = T_{af_o}G$, sensor and actuator diagnostic performance have to be traded off against one another.
- Since $T_{af_i} = C_{12}G$, with C_{12} stable, the right half plane zeros of G must be in T_{af_i} in a stable closed loop system. If G is non-minimum phase, this imposes limitations on the achievable actuator diagnostics.

3.3.3 The perturbed case

In this part the sensitivity of the nominal transfer matrices to perturbations in the plant are discussed. The sensitivity relations of the transfer matrices in Table 1 are given in Table 2. The uncertainties are best described in Figure 9.

Table 2 : Sensitivities of the closed-loop transfer matrices

$\tilde{G}$	$\tilde{T}_{oi}$	oi
$G(I+\Delta)$	$(I+H_i\Delta)^{-1}T_{oi}$	un_i, un_o, uw, ur
$G(I+\Delta)^{-1}$	$T_{oi}(I+\Delta S_i)^{-1}$	an_i, yn_i
	$T_{oi} - T_{an_i}(I+\Delta S_i)^{-1}\Delta T_{ur}$	ar
$(I+\Delta)G$	$T_{oi}(I+\Delta H_o)^{-1}$	yw, un_o, uw, an_o, aw
$(I+\Delta)^{-1}G$	$(I+S_o\Delta)^{-1}T_{oi}$	yn_o, yn_i, yr
	$T_{oi} - T_{an_o}\Delta(I+S_o\Delta)^{-1}T_{yr}$	ar

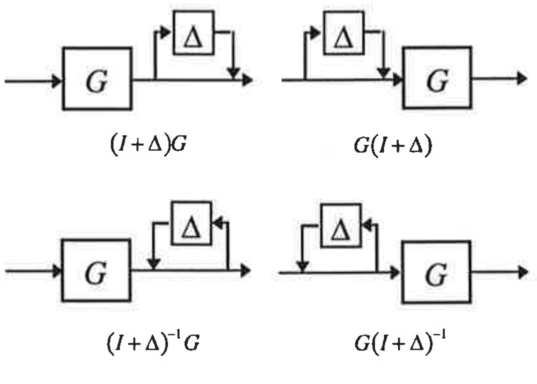


Figure 9: Different uncertainties

Excluding T_{ar} the sensitivities are wholly determined by the sensitivity and the complementary sensitivity transfer matrices. This means that the sensitivities of the control and diagnostic nominal performance to modeling errors are linked.
Some interesting points can be made about T_{ar}:

- Since $T_{ar} = C_{11}$ this nominal transfer matrix can, and often will, be designed to be 0. Therefore a relative sensitivity analysis is inappropriate for this transfer function. The expressions for $\tilde{T}_{ar}$ in Table 2 therefore reflect an absolute sensitivity analysis.
- The two expressions for $\tilde{T}_{ar}$ given in the table both involve a) one of the two sensitivity transfer matrices, determined by K_{22} or C_{22}, b) a "nominal control transfer matrix" determined by K_{22} or C_{22}, and c) a "nominal diagnostic transfer matrix" determined by K_{12} or C_{12}. This implies that nominal control performance, nominal diagnostic performance, and the sensitivity of the control/diagnostic interaction have to be traded off against one another.
- If we assume that $T_{ar} \approx 0$, $I + \Delta S_i \approx I$, and $I + S_o\Delta \approx I$, some insight can be gained. These assumptions do not impose any significant loss of generality. Under these assumptions the expressions for $\tilde{T}_{ar}$ become

$$\tilde{T}_{ar} \approx -T_{af_i}\Delta T_{ur}, \quad \tilde{T}_{ar} \approx -T_{af_o}\Delta T_{yr}$$

  and the tradeoffs between nominal control performance, nominal diagnostic performance and the sensitivity of the control/diagnostic interaction become a little clearer. (e.g. if the transfer matrix T_{af_i} is "large", which is exactly what is wanted if actuator faults are to be tracked, so will the sensitivity of the control/diagnosis interaction to uncertainties be).

3.4 The standard problem

To make a unified treatment it is now common practice to study the so called "standard problem". This setup was introduced around 1980 as a unified representation of a large collection of different problems in control and signal estimation. The setup is shown in Figure 10.

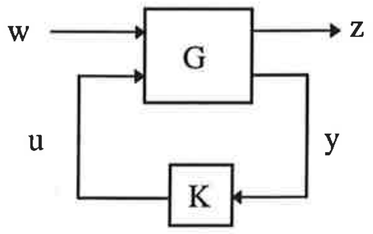


Figure 10: The setup in the standard problem

The signals defined in the figure are:

$$u = \textit{control inputs}$$
$$y = \textit{measured outputs}$$

$$w = \textit{exogenous inputs} = \begin{cases} \textit{fixed commands} \\ \textit{disturbances} \\ \textit{noise} \\ \vdots \end{cases}$$

$$z = \textit{regulated outputs} = \begin{cases} \textit{tracking errors} \\ \textit{control inputs} \\ \textit{measured outputs} \\ \textit{states} \\ \vdots \end{cases}$$

In H_∞-synthesis the regulated outputs z are to be minimized and contain in this thesis for example the difference between an alarm signal a and a fault that is to be detected.

The open loop system $G(s)$ is given by

$$z = G_{11}(s)w + G_{12}(s)u$$
$$y = G_{21}(s)w + G_{22}(s)u$$

or in state space form

$$\dot{x} = Ax + B_1w + B_2u$$
$$z = C_1x + D_{11}w + D_{12}u$$
$$y = C_2x + D_{21}w + D_{22}u$$

The closed loop system if $u = K(s)y$ is then given by

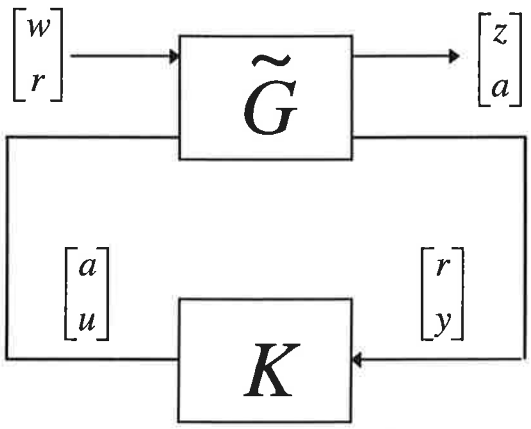
$$z = T_{zw}(s)w = \left(G_{11} + G_{12}K(I - G_{22}K)^{-1}G_{21}\right)w .$$

3.4.1 An equivalent system

A four degree of freedom problem can easily be translated to a standard problem. One way is to rewrite the plant as

$$\tilde{G} = \begin{bmatrix} G_{11} & 0 & 0 & G_{12} \\ 0 & 0 & I & 0 \\ 0 & I & 0 & 0 \\ G_{21} & 0 & 0 & G_{22} \end{bmatrix}$$

where I denotes an identity matrix of appropriate dimensions, and G_{ij} come from the original process. The controller stays exactly the same as before.



$\begin{bmatrix} w \\ r \end{bmatrix}$
$\tilde{G}$
$\begin{bmatrix} z \\ a \end{bmatrix}$
$\begin{bmatrix} a \\ u \end{bmatrix}$
$\begin{bmatrix} r \\ y \end{bmatrix}$
K

Figure 11: The four degree of freedom controller as a standard problem.

Now all inputs and outputs are to the system are inputs and outputs of G, and it is possible to use results from (for example) H_∞, H_2 and L_1 to design integrated control and diagnostic systems. In my master thesis I have used H_∞ – and μ-synthesis for the design.

4 H_∞ - Control

When a model based approach is used to design a controller there is a problem of a mismatch between the model and reality. This mismatch can, for example, be due to wear or tolerances. To improve the model can be difficult or costly. Sometimes it can be desirable to work with a smaller (less complex) model, accepting that there is a slight mismatch.
To address these uncertainties Zames and others formulated what is now called the H_∞ - problem. Here the plant uncertainty is directly incorporated in the synthesis, and the obtained controller is optimally robust in a rather restricted sense. During the last ten to fifteen years this has been a very popular research area, and there is a number of papers and books (see for example [3] or [5]) on the subject. Two MatLab toolboxes for robust control also exist - the Robust Control Toolbox and μ-Tools [1]. I have used μ-Tools in this thesis.
The following sections contain a short presentation of the general ideas of H_∞-control. Proofs are therefore omitted. The presentation follows [2] closely.

4.1 Basic definitions

The energy of a signal is defined as:

$$\text{Energy} = \int_{-\infty}^{\infty} |u(t)|^2 \, dt$$

and the L_2-norm as

$$\|u\|_2 = \left(\int_{-\infty}^{\infty} |u(t)|^2 \, dt \right)^{1/2} = \left(\frac{1}{2\pi} \int_{-\infty}^{\infty} |U(j\omega)|^2 \, d\omega \right)^{1/2}.$$

The last equality says that the energy can be evaluated both in the time domain and in the frequency domain. This is Parseval's theorem. The L_2-norm is (with some changes) also known as the RMS-value (Root-Mean-Square). The H_∞-norm of a stable system with transfer function $G(s)$ is now defined as the maximal RMS-gain:

$$\|G\|_\infty = \sup_{u \neq 0} \frac{\|Gu\|_2}{\|u\|_2} = \sup_{\|u\|_2 = 1} \|Gu\|_2$$

4.1.1 Single-input single-output systems

If $g(s)$ is a stable, linear, single-input single-output (SISO) system, then the H_∞-norm equals the largest value of the amplitude of the transfer function on the imaginary axis $(s = j\omega)$.

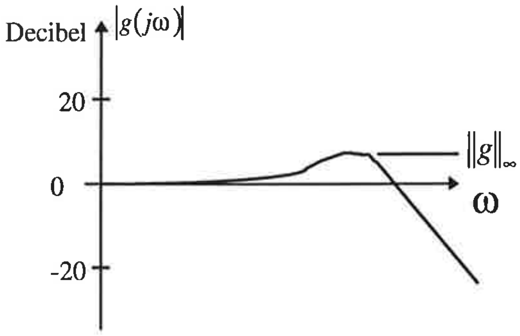


Figure 12: The norm is given by the maximum of the amplitude diagram

This can be written as:

$$\|g\|_\infty = \sup_\omega |g(j\omega)|$$

The H_∞-norm has some interesting characteristics:

$$\|f+g\|_\infty \le \|f\|_\infty + \|g\|_\infty$$
$$\|fg\|_\infty \le \|f\|_\infty \|g\|_\infty$$

The last expression is called the submultiplicability of the H_∞-norm. The submultiplicability separates the H_∞-norm from many other norms and makes it possible to use the H_∞-norm to guarantee robustness properties. This is described later in this section.

4.1.2 Multivariable systems

For multivariable systems there is an extra problem of interactions, which means that the different inputs influence not only one output, but two or more. The ideal would be to be able to treat a multivariable system as several unrelated SISO

systems and make the design for each separately. This is not always easy to do. A good analysis tool must therefore be able to handle interactions.
One way to do this is to use singular values. The singular values of a matrix A are defined by

$$\sigma_i(A) = \left(\lambda_i(A^*A)\right)^{1/2},$$

where λ_i denotes the ith largest eigenvalue and A^* is the complex conjugate transpose of A. The largest singular value σ_1 satisfies

$$\sigma_1(A) = \max \frac{\|Ax\|}{\|x\|}.$$

From this it can be shown that

$$\sigma_1(A+B) \leq \sigma_1(A) + \sigma_1(B)$$
$$\sigma_1(AB) \leq \sigma_1(A)\sigma_1(B).$$

If $G(s)$ is a stable linear multivariable system then

$$\|G\|_\infty = \sup_\omega \left(\sigma_1(G(j\omega))\right)$$

which means the maximum of the largest singular value.

4.2 H_∞ and robust control

There are many ways of modeling uncertain elements in a control loop. In many cases the perturbed plant can be represented as the nominal plant G_K with a feedback loop with Δ which contains all uncertain elements. This situation is shown in Figure 3.

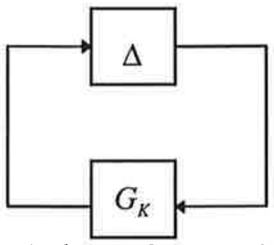


Figure 13: Nominal control system with uncertainty. All uncertain blocks are included in Δ.

Many different assumptions can be made on the matrix Δ. One common setup is a block diagonal Δ:

$$\Delta = \operatorname{diag}\left(\Delta_R,\ \Delta_C,\ \Delta_{full}\right)$$

Here Δ_R is a diagonal matrix containing real scalar elements, Δ_C is a diagonal matrix containing complex scalar elements, and Δ_{full} is a full complex matrix containing dynamic elements.
The optimal robustness problem is now to choose $K(s)$ such that the loop around G_K is stable against as large Δ as possible. In H_∞-control theory only one block, Δ_{full}, is present.

Theorem 1: Small Gain Theorem

Let G_K be a stable system. Then the loop in Figure 13 is stable for all stable $\Delta(s)$ with $\|G_K\Delta\| < 1$. Because of the submultiplicability of the H_∞-norm the condition is automatically satisfied if $\|G_K\|_\infty \|\Delta\|_\infty < 1$.

The theorem is not conservative, meaning that there exists a complex matrix Δ with $\|\Delta\|_\infty = \|G_K\|_\infty^{-1}$ that destabilizes the loop. Therefore, to maximize robustness against unstructured perturbations means to:

Find a controller K such that $\min\limits_K \|G_K\|_\infty$ is achieved.

4.2.1 Examples of uncertainties

Theorem 1 can be used in a number of ways. G_K often has the interpretation of some function of the nominal closed loop system and Δ can be either additive, multiplicative, uncertainty in the feedback etc. If the theorem is used on the system in Figure 13, and the nominal plant is disturbed by an additive model uncertainty, so that the actual system is $G+\Delta$ instead of G, then the system will be robust against as large Δ as possible if the H_∞-norm of $K(I-GK)^{-1}$ is minimized. This is shown in Figure 14.

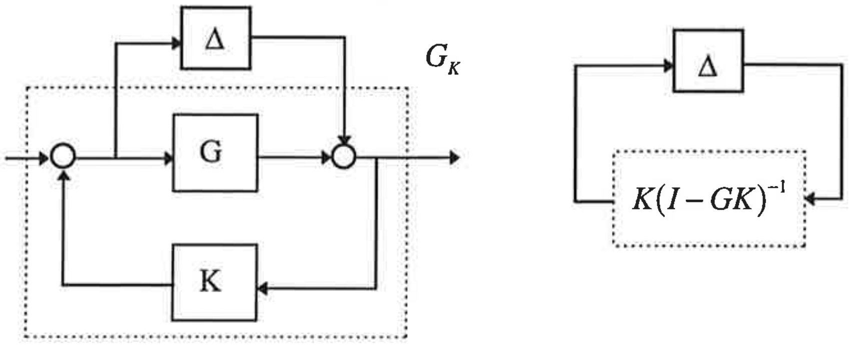


Figure 14: Additive uncertainty and the transfer function to be minimized.

An alternative to additive uncertainty is multiplicative uncertainty, where G is changed to $(I+\Delta)G$ if the uncertainty is on the output and to $G(I+\Delta)$ if the uncertainty is on the input. Figure 15 shows the situation with a multiplicative uncertainty on the input. Depending on how the uncertain elements occur different transfer functions are to be minimized. In the case of multiplicative input uncertainty it is $(I-KG)^{-1}KG$. Of course there is a lot of different ways to model the uncertainty.

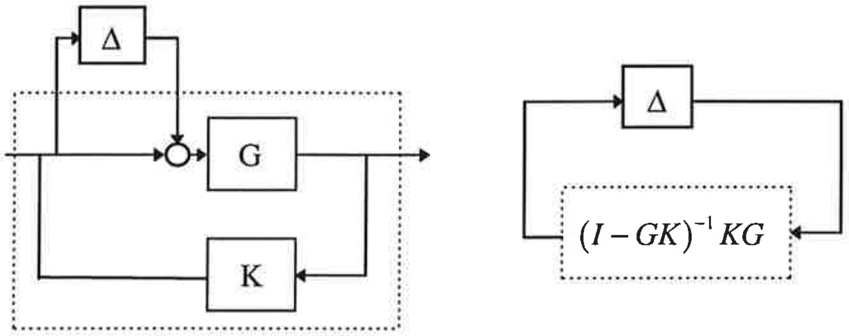


Figure 15: Multiplicative input uncertainty and the transfer function to be minimized.

4.2.2 Robust performance or robust stability?

The H_∞-theory solves a robust stability problem. This means that the system is guaranteed to maintain stability for a certain set of model perturbations. Most control systems can however become useless when much smaller system variations occur without causing instability.
The goal of a robust controller design is to achieve good performance in face of plant uncertainty. This is called robust performance. To achieve robust performance is much harder than to achieve robust stability. One reason for this is

that there is no explicit formula for good performance - it takes engineering skills to specify what is "good". The robust stability problem is closer to pure mathematics.
The standard H_∞-problem gives equal weight to all frequencies. In a real design it is often desirable to emphasize different aspects at different frequencies. This is done by adding weights or filters when formulating the problem. These filters must be stable, proper transfer functions (not $W = 1/s$ or $W = s$). This does not impose any larger limitations, since it is easy to move the poles or zeros a little bit into the left half plane.

4.3 How to compute the norm and the controller

So far only the definitions and interpretations of the H_∞-norm have been discussed. The following two sections give some information about how to compute the norm and the controller.

4.3.1 The norm

The first way to compute the H_∞-norm that comes to mind is to plot $\sigma_1(G(j\omega_k))$ for a sufficiently large number of ω_k, and find the peak value. This method of course has some drawbacks - it can be quite slow. A faster way is as follows:
Given $G(s) = C(sI - A)^{-1}B$, where A is stable, and $\gamma > 0$ the matrix G_γ is formed as:

$$G_\gamma = \begin{bmatrix} A & \gamma^{-1}BB^T \\ -\gamma^{-1}C^TC & -A^T \end{bmatrix}$$

It is then possible to show that

$$\|G\|_\infty < \gamma \quad \Leftrightarrow \quad G_\gamma \text{ has no imaginary eigenvalues}$$

Thus we can check if $\|G\|_\infty < \gamma$ by forming G_γ and computing its eigenvalues. The condition can also be expressed in terms of the existence of a real, positive definite solution to an algebraic Riccati equation, or in terms of a linear matrix inequality.

4.3.2 The controller

Following the setup in the standard problem presented above we have

$$\begin{aligned}\dot{x} &= Ax + B_1 w + B_2 u\\ z &= C_1 x + D_{11} w + D_{12} u\\ y &= C_2 x + D_{21} w + D_{22} u\end{aligned}$$

For simplicity $D_{11} = 0$ and $D_{22} = 0$ is assumed. It is very difficult to find the controller that minimizes the H_∞-norm directly. The easiest way is to find a controller such that $\|T_{zw}\|_\infty < \gamma$ (the sub-optimal problem) for a given γ. The value of γ is then decreased until no such controller exists. This is called γ-iteration.

If the system is given as above and the conditions

(A1) $(A,\ B_2)$ is stabilizable and $(C_2,\ A)$ is detectable
(A2) D_{12} and D_{21} have full rank
(A3) $\begin{bmatrix} A - j\omega I & B_2 \\ C_1 & D_{12} \end{bmatrix}$ has full column rank for all ω
(A4) $\begin{bmatrix} A - j\omega I & B_1 \\ C_2 & D_{21} \end{bmatrix}$ has full row rank for all ω

are satisfied, then there is a controller $K(s)$ such that $\|T_{zw}\|_\infty < \gamma$ if and only if the Riccati equations

$$\begin{aligned}0 &= XA + A^T X + C_1^T C_1 - X\left(B_2 B_2^T - \gamma^{-2} B_1 B_1^T\right)X\\ 0 &= AY + YA^T + B_1 B_1^T - Y\left(C_2^T C_2 - \gamma^{-2} C_1^T C_1\right)Y\end{aligned}$$

have positive definite solutions X and Y such that $\gamma^2 Y^{-1} - X > 0$ and such that $A - \left(B_2 B_2^T - \gamma^{-2} B_1 B_1^T\right)X$ and $A^T - \left(C_2^T C_2 - \gamma^{-2} C_1^T C_1\right)Y$ are stable. One such controller ("the central") is given by

$$\begin{aligned}\dot{\hat{x}} &= A\hat{x} + B_2 u + \gamma^{-2} Y C_1^T C_1 \hat{x} + Y C_2^T (y - C_2 \hat{x})\\ u &= -B_2^T X\left(I - \gamma^{-2} Y X\right)^{-1}.\end{aligned}$$

The conditions in (A1) are weaker forms of controllability and observability conditions, respectively. That D_{12} has to have full rank and (A3) means that all control signals have to be punished at all frequencies (no zero on the imaginary axis), and that no roll off is allowed at high frequencies. Correspondingly, the condition on D_{21} and (A4) means that all measurements used in the control have to be noisy at all frequencies. Not satisfying one of the last two conditions would mean that the control signal could be unlimited at some frequency, or that a disturbance of a certain frequency would pass undetected.

4.4 Structured singular values and DK-iteration

In H_∞-control only the full block uncertainty Δ_{full} is present. This often causes the controller to be very conservative, meaning that it is designed to be robust against uncertainties that are not going to appear. Figure 16 shows a plant with input uncertainty. If this uncertainty is in the actuators (for example uncertain actuator gain), it is reasonable to assume that there are no cross terms in Δ. This can be expressed as:

$$\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} \Delta_{11} & 0 \\ 0 & \Delta_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

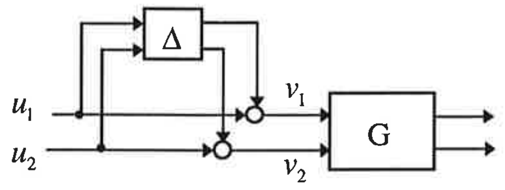


Figure 16: Plant with input uncertainty

In pure H_∞-control the controller would allow for cross terms to be present. This means that the design algorithm assumes that the actuators influence each other directly. To be able to handle uncertainties which are not full blocks one can use *structured singular values* and *DK-iteration* [1].

4.4.1 Structured singular values

The structured singular value is a matrix function denoted $\mu(\cdot)$. In the definition of $\mu(M)$, where $M \in C^{n \times n}$, there is an underlying structure Δ, which is a prescribed set of block diagonal matrices as mentioned in section 4.2. In this case the two blocks Δ_C and Δ_{full} are present. With these blocks we define $\Delta' \subset C^{n \times n}$ as

$$\Delta' = \{ \operatorname{diag}[\Delta_C, \Delta_{full}] \}.$$

Here the scalar block comes first for notational reasons. The blocks can come in any order - even mixed. For $M \in C^{n\times n}$, $\mu_\Delta(M)$ is defined

$$\mu_\Delta(M) = \frac{1}{\min\{\sigma_1(\Delta) : \Delta \in \Delta', \det(I - M\Delta) = 0\}}$$

unless no $\Delta \in \Delta'$ makes $I - M\Delta$ singular, in which case $\mu_\Delta(M) = 0$.
To interpret the definition of $\mu_\Delta(M)$ the feedback loop shown in Figure 17 can be useful. Here $M \in C^{n\times n}$ is given.

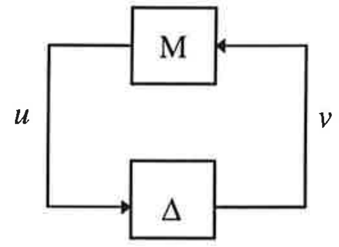


Figure 17: Feedback loop used to interpret the definition of μ(M).

This figure is meant to represent the loop equations

$$u = Mv$$
$$v = \Delta u.$$

As long as $I - M\Delta$ is non-singular the only solution u,v to the loop equations is $u = v = 0$. If $I - M\Delta$ is singular there are infinitely many solutions, and the norms $\|u\|$ and $\|v\|$ can be arbitrarily large. With an abuse of convention the system is then called "unstable". Similarly the system is called "stable" when the only solution is identically zero. This means that $\mu_\Delta(M)$ is a measure of the smallest structured Δ that causes "instability" of the feedback loop in Figure 17. Note that μ is not a norm, since it does not satisfy the triangle inequality. However, for any $\alpha \in C$, $\mu(\alpha M) = |\alpha|\mu(M)$, so in some sense it is related to how "big" the matrix is in a norm sense.

$\mu_\Delta(M)$ is very hard to compute exactly. Instead an upper and a lower bound are used. These bounds are

$$\max_{U} \rho(QM) \leq \mu_\Delta(M) \leq \inf_{D} \sigma_1(DMD^{-1}),$$

where $\rho(QM)$ is the spectral radius of QM, D commutes with Δ $(D\Delta = \Delta D)$ and $Q \in \mathcal{Q}$, $\mathcal{Q} = \{Q \in \Delta\text{: } Q^*Q = I\}$. For a more thorough definition see [1].

4.4.2 DK-iteration

The idea behind DK-iteration is to minimize the upper bound of μ, that is to find

$$\min_{D,K} \sup_{\omega} \sigma_1(DMD^{-1}).$$

This is done as follows:
1. Fix D and find K with H_∞-theory
2. Fix K and optimize $D(j\omega)$ (the D-scales) for every ω
3. Approximate these $D(j\omega)$ with a dynamical system
4. Include $D(s)$ and $D^{-1}(s)$ into G

Iterate these steps until μ converges or $\mu < 1$. If the resulting $\mu > 1$ robust control has not been achieved.

4.4.3 Interpretation

The DK-iteration scheme can be interpreted with the help of Figure 18.

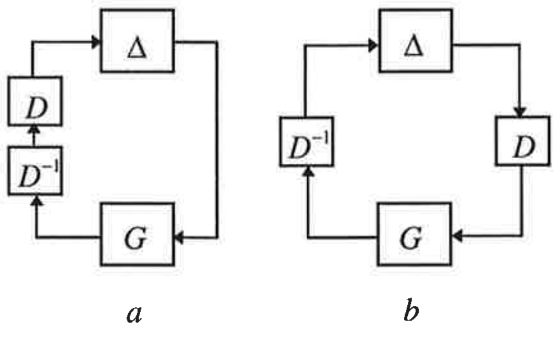


Figure 18: Two identical loops showing the idea behind the D-scales.

In Figure 18a $D^{-1}D = I$ has been inserted in the loop. In Figure 18b D has been moved to the other side of Δ. This is allowed, since D and Δ commute. D is then a scaling that drives the unstructured block uncertainty underlying the H_∞-design towards a Δ with the wanted structure. This can not be done in one step - hence the iteration.

4.5 Limitations

The design methods discussed above have some drawbacks. One is that the resulting controllers often are of very high order. Therefore some model reduction techniques are required. Also the speed of the poles and zeros of the controllers vary a lot - differences of an order of 10^{10} in magnitude are not unusual. This makes the controllers hard to simulate and implement. As mentioned above the H_∞-controllers alone can be very conservative, which led to the DK-iteration. The controllers which are the result of a DK-iteration can also be somewhat conservative. This since all elements of the uncertainty are allowed to be complex. Sometimes it is reasonable to assume that the uncertainty does not have any imaginary part, for example when the gain in an actuator is uncertain. There exists theory for real parameters, but it was not supported in the software used in this thesis.

5 An example

Most papers on H_∞ concentrate on proving and elucidating certain properties of the norm and the resulting controllers. Only few provide descriptions of examples or complete designs. One exception is [8], which describes the design of an integrated control and diagnostic module using the four degree of freedom controller. The plant in the paper is a two input two output plant, with a 10% uncertainty in the gain of each input channel. The diagnostic objective is to detect faults in the actuators.
The first part of this thesis was a thorough investigation of this example with the aim to reproduce the results and to understand the tradeoffs made.

5.1 The setup in the article

The process presented in the article is a first order two input two output process. The state space description is as follows:

$$\dot{x} = \begin{bmatrix} -0.1536 & 0.1914 \\ -0.1583 & -0.5311 \end{bmatrix} x + \begin{bmatrix} -0.2290 & 0.1701 \\ -0.1719 & -0.0166 \end{bmatrix} u$$
$$y = \begin{bmatrix} 0.1301 & -0.0210 \\ 0.2538 & -0.1714 \end{bmatrix} x$$

The transfer functions of the process are shown in Figure 19.

The first step was to recreate the filter (weight) and process configuration. Figure 20 shows the control and diagnostic configuration as presented in the paper.
To be able to use H_∞ – and μ - synthesis the design problem must be formulated as a standard problem. This means that all external inputs and outputs should be inputs and outputs of the process, not of the controller. Therefore the reference command r is fed through the interconnection and is viewed as a measurement for the controller.
For the same reason the alarm signal a must be fed through the interconnection and is viewed as a control signal. Since all control signals must be punished (see chapter 4.4.2, conditions A2 and A4) also a must be punished, which is why the weight W_a is included in the interconnection shown in Figure 21.
W_c is there to emphasize the reference tracking, and W_d to make sure that the alarm signals track the faults at the right frequencies. An uncertainty of 10% in the gain of each input channel is assumed. This assumption is modeled as $W_u = 0.1I, \Delta_u = \text{diag}(\Delta_1, \Delta_2)$, where Δ_1 and Δ_2 are scalar parameters and

$\|\Delta_u\|_\infty \le 1$. Any other weights are not mentioned in the paper. The resulting interconnection is shown in Figure 21.

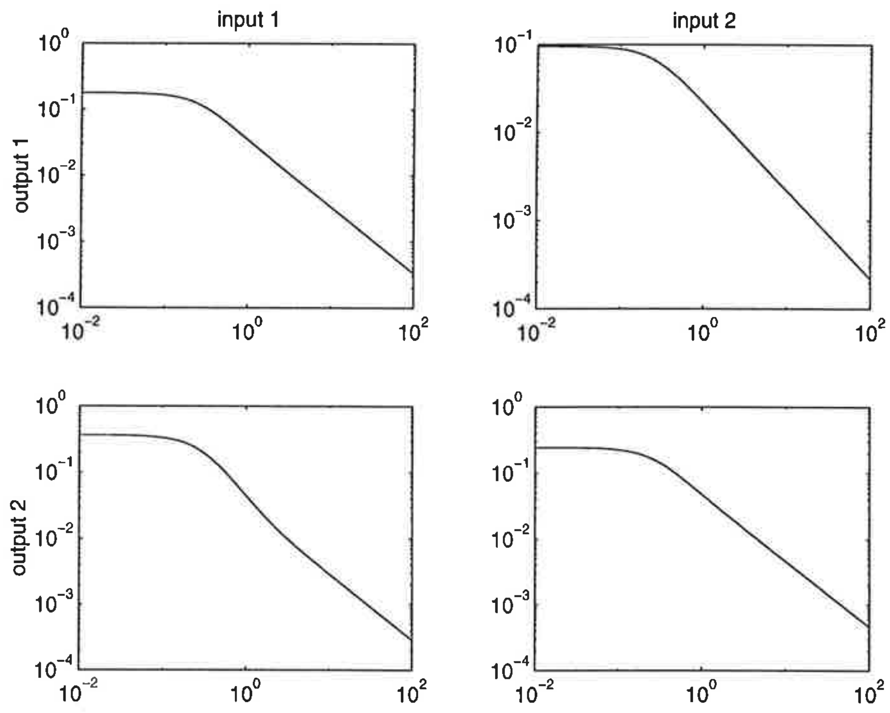


Figure 19: The transfer functions of the process in the example

If T_c is the transfer function from $[f^T, n^T, r^T]^T$ to *y-r*, T_d the transfer function from the same inputs to *a-f* and T_a the transfer function from the same inputs to *a*, then the design objective can be expressed as follows: Find a stabilizing *K* such that

$$\left\| \begin{matrix} W_c T_c \\ W_d T_d \\ W_a T_a \end{matrix} \right\|_\infty \le 1 \text{ for all } \Delta_u \text{ such that } \|\Delta_u\|_\infty \le 1.$$

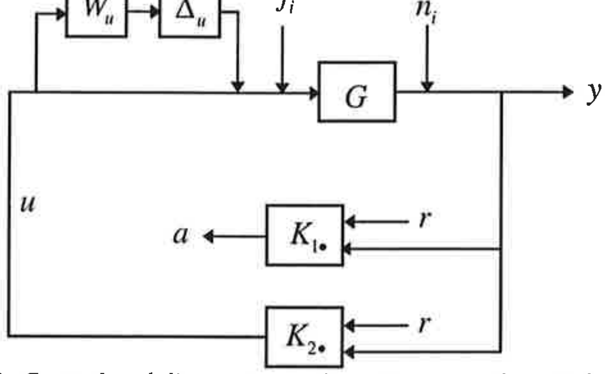


Figure 20: Control and diagnostic configuration as in the article. Note how n comes in.

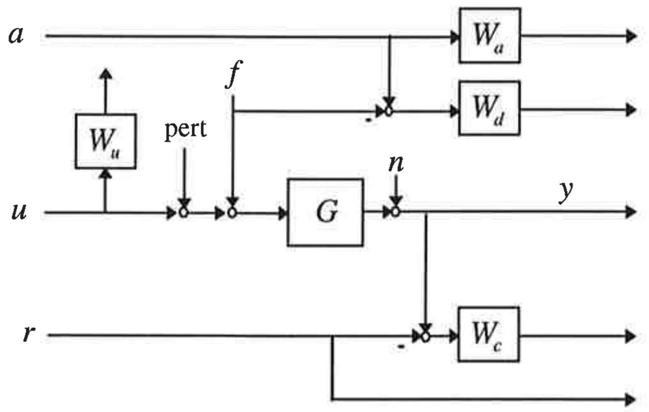


Figure 21: Interconnection of weights and process. y and r are seen as measurements. The other outputs are to be minimized.

The next step was to approximate the weights which were given in a figure in the paper. All signals shown in Figure 21 are vectors of length 2. Therefore all weights are diagonal two input two output filters and have - in this example - the same weight on both components.
W_c is a first order low pass filter, with a low frequency gain of 10 and a bandwidth of 10^{-4} radians per second. W_a is a first order high pass filter, with a high frequency gain of 1 and low frequency gain of approximately $5*10^{-7}$. W_d is a band pass filter with peak value at frequencies just above the process bandwidth. The approximated filters are shown in Figure 22. In the H_∞-synthesis, the outputs from W_a, W_c and W_d are to be minimized. The outputs y and r (r is fed straight through from the input to the output) are measurements for the controller.

Minimizing the output from W_a (with W_a chosen as in Figure 22) means that higher frequencies of the alarm signal a are punished. Since a is a control signal, it has to be punished at all frequencies.

W_c says that the control error y-r is to be minimized. As mentioned in chapter 4, the H_∞-norm of a system can be interpreted as the maximal RMS-gain of the system. If the L_2-norm of the inputs and the achieved H_∞-norm of the system are both less than 1, the norm of the output from W_c will also be less than 1. Since the output from W_c is the control error multiplied by 10 at low frequencies, a steady state error of less than 10% is achieved. If this steady state gain of W_c is raised, the steady state control error will get smaller, but it will be harder to achieve a H_∞-norm of less than 1 for the system. If the breakpoint of W_c is moved to a lower frequency, the control will be slower. Thus W_c strongly influences the control performance.

The alarm signal a is supposed to track the fault f at the frequencies where W_d is large. What a is supposed to be at other frequencies is not defined. Therefore a will contain "undefined" information, which has to be removed in order to get a clean alarm signal. To do this, a is filtered with W_d.

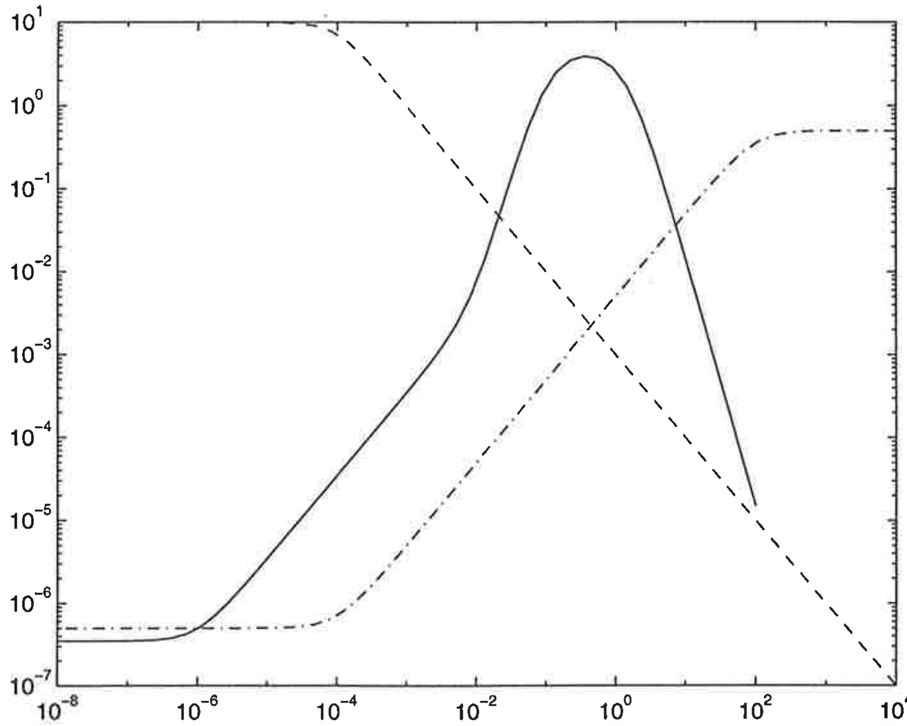


Figure 22: The approximated performance weights. Wa is dash-dotted, Wc dashed and Wd solid.

After some iterations on the DK-scheme a μ-value of 3.92 is achieved for this interconnection. If a fourth order fit is used for approximating the D-scales the controller will be of order 30. Simulation results of this design are shown in Figure 23 and Figure 24.

The reference is a step change from 0 to -1 at t=10 in the reference for output two. The fault is added to the control signal of input 1. It is a ramp from 0 to 1, starting at t=100 and stopping at t=200.
The outputs are shown in the top half of Figure 24 (output 1 is shown solid, and output 2 dash-dotted). The control is slow, and sensitive to actuator faults. The bottom half of the same figure shows the alarm signals. Alarm 1 (solid) is supposed to track actuator one faults, and alarm 2 (dash-dotted) actuator two faults. It is clearly seen that a false alarm is generated on alarm 2 when the fault on actuator one occurs.

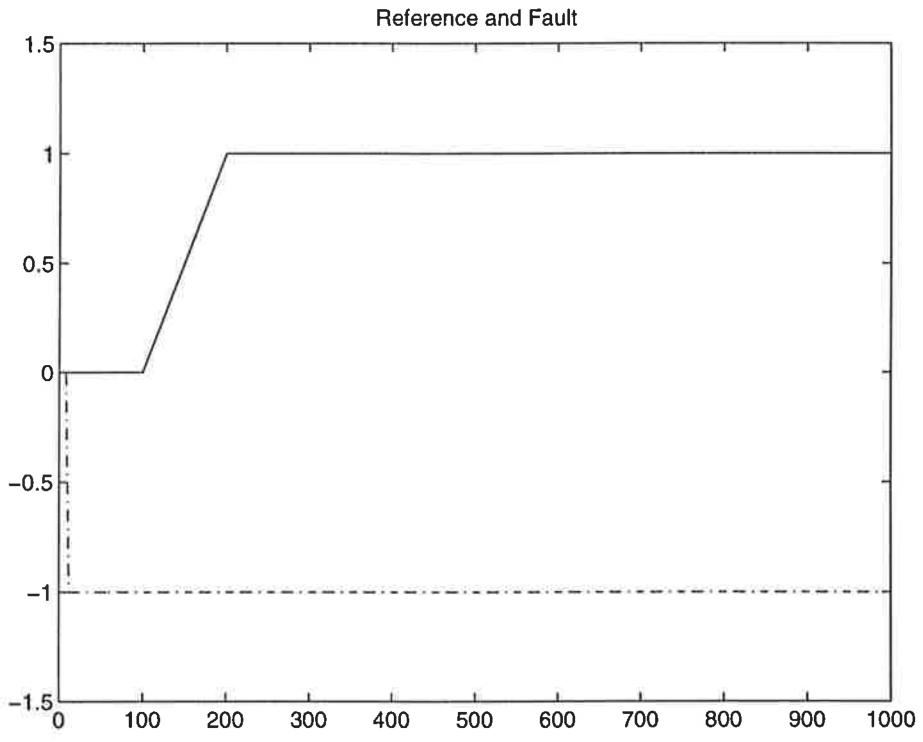


Figure 23: Reference signal (dash-dotted) and fault (solid) used in the simulations

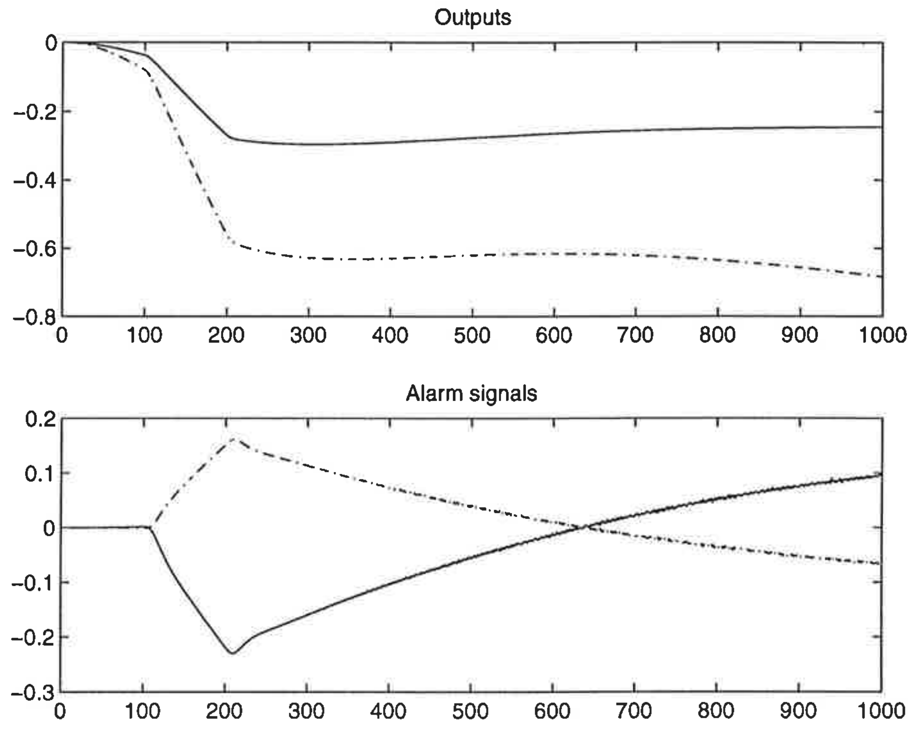


Figure 24: Outputs and alarm signals achieved with the setup from the article. Top: Output 1 (solid) and output 2 (dash-dotted). Bottom: The alarm signal supposed to track the fault on actuator 1 (solid) and the alarm signal for actuator 2 (dash-dotted).

5.2 The correct setup

The simulations presented in the article show a much better behaviour on both control and diagnostics than I could achieve using the given information. This indicated that I had misinterpreted the setup in the article or that it was not complete. After some communication with Matthew Tyler, he sent a copy of the MatLab code he had used. This code showed some parts that, for some reason, are not mentioned in the article:

- A weight on the fault, W_f.
- A weight on the noise, W_n. It is also a pure measurement noise, and does not enter W_c.
- A different state space description of the process.

The interconnection really used in the article is shown in Figure 25 and the additional weights in Figure 26.

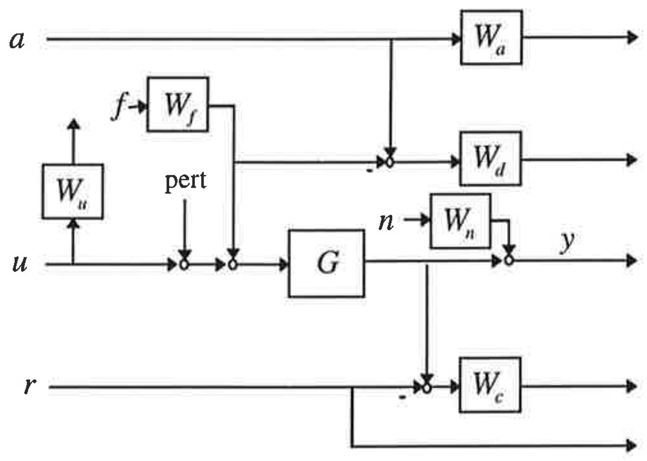


Figure 25: The complete interconnection. y and r are seen as measurements. The other outputs are to be minimized.

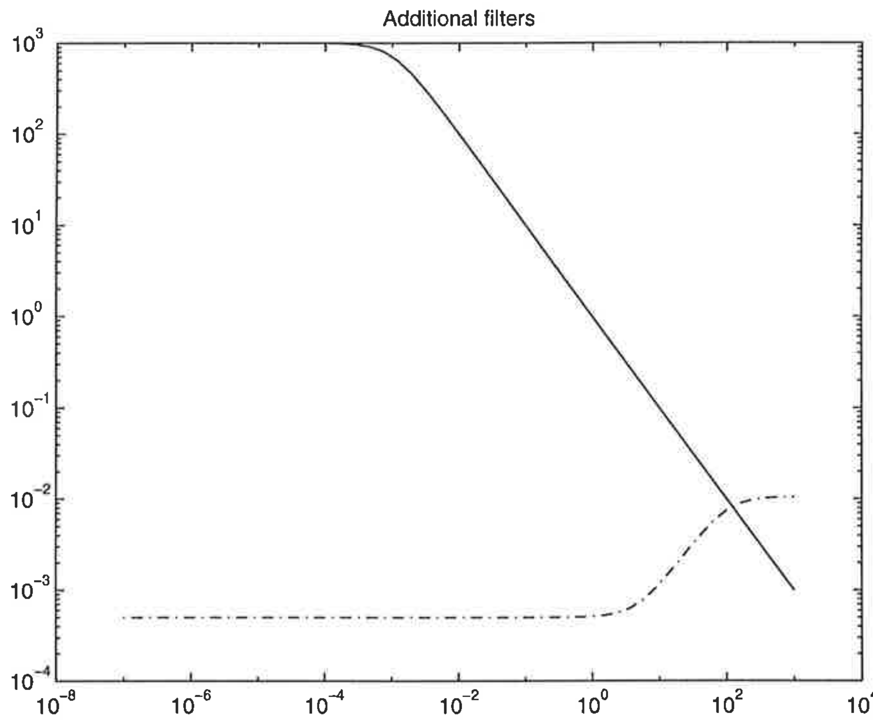


Figure 26: The additional weights. Wf is solid and Wn dash-dotted.

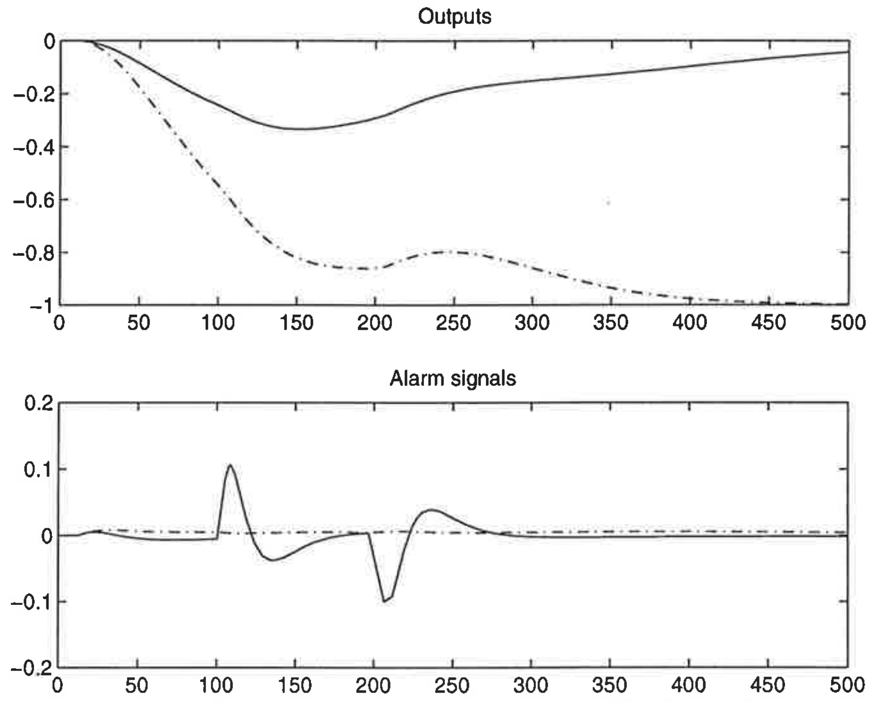


Figure 27: The simulation results as presented in the article. The faults and reference signals are the same as before.

The weight W_n is there to describe the frequency content and amount of measurement noise. The decrease of the noise level at low frequencies as shown in Figure 26 is not necessary to achieve a H_∞-norm of less than 1 in this example.
W_f "exaggerates" the influence of the fault on the process. This makes the controller concentrate more on the faults than it would do otherwise, and the resulting system should turn out to be less sensitive to faults.

The simulation result for the complete setup is shown in Figure 27. Here both the control and the diagnostics are better. There are no false alarms on the reference change and the fault on actuator 1 only causes alarm signal 1 to get large. This means that the control and diagnostic objectives are met.

6 The inverted pendulum

This chapter contains some examples of integrated control and diagnosis using the four degree of freedom controller. The process for which the designs are made is the inverted pendulum. The inverted pendulum was found to be very suitable to do the designs on since it is a multivariable process, where the two outputs influence each other strongly and have very different characteristics.

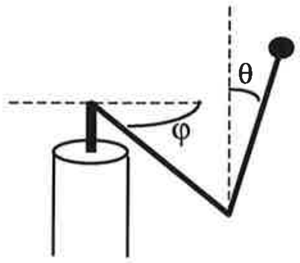


Figure 28: The inverted pendulum

Figure 28 shows a simple sketch of an inverted pendulum. A DC-motor is mounted in a shaft. A horizontal arm is attached to the motor spindle. At the end of this arm the actual pendulum is attached. At the end of the pendulum there is a small weight. The point where the arm and the pendulum are connected is called the pivot point. When a voltage is fed to the motor, the pivot point will accelerate with an acceleration proportional to the voltage. The angle θ between the pendulum and the vertical line and the angle φ between the arm and a fixed horizontal line are measured. The control task is to balance the pendulum in the upright position and to keep the arm at a certain angle to the fixed horizontal line.

6.1 The mathematical model

With the state variables

$$\begin{cases} x_1 = \theta \\ x_2 = \dot{x}_1 = \dot{\theta} \\ x_3 = \varphi \\ x_4 = \dot{\varphi} \end{cases}$$

the process model becomes (see for example [9]):

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = \dfrac{mgl_p}{J_p}\sin x_1 + \dfrac{ml_p}{J_p}u\cos x_1 \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = \dfrac{1}{l_a}u \end{cases}$$

where m is the mass of the weight at the end of the pendulum, J_p the moment of inertia, g the gravity, l_p the length of the pendulum and l_a the length of the arm. The control signal u is in m/s^2. The first two states describe the movement of the pendulum and the two last the position of the arm.
Linearizing the model given above around $[0\ 0\ 0\ 0]^T$, which is the upright position with zero speed, gives the following state space model:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = \dfrac{mgl_p}{J_p}x_1 + \dfrac{ml_p}{J_p}u \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = \dfrac{1}{l_a}u \end{cases}$$

The linear model is needed for the synthesis. All simulations have been made with the non-linear model.

6.2 Different designs

A number of designs were made for different setups of uncertainties and faults on the inverted pendulum. Here only the fault and uncertainty setup and the simulation results are presented. For a more exact description of the chosen weights see the MatLab files.

6.2.1 Fault and uncertainty on the input

The first design is for an uncertainty of 10% on the input gain and an actuator fault. In comparison with the example of Chapter 5, all weights first were moved up in frequency, so that the relation between process bandwidth and the weights was roughly kept. The control errors to be minimized are the pendulum angle and the difference between the arm angle (arm position) and its reference value. The weights for these two control errors are not the same, which reflects the fact that the most important task is to balance the pendulum - if it falls it is not interesting to keep the arm at a certain position. Therefore the weight on the pendulum angle is larger. Since the only input to the pendulum is the pivot point acceleration (or motor voltage), the uncertainty and the fault occur on only one signal.

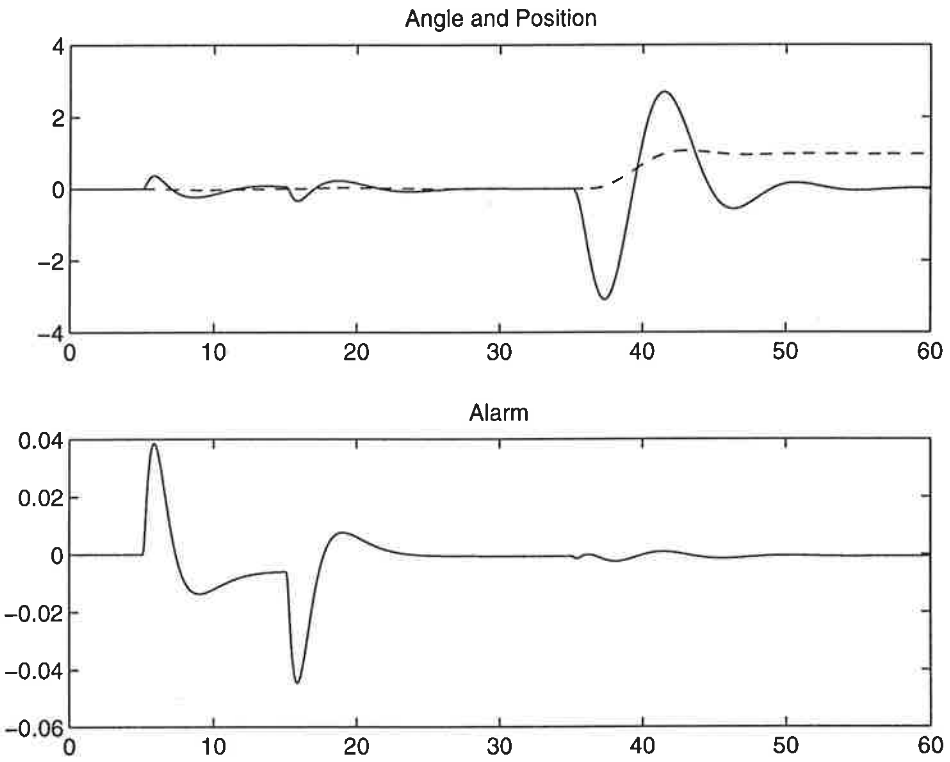


Figure 29: Simulation results of the pendulum with an uncertainty of 10% on the input and an actuator fault. Top: pendulum angle (solid, multiplied by 1000) and arm position (dashed). Bottom: alarm signal.

In Figure 29 the simulation results are shown. The simulation is made with an extra gain of 1.1 on the input. The fault is a ramp from 0 to 1 starting at t=5 and stopping at t=15. The reference for the arm position is a step from 0 to 1 at t=35. Note that the pendulum angle has been multiplied by 1000 in the figure. Otherwise the angle variations would not be visible.
The control is all right - the pendulum does not sway too much, and the arm follows the reference nicely. The alarm signal is only large when there is a change in the fault and not when there is a reference change. If the control is weighted more false alarms on the reference change will occur. This is shown in Chapter 6.4.

6.2.2 Fault and uncertainty on the input II

In comparison with the previous design, the only difference made here is that the diagnostic weight W_d is moved down in frequency and made a bit smaller, which causes the alarm signal to look a bit different than before.

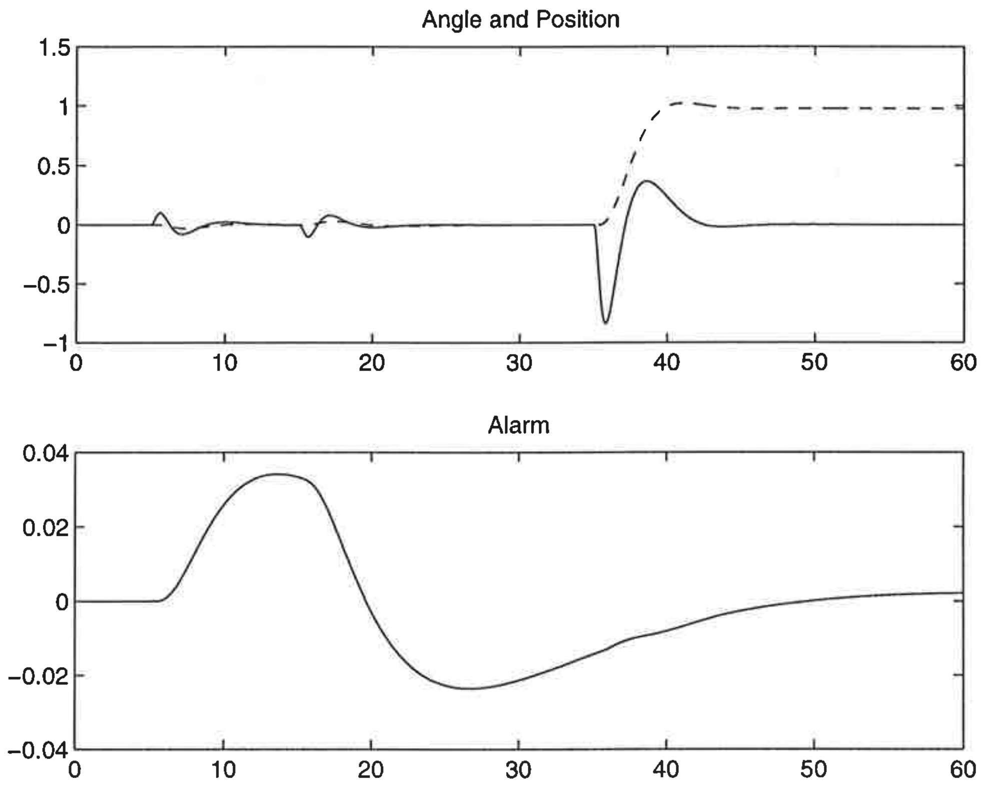


Figure 30: The pendulum with a 10% input uncertainty and actuator fault. The diagnosis is made at lower frequencies than before. Top: Pendulum angle (solid, multiplied by 100) and arm position (dashed). Bottom: Alarm signal.

If the weights on the control in this design are made larger, there will not - unlike in the previous design - occur any false alarms on reference changes, which implies that the trade-offs between control and diagnostic performance must not be too large if the diagnostics are made at the right frequencies.

6.2.3 Fault on the input and uncertainty on the output

This design is for an uncertainty of 10% in the sensor gain and an actuator fault. The main difference to the design in Chapter 6.2.1 is that an extra weight, W_y, is introduced to model the uncertainty. Since all control signals have to be punished, the weight W_u is still there, but is made smaller. The weight on the pendulum angle is made smaller and the weight on the arm position larger. W_d is almost the same as in the first design.

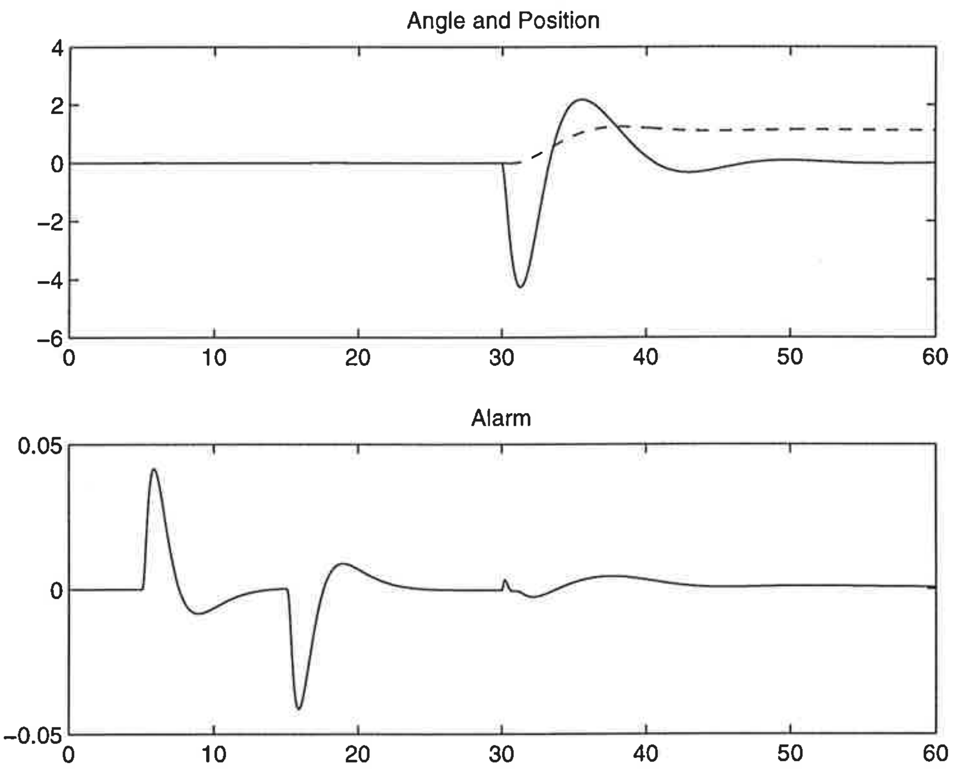


Figure 31: Simulation of pendulum with sensor uncertainty (10%) and actuator fault. Top: pendulum angle (solid, multiplied by 100) and arm position (dashed). Bottom: alarm signal.

Figure 31 shows simulation results of this setup. The used fault is the same as in the previous design. The reference change is at t=30. Also here the control performance and alarm signal are quite good.

6.2.4 Uncertainty on the input and sensor faults

In the two previous designs actuator faults were considered. In both designs the weighting on the fault, W_f, was quite large in order to make the system robust against as large faults as possible. It turned out to be much harder to make the system robust against sensor faults, why W_f is made much smaller in this design. Since the fault is smaller the diagnostic weight W_d is made larger. Another change is that a no longer tracks the faults f_o. Instead it tracks the "amplified" fault $80 * f_o$. The control also had to be weighted less and the uncertainty is only 5%.

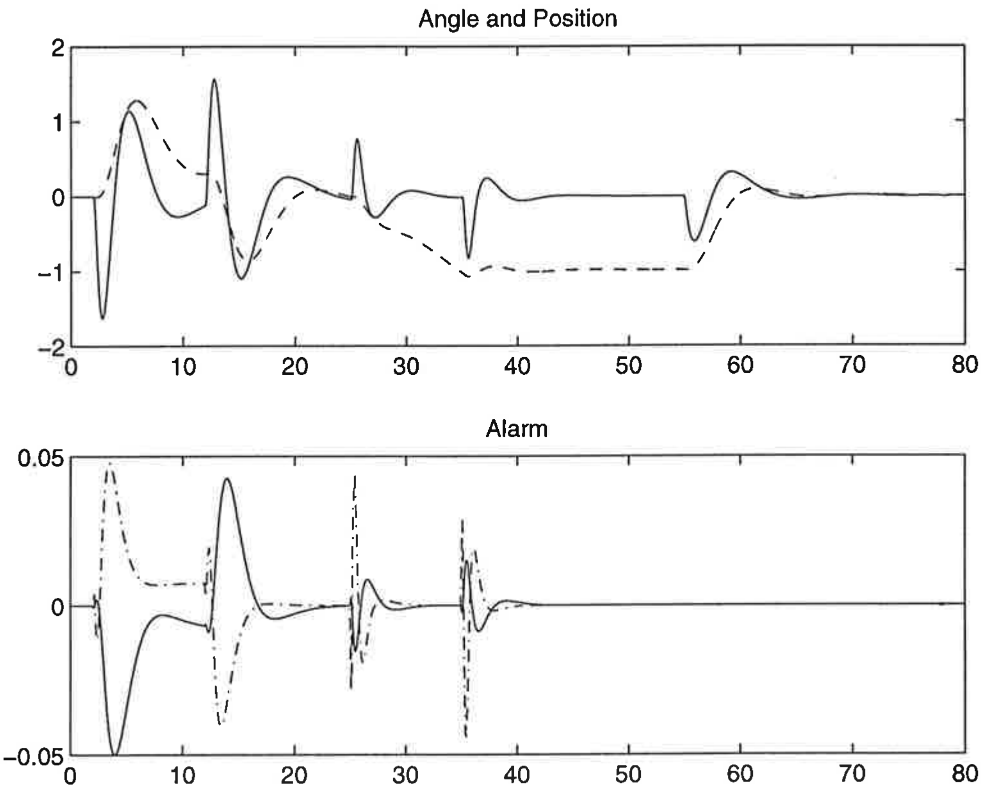


Figure 32: Simulation with 5% uncertainty on the input and sensor faults. Top: pendulum angle (solid, multiplied by 100) and arm position (dashed). Bottom: alarm for sensor 1 (solid) and for sensor 2 (dashed).

In Figure 32 a simulation of this setup is shown. Two faults have been used. The first is a ramp from 0 to 1 at t=2 to t=12 in sensor 1. The other is a ramp from 0 to 1 at t=25 to t=35 in sensor 2. A step reference change from 0 to 1 is made at t=55. The simulation shows the actual pendulum angle and arm position - not the measured ones. The controller manages to balance the pendulum despite of the fault in sensor 1. However, it does not manage to compensate for the fault in sensor 2. This is not surprising - the last two states in the state space description of the pendulum build a double integrator, and a bias (which is exactly what the fault is) on a double integrator is not observable.

The control is as expected slower than in the previous designs. Alarm signal 1 (for sensor 1 faults) shows a nice behaviour, but alarm signal 2 shows a large false alarm for faults in sensor 1. This could however be taken care of by some special decision logic.

In the first setup, where a was filtered by W_d a bias was added to both alarm signals when a fault on sensor 1 occurred. On alarm signal 1 this is good, but for

alarm signal 2 it is not acceptable. Therefore a is filtered by a special filter which behaves like W_d at the interesting frequencies and suppresses low frequencies more.
Moving W_d to lower frequencies as in the design in Chapter 6.2.2 renders the alarm signals useless.

Some attempts were made with both uncertainty and faults on the output. This turned out to be very hard. The pendulum could be kept upright, but good position control seemed impossible. The alarm signals also showed a strange behaviour. This indicates that an unstable process with uncertain, faulty measurements is hard to control.

6.3 Numerator-Denominator uncertainties.

The uncertainties underlying the designs presented so far have all been in the sensor or actuator gain. An alternative way of modeling uncertainties - the numerator-denominator perturbation model - is presented in [10]. It represents perturbations in the form

$$P_0 = \frac{N_0}{D_0} \rightarrow P = \frac{N_0 + \Delta_N}{D_0 + \Delta_D}$$

where N_0 and D_0 are the nominal numerator and denominator of the process transfer function, respectively.
The transfer function of the linearized pendulum (omitting the arm position) is given by

$$\theta = \frac{\frac{ml_p}{J_p}}{s^2 - \frac{mgl_p}{J_p}} u .$$

Letting the parameters m, l_p and J_p be perturbed to

$$\frac{ml_p}{J_p} + \Delta$$

can then be modeled as in Figure 33. The block g is there in order to get the same perturbation in both numerator and denominator.

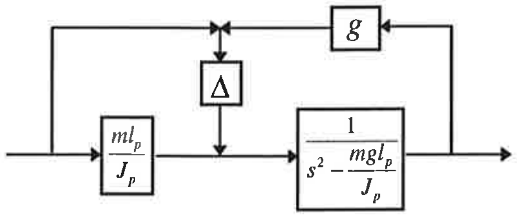


Figure 33: Model of a Numerator Denominator perturbation.

This setup turned out to be easier to both control and to make diagnosis on - even with an uncertainty of 30%.

6.3.1 Low frequency diagnosis

Another thing in common for the three design examples above is that the diagnostic weight W_d is a rather steep band pass filter, just above the process bandwidth. One idea is to let W_d be a low pass filter and a track the fault at all times - not just when there is a high frequent change. This turned out to be very hard when the uncertainty was on the input or output. To get a good alarm signal at low frequencies the control performance had to be made quite bad.
The numerator-denominator uncertainty turned out to work well with low frequency diagnosis. In the following example there is an uncertainty of 30% in the "combined" parameter ml_p/J_p. The weight on the control errors has been made faster. This also goes for the weight on the fault, which means that the resulting system is robust against larger errors. Figure 34 shows simulation results of this setup. The inputs to the system are the same as in the previous simulations of the pendulum with the fault on the input. The control is relatively fast and insensitive to faults. The alarm signal tracks the fault very good.

Tracking a fault on the pendulum angle sensor at low frequencies turned out to work just as good as tracking the actuator fault. As expected, a low frequency tracking of a fault on the arm position sensor was not possible. Therefore the "band pass approach" has to be used for this fault. A combined diagnosis, i.e. using the band pass for arm sensor faults and low frequency detection for the pendulum angle sensor, did not work well - it was difficult to avoid false alarms on the alarm signal for the arm position.

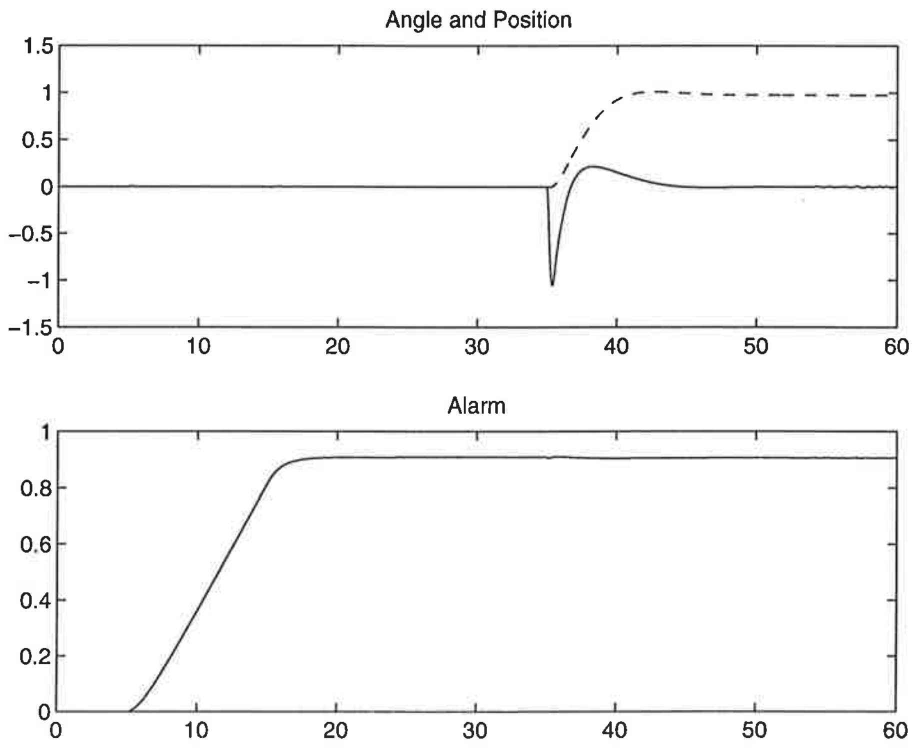


Figure 34: Simulation result of pendulum with 30% numerator-denominator uncertainty and low frequency diagnosis. Top: Pendulum angle (solid, multiplied by 100) and arm position (dashed). Bottom: Alarm.

6.4 Choosing weights

In Chapters 5 and 6 the different weights have been discussed a lot - moving them to higher or lower frequencies, changing the bandwidth or the gain. This section is a short summary of the ideas and thoughts behind the weights.

W_a is the weight on the (unfiltered) alarm signal a. Throughout this thesis it has been roughly unchanged. It is not allowed to skip it totally, since a is a control signal and must be punished. Choosing W_a to be a small constant does not alter the simulation results received in this thesis very much. The raise of W_a at higher frequencies used here is there to suppress high frequency noise in the alarm signal. This can also be achieved by filtering the high frequencies of a harder. W_a is not allowed to roll off at high frequencies.

W_c is the weight on the control performance. It specifies how fast the control is supposed to be, and the size of the steady state error. An integrator can be introduced in the controller by letting W_c get a higher and higher gain at low frequencies. Achievable control bandwidth and steady state error are limited by the uncertainty - an uncertainty on the output imposes large limitations, while the numerator-denominator uncertainty imposes smaller. The diagnostics also impose limitations on the control performance. If the control is emphasized too much, false alarms on reference changes will be the case. An example of this is shown in . The weights used in this design are, with the exception of W_c which is made faster and larger, the same as in the design in Chapter 6.2.1. The alarm signal shows a large false alarm on the reference change.

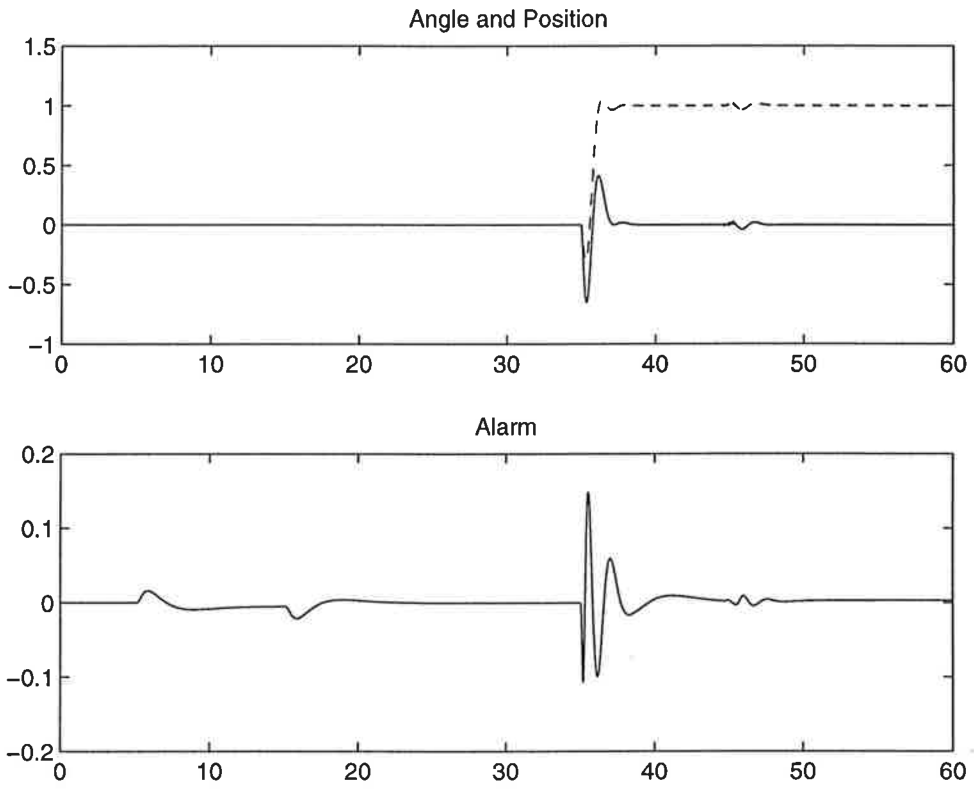


Figure 35: Simulation results of the pendulum with a 10% input uncertainty and actuator fault. The control is emphasized too much - thus the false alarm at t=35. Top: pendulum angle (solid, multiplied by 10) and arm position (dashed). Bottom: Alarm signal.

W_d is the diagnostic weight. It can primarily be chosen in two ways: as a low pass filter, making the alarm signal track the fault at low frequencies, or as a band pass filter making the alarm signal detect changes of higher frequencies in the fault.
The band pass approach can be used with all of the uncertainties and faults used in this thesis. Depending on where the fault occurs W_d can be placed at different frequencies - when the fault was on the input W_d could be moved to lower

frequencies, but when the fault was on the output W_d had to be chosen more high frequent.
The low pass approach can only be used with uncertainties that are not too severe, like the numerator-denominator perturbation. When the low pass approach was used on a pendulum with a 10% input uncertainty, the trade-off between control and diagnosis for uncertain plants became apparent - either control or diagnostic performance had to be to be made worse. One other condition that has to be fulfilled in order for the low pass approach to work is that the fault must be observable.

W_f is a weight on the incoming fault. Making W_f larger causes the resulting system to be more robust against faults. In this thesis W_f has been a first order low pass filter. It is possible to let it be a constant gain, but this makes it harder to achieve robust performance. When the fault is on the input W_f can be chosen to have a large static gain. If the fault is on the output W_f must be chosen smaller.

W_n specifies the amount of and frequency content of measurement noise. This weight is not allowed to roll off at high frequencies.

W_u is the weight on the control signal, and must always be present. It also sets the limit of the input uncertainty. (e.g. W_u=0.1 gives a 10% uncertainty)

W_y is used as a limit on (if there is any) the output uncertainty, and a weight W_{delta} is used for the numerator-denominator uncertainty.

It is not only the absolute value of the weights that decide the resulting performance. The relative size is just as important. If (e.g.) a design shows a nice control behaviour but a somewhat poor alarm signal, raising W_d can result in poor control. This trade-off and other that have to be taken show up automatically when designing a control and diagnostic module. It turned - for example - out to be impossible to get a good diagnosis on both sensors and the actuator at the same time. The alarm signal also had to traded off against the size of the uncertainty. A large uncertainty makes it hard to get good diagnosis without false alarms.

6.5 Limitations

One limitation in this thesis is that the controllers have not been sampled. Sampling the controllers would make it easier to make simulations with noise. Simulating continuous systems with noise is very hard. Some attempts were made to do this, but without any greater success.
To be able to sample the controllers they would have to be reduced with some model reduction technique. Also here some unsuccessful attempts were made - the reduced controllers did not become small or accurate enough. Trying to reduce the controllers I experienced some numerical problems in μ-tools - the command `szeros` which is supposed to compute the transmission zeros of a system gave different results when called twice consecutively.

7 Conclusions

The four degree of freedom controller can be used to design integrated control and diagnostic modules for plants with uncertainties, but the design procedure is very difficult - a lot of different weights have to be chosen.
The achievable control and diagnostic performance is limited by the uncertainty and depends on where and how the faults occur in the plant. The limits vary considerably with respect to the uncertainty. The faults that can be detected are mainly additive - multiplicative faults should preferably be detected with some other method, for example parameter identification.
The H_∞ – synthesis provides an optimally robust controller for the formulated problem, but there is no guarantee that the control and diagnostic performance is optimal, or even good. The results will never be better than the chosen weights, why a lot of work has to be done on choosing them so that all requirements are met. To do this takes a lot of knowledge about the plant and the faults that are likely to occur. One other problem is that the obtained controllers often are of very high order, and have to be reduced before they are implemented. It is also possible that some other norm could be used for the design, i.e. H_2 or L_1.

8 Literature

[1] G. J. Balas, J. C. Doyle, K. Glover, A. Packard and R. Smith: μ-Analysis and Synthesis Toolbox (μ-Tools). MatLab Functions for the analysis and Design of Robust Control Systems. The Mathworks, Inc., 1991.

[2] B. Bernhardsson: The H_∞-Approach. EURACO Conference on Robust and Adaptive Control, Dublin 1994.

[3] J. C. Doyle, K. Glover, P. P. Khargonekar and B. A. Francis: State-Space Solutions to Standard H_2 and H_∞ Control Problems. IEEE Transactions on Automatic Control, August 1989.

[4] P. M. Frank: Fault Diagnosis in Dynamic Systems Using Analytical and Knowledge-based Redundancy - A Survey and Some New Results. Automatica, Vol. 26, 1990.

[5] M. Green and D. Limebeer: Linear Robust Control. Prentice Hall 1994.

[6] R. Isermann: Process Fault Detection Based on Modeling and Estimation Methods - A Survey. Automatica, Vol. 20, 1984.

[7] C. A. Jacobsson and C. N. Nett: An Integrated Approach to Controls and Diagnostics Using the Four Parameter Controller. IEEE, October 1991.

[8] M. Tyler and M. Morari: Optimal and Robust Design of Integrated Control and Diagnostic Modules. Proceedings of the ACC, June 1994.

[9] K. J. Åström and K. Furuta: Swinging Up a Pendulum by Energy Control. To appear at IFAC World Congress, 1996.

[10] H. Kwakernaak: Robust Control and H_∞-Optimization - Tutorial Paper. Automatica, Vol 29, March 1993.

Appendix A

The MatLab toolbox μ-Tools provides commands for H_∞- and μ-synthesis. It also provides several commands for connecting different subsystems to a larger interconnection structure, and commands for analyzing the resulting controllers and control loops. This appendix is only meant to be a short presentation of the commands used in the thesis. For a more thorough presentation see [1].

In μ-Tools a system can be represented in two ways:

- as a SYSTEM matrix, or
- as a VARYING matrix.

Also there is the possibility of storing things as an ordinary matrix - a CONSTANT matrix.

A system described by the state space representation

$$\begin{aligned}\dot{x} &= Ax + Bu \\ y &= Cx + Du\end{aligned}$$

is represented as a SYSTEM matrix as

$$\begin{bmatrix} A & B & nx \\ C & D & 0 \\ 0 & 0 & -Inf \end{bmatrix}$$

where nx is the number of states, and *-Inf* indicates to the μ-Tools functions that the matrix is a SYSTEM matrix.

A VARYING matrix is a time or frequency response of for example a SYSTEM matrix, and is represented as

$$\begin{bmatrix} [G_1] & x_1 \\ \vdots & \vdots \\ [G_i] & x_N \\ & 0 \\ \vdots & \vdots \\ [G_N] & 0 \\ 0 \cdot\cdot 0\ N & Inf \end{bmatrix}$$

where *Inf* in the bottom right corner indicates that this is a VARYING matrix. G_i is the response at point *i* and $x_1 - x_N$ is the independent variable. *N* indicates how many points are represented.

Building an interconnection

The following commands are used to create an interconnection for the design of a control system.

sys = pck(A, B, C, D)

creates the SYSTEM matrix sys of the matrices A, B, C and D from the state space representation.

[A, B, C, D] = unpck(sys)

returns A, B, C and D.

sys = nd2sys(num, den, gain)

converts a SISO numerator/denominator transfer function to a SYSTEM matrix. num and den are vectors containing the parameters of the respective polynomials.

sys = zp2sys(zeros, poles, gain)

converts zeros and poles of a SISO transfer function to a SYSTEM matrix. zeros and poles are vectors containing the wanted zeros and poles respectively.

sys = daug(mat1, mat2, ... , matN)

is best described by Figure 36. daug takes smaller systems and puts them in a bigger SYSTEM matrix. This can be very helpful when building larger MIMO systems.

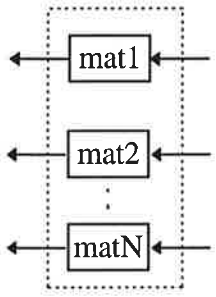


Figure 36: daug(mat1, mat2, ... , matN)

out = minv(sys)

calculates the inverse of the matrix sys, which can be a CONSTANT, VARYING or SYSTEM matrix.

out = mmult(mat1, mat2, .. matN)

multiplies the matrices mat1-matN according to Figure 37.

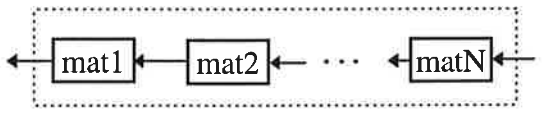


Figure 37: mmult(mat1, mat2, .. , matN)

sysic

forms linear interconnections of SYSTEM and CONSTANT matrices, by writing the loop equations. To use sysic one has to set up several variables in the MatLab workspace. The following is a list of the required variables.

- systemnames is a character string containing the names of the matrices used in the interconnection. Each named system must exist in the MatLab workspace before sysic is run. The names are limited to 10 characters, and should be separated by spaces or tabs.
- inputvar is a character string containing the names of the external inputs that are present in the final interconnection. The names are separated by semicolons, and the entire list is enclosed by square brackets []. A multivariable input is defined by a number enclosed by curled brackets. For example noise{3} means that the input noise is a vector of three signals.
- outputvar is a character string describing the outputs of the interconnection. The outputs are formed of linear combinations of the outputs of the subsystems and the external inputs. For multivariable

systems, arguments within parenthesis specify which outputs are to be used and in what order.

- input_to_sys is a character string which defines the input to the subsystem sys. If the subsystem name is redlight, then the variable must be called input_to_redlight. There must be such a string for every subsystem in the interconnection. It is specified in the same manner as outputvar. Separate channels are separated by semicolons, and the order of the inputs in the variable should match the order of the inputs in the system itself.
- sysoutname is an optional character string which defines what the resulting interconnection is supposed to be called. If sysoutname does not exist, then the interconnection is called ic_ms.
- cleanupsysic is an optional character string. If it is set to 'yes' the variables mentioned above will be cleared when sysic is finished. The default value is 'no'. The subsystems used in the interconnection are not cleared.

Commands for analyzing a system

out = frsp(sys, omega)

calculates the frequency response of the system in the SYSTEM matrix sys for the given vector of frequency points omega (for example created with logspace). The output matrix out is a VARYING matrix.

vplot('axis', vmat1, vmat2, ...)

plots multiple VARYING matrices in the same graph. The optional argument axis specifies the plot type, and can for example be

• iv,d	matrix vs. independent variable
• liv,m	magnitude vs. log independent variable
• liv,lm	log magnitude vs. log independent variable
• nyq	Nyquist plot
• bode	Bode plot.

The default value of axis is iv,d.

out = spoles(sys)

calculates the eigenvalues of the SYSTEM matrix sys.

out = szeros(sys)

calculates the transmission zeros of the SYSTEM matrix sys.

see(sys)

displays the SYSTEM matrix sys by showing the A, B, C and D matrices.

Commands for controller synthesis

[k, g, gfin, ax, ay, hamx, hamy] = hinfsyn(p, nmeas, ncon, gmin, gmax, tol)

calculates a H_∞-control law which achieves the H_∞-norm gfin for the interconnection structure p. The control law k stabilizes the system matrix p and has the same number of states as p. The closed loop is returned in g. ax, ay, hamx and hamy is information about the solutions of the two Riccati equations mentioned in chapter 3. nmeas is the number of measurements the controller uses and ncon the number of control signals. gmin and gmax are the upper and lower limits for the γ-iteration. The iteration stops when the relative difference between test γ-value and previous γ-value is less than tol. Except for those arguments there are three optional ones : ricmethod, epr and epp. They are not described here.

[bnds, dvec, sens, pvec] = mu(matin, blk, options)

computes the upper and lower bounds for the structured singular value μ of the VARYING matrix matin. The underlying uncertainty structure is blk. options lets the user decide which methods are to be used for the calculations. The output bnds contains the upper and lower bounds of μ. dvec contains the D scaling matrices that have produced the upper bound in bnds. sens is a sensitivity measure of the maximum singular value of dleft*matin*dright^{-1} with respect to dleft and dright. It is mainly used in musynfit. pvec contains a perturbation matrix Δ which makes I-matin*Δ singular.

[dleft, dright] = unwrapd(dvec, blk)

The D scales contain a lot of zeros, and are stored in vector form (in dvec) to save memory. They are “inflated“ to their proper size with unwrapd.

[dsysL, dsysR] = musynfit(pre_dsysL, dvec, sens, blk, nmeas, ncon)

approximates the D scales in dvec with stable minimum phase SYSTEM matrices dsysL and dsysR. The first time it is run pre_dsysL must be set to the character string ‘first’, and then the previous dsysL. sens is the sensitivity mentioned above and blk is the underlying uncertainty structure. nmeas and ncon are the same as in hinfsyn.

Appendix B

```
% Defines weights, creates the interconnection and iterates.
% The linear pendulum model is assumed to be present in the SYSTEM
% matrix "Process". There is an uncertainty of 10% on the input. The fault
% is also on the input.

% This is the weight for the control signal. It is also the limit for the uncertainty.
Wu=0.1;

% For punishing the alarm signal
Wa=nd2sys([1000, 1],[1, 1000], 0.0005);

% Reference tracking/control performance
Wctheta=nd2sys([50], [100, 1]);                       % For the pendulum angle
Wcphi=nd2sys([1], [100, 1]);                  % For arm position
Wc=daug(Wctheta, Wcphi);                      % Put them together to one weight.

% Diagnostic filter
Wd=zp2sys([-.08+.03*i, -.08-.03*i, -.00001], [-14,  -15,  -15,  -1,  -1,  -1], 6000);
% A band pass filter with poles and zeros at the specified locations.
[WdA, WdB, WdC, WdD]=unpck(Wd);               % The post-filter for the simulations.

% Noise filter
Wn=.01*eye(2);                         % Static gain for the noise

% The fault filter
Wf=nd2sys(10, [1 .01]);                       % Low pass filter for the fault.

systemnames = ' Process Wa Wc Wd Wu Wf Wn';
% Specifies which blocks are in the interconnection.

inputvar ='[pertin{1} ; f{1} ; n{2}; r{1}; a{1} ; u{1}]';
% Specifies the inputs and their order.

outputvar ='[Wu ; Wc ; Wd ; Wa ; r ;Process + Wn]';
% Specifies the outputs and their order.

input_to_Process='[u +Wf + pertin ]';
input_to_Wf='[ f ]';
input_to_Wn='[ n ]';
input_to_Wu = '[ u ]';
input_to_Wc = '[Process(1);Process(2)-r]';
input_to_Wa = '[ a ]';
input_to_Wd = '[ a-Wf ]';
% Specifies the inputs to each block.

sysoutname='System';                          % The wanted interconnection name.
cleanupsysic='yes';                           % Remove unnecessary variables.
```

```
sysic;                                    % Create the interconnection.

%-----------------------------------------------------
% DK-iteration.

% Variables for the DK-iteration
Nmeas=3;                                  % Number of measurements
Ncon=2;                                   % Number of control outputs
blk=[1 1 ;4 4];                           % Uncertainty structure.

omega=logspace(-3, 3, 30);                % A vector of frequencies

[Knew, Gnew]=hinfsyn(System, Nmeas, Ncon, 90, 110, 0.1);
%Compute the first controller

Gnewg=frsp(Gnew, omega);
%Compute response of closed loop.

[bndsnew, dvec, sens, pvec]=mu(Gnewg, blk);
% Compute the D-scales.

[dsyslnew, dsysr]=musynfit('first', dvec, sens, blk, Nmeas, Ncon);
% Fit the D-scales with a dynamic system

muic=mmult(dsyslnew, System, minv(dsysr));
% Incorporate the dynamic system in the interconnection.

%----------------------------------------------------------
% Perform second iteration

[Knew, Gnew]=hinfsyn(muic, Nmeas, Ncon, .5, 2, .01);

Gnewg=frsp(Gnew, omega);

[bndsnew, dvec, sens, pvec]=mu(Gnewg, blk);

[dsyslnew, dsysr]=musynfit('first', dvec, sens, blk, Nmeas, Ncon);

muic=mmult(dsyslnew, System, minv(dsysr));

%-------------------------------------------------------------
% Compute the last (final) controller

%[Knew, Gnew]=hinfsyn(muic, Nmeas, Ncon, .1, 1, .1);


[kA, kB, kC, kD]=unpck(Knew);
% Put the controller on the workspace in a form that can be used by Simulink.
```