Sequential ellipsoidal unfalsified MIMO control

Sequential Ellipsoidal Unfalsified MIMO Control

M.F. Ypma

DCT 2006.90

DCT report

Technische Universiteit Eindhoven
Department Mechanical Engineering
Dynamics and Control Technology Group

Eindhoven, September, 2006

Contents

Chapter 1

Introduction

In this report, 'ellipsoidal unfalsified control' (EUC), as developed by [1] for single-input, single-output (SISO) systems, is extended for use on multi-input multi-ouput (MIMO) systems. EUC is a control design method that finds a controller capable of meeting a certain performance requirement for an unknown SISO system. To achieve this, only in- and outputdata are used without knowledge of the plant that is controlled. In this report a first attempt is made to expand this theory for use on any given plant. In chapter 2 the concept of EUC is explained and the theory that supports EUC is given. In chapter 3 this theory is adapted for MIMO systems in an approach with sequential updating of the controller. In chapter 4 the chosen structure is examined in simulations on a simple plant. Finally, a conclusion is given.

Chapter 2

EUC for SISO systems

2.1 The principles of EUC

Unfalsified control is a control design method that finds a controller for an unknown system by using merely input and output data. The measured data of the input and output are used to check wether sets of controller parameters fail to meet a performance requirement, and are therefore falsified. Initially, an arbitrary choice is made for the values of the controller parameters that are used in the controller. Also, because no data is available, all other values (which are of course infinitely many) must theoretically not yet be falsified, i.e. are *un*falsified. EUC defines an ellipsoid in a space spanned by the controller parameters (the parameter-space) as the space that consists of all values which are unfalsified. Because this ellipsoid has to be of finite dimensions for the algorithm to work, a choice is initially made for the shape and the center of this ellipsoid. So, not an infinite number of values are unfalsified, but just a space big enough to ensure that there exists an optimal parameterset inside of the initial ellipsoid. Obviously, the parameter set that is initially used in the controller has to be within this ellipsoid.

The performance requirement is a maximum allowed tracking error, which in general gradually decreases in time, leaving less sets of control parameters unfalsified. The formulation of this performance requirement is developed such that it generates two parallel half-spaces in the parameter-space, consisting of parameters which fail to meet the performance requirement, which can cut through the ellipsoid. A new, smaller, ellipsoid is then defined by the minimal-volume outer-bounding ellipsoid through the points of intersection (see Fig. 2.1 for a 2D interpretation). The performance requirement gradually decreases (instead of instantaneously demanding a minimal tracking error) to take into account the start-up behavior and disturbances. The ellipsoid can be updated without changing the actual controller parameters, but to achieve good dynamic behavior as fast as possible, a new set of controller parameters is taken from the ellipsoid once the set used in the controller lies outside the new ellipsoid and is falsified.

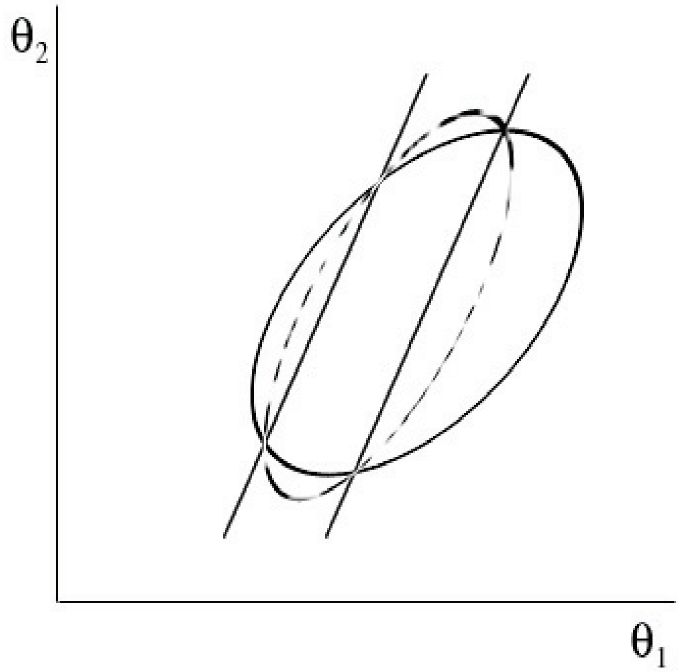

Figure 2.1: *Solid line ellipsoid containing unfalsified parameter values that is cut by two half-spaces (two straight lines), defining new ellipsoid (dashed line)*

2.2 The extension to MIMO

The existing algorithm which calculates the new ellipsoid and the new set of controller parameters which are to be implemented, is unchanged for MIMO systems. However, the development of the input, as well as the use of the output of the algorithm is different. Because the plant model is unknown, using a SISO EUC controller for every input is inadequate. This would assume that the MIMO plant is diagonal, i.e. an input only has influence on one output, and it is the only one to influence that output. In MIMO systems, this is generally not the case. In fact, several inputs can affect a single output (directionality) and one input can affect several outputs (dispersion) (see Fig. 2.2).

Figure 2.2: *Dispersion and directionality*

2.3 Fictitious reference

In this section, the concept and purpose of a fictitious reference is explained. Consider a closed loop feedback system as in Fig. 2.3, where $r(t_k)$ is the reference, $u(t_k)$ the plant input, and $y(t_k)$ the plant output.

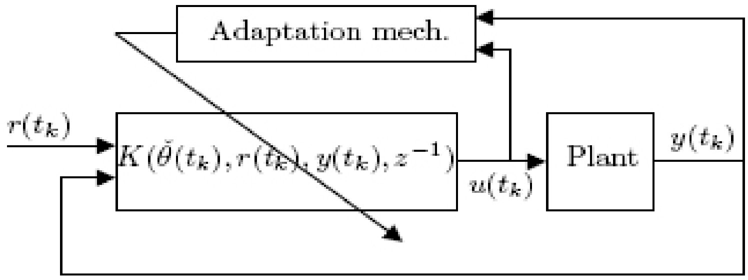

Figure 2.3: *Closed loop feedback system with adaptation of the controller parameters*

Input $u(t_k)$ can typically be written as

$$u(t_k) = K(\bar{\check{\theta}}, r(t_k), y(t_k), z^{-1}) \tag{2.1}$$

with $\bar{\check{\theta}}$ being a set of controller parameters which are used in the controller, and z^{-1} the discrete time shift operator (by definition: $z^{-1} * t_k = t_{k-1}$, so $u(t_k)$ can also depend on past values of $r(t_k)$ and $y(t_k)$). The fictitious reference is now defined by rewriting (2.1) for $r(t_k)$. This can be done by first dividing K in two functions, K_r and K_y

$$K(\bar{\check{\theta}}, r(t_k), y(t_k), z^{-1}) = K_r(\bar{\check{\theta}}, z^{-1}) * r(t_k) + K_y(\bar{\check{\theta}}, y(t_k), z^{-1}) \tag{2.2}$$

from which the fictitious reference is stated as

$$r_{\text{fict}}(\bar{\theta}, z^{-1}) = K_r^{-1}(\bar{\theta}, z^{-1}) * (u(t_k) - K_y(\bar{\theta}, y(t_k), z^{-1})) \tag{2.3}$$

r_{fict} is a function of the *candidate* controller parameters that are to be evaluated: $\bar{\theta}$, not to be mistaken with $\bar{\check{\theta}}$, the set of parameters which is actually in the controller. Implementing $\bar{\check{\theta}}$ in (2.3) obviously results in (2.2). All fictitious references made up by different parameter sets can be compared to the real reference, which is done in the performance requirement, which then defines a region of unfalsified parameter values (as said, the region between two parallel half-spaces), by using just the input and output data.

2.4 Unfalsified region: the ellipsoid

Now consider a space spanned by $\bar{\theta}$ in which an ellipsoid is defined.

$$(\bar{\theta}-\bar{\theta}_c)^T\Sigma^{-1}(\bar{\theta}-\bar{\theta}_c) \leq 1 \tag{2.4}$$

Herein, $\bar{\theta}_c$ is the center of the ellipsoid, and the matrix Σ defines the shape of the ellipsoid. The space within this ellipsoid is defined to consist of all $\bar{\theta}$ which are unfalsified.

2.5 Performance requirement

The performance requirement is given by

$$-\Delta(t_k) \leq G_{\text{m}} r_{\text{fict}} - y(t_k) \leq \Delta(t_k) \tag{2.5}$$

from which then the area of unfalsified parametervalues U can be defined:

$$U = \left\{\bar{\theta} \mid -\Delta(t_k) \leq G_{\text{m}} r_{\text{fict}} - y(t_k) \leq \Delta(t_k)\right\} \tag{2.6}$$

G_m is a reference model, defining the desired closed loop dynamics. So $G_m r_{fict}$ is a fictitious y and is compared to the measured y, resulting in a tracking error. Bound $\Delta(t_k)$ is the maximum allowed tracking error, which depends on t_k, because it is chosen to decrease with time. Equation 2.5 represents two parallel half-spaces which can cut through the ellipsoid of the previous section, as shown in Fig. 2.1.

r_{fict} can be written as

$$r_{\text{fict}}(\bar{\theta}, z^{-1}) = w(u(t_k), y(t_k), z^{-1})\bar{\theta} \tag{2.7}$$

with w given in a general notation

$$w(u(t_k), y(t_k), z^{-1}) = \begin{bmatrix} u(t_k) \\ \Lambda_u(z^{-1}) * u(t_k) \\ \Lambda_y(z^{-1}) * y(t_k) \\ f(u(t_k), y(t_k), z^{-1}) \end{bmatrix} \tag{2.8}$$

with $\Lambda_u(z^{-1})$ and $\Lambda_y(z^{-1})$ being vectors of stable linear filters, and $f(u(t_k), y(t_k), z^{-1})$ being a vector of non-linear functions. In this form (2.7) r_{fict} is substituted in (2.5) to obtain

$$PR = \left\{\bar{\theta} \mid -1 \leq \phi_k^T\bar{\theta} - y_k \leq 1\right\} \tag{2.9}$$

where

$$\phi_k = \frac{G_{\text{m}} w(u(t_k), y(t_k), z^{-1})}{\Delta(t_k)}$$

$$y_k = \frac{y(t_k)}{\Delta(t_k)}, \quad for \quad \Delta(t_k) > 0$$

The new ellipsoid is calculated from the intersection of (2.9) with the ellipsoid of (2.4), and the new set of actual controller parameters is chosen from this ellipsoid using the methods of (J. van Helvoort, 2005)). This part of the theory does not need to be altered for use on MIMO systems. In the next section it is shown that a typical ϕ_k and y_k can also be formulated for MIMO, which can be used for the calculation.

Chapter 3

Extended EUC for MIMO systems

3.1 Introduction

In the previous section, the general procedure of EUC is recalled. As said, in this section ϕ_k and y_k are derived for MIMO use. It is important to realize that the theory defines the controller structure as in (2.1), in relation to one input. Being able to directly derive formulations for fictitious references, the same is done for MIMO. So a setup with a separate controller for every input is chosen. Every controller will however be evaluated for every reference. This sequential evaluation is a logical consequence of section 2.2, but it also follows from the theory given in this section. A drawback of this choice is that there is no correlation implemented between controllers, while they have the same structure. A different approach of the problem would be to design one adaptable controller which makes a matrix-wise evaluation of a parameterset which controls every input at once, considering all references and all outputs as vectors.

3.2 Controller structure

A system with 2 inputs (u_1, u_2), 2 outputs (y_1, y_2) and 2 references (r_1, r_2) is considered. The inputs can be written in discrete time as

$$\begin{aligned} u_1(t_k) &= K(\bar{\tilde{\theta}}_1(t_k), r_1(t_k), r_2(t_k), y_1(t_k), y_2(t_k), z^{-1}) \\ u_2(t_k) &= K(\bar{\tilde{\theta}}_2(t_k), r_1(t_k), r_2(t_k), y_1(t_k), y_2(t_k), z^{-1}) \end{aligned} \tag{3.1}$$

Because no plant model is known, an *equal* dependency on both references and on both outputs must be assumed for u_1 and u_2 to consider full dispersion and directionality.[1] Therefore, u_1 and u_2 can be formulated with the same function 'K',

[1]For instance, the plant can be of the following basic structures

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \tag{3.2}$$

using a different set of control parameters.

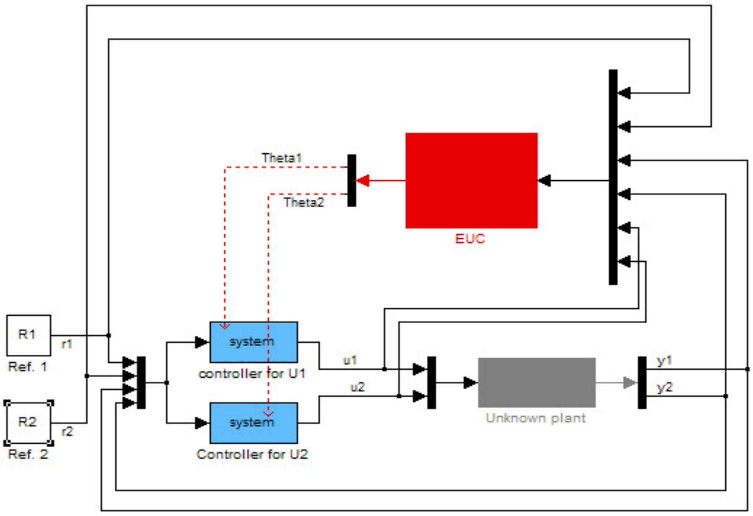

Figure 3.1: *Controller structure; controller parameters 'Theta1' and 'Theta2' are outputs of the EUC algorithm and input for the variable controllers.*

As can be deduced from (3.1) and be seen in Fig. (3.1), a separate controller for each input is used. They have the same structure, but use different parameter sets. The parameter sets are all an output of the same single EUC-algorithm (the red box). The EUC-algorithm uses all signals $(u_1, u_2, y_1, y_2, r_1, r_2)$ as input. How these are used to calculate the parameter sets, is explained furtheron.

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \tag{3.3}$$

For (3.2), $u_1 = y_1$, so the controller structure must be so that u_1 is a function of only r_1. For (3.3), one can see that the 'opposite' case is desirable. With the right choice for $\bar{\theta}_1$ and $\bar{\theta}_2$, (3.1) can result in both, if r_1 and r_2 are represented equally in 'K'.

3.3 Sequential ellipsoidal unfalisified control method

3.3.1 Fictitious reference

A fictitious reference is given, analogous to section 2.3. Considering only u_1, $r_{1,fict}$ and $r_{2,fict}$ can both be formulated. These are dependent on the same controller parameter set $\bar{\theta}_1$. As shown furtheron, this parameter set will be evaluated sequentially for $r_{1,fict}$ and $r_{2,fict}$ using the ellipsoidal unfalsified control method.

Consider a simple choice for $K(\bar{\check{\theta}}_1(t_k), r_1(t_k), r_2(t_k), y_1(t_k), y_2(t_k))$

$$u_1(t_k) = \frac{1}{\check{\theta}_1}(-\check{\theta}_2 y_1(t_k) - \check{\theta}_3 y_2(t_k) - \check{\theta}_4 r_1(t_k) - \check{\theta}_5 r_2(t_k)), \check{\theta}_1 \neq 0 \tag{3.4}$$

with

$$\bar{\check{\theta}}_1 = [\check{\theta}_1, ..., \check{\theta}_5]^T$$

Then $r_{1,fict}$ and $r_{2,fict}$ are given by

$$r_{1,\text{fict}} = \frac{1}{\theta_4}(-\theta_1 u_1(t_k) - \theta_2 y_1(t_k) - \theta_3 y_2(t_k) - \theta_5 r_2(t_k)) \tag{3.5}$$

$$r_{2,\text{fict}} = \frac{1}{\theta_5}(-\theta_1 u_1(t_k) - \theta_2 y_1(t_k) - \theta_3 y_2(t_k) - \theta_4 r_1(t_k)) \tag{3.6}$$

Notice that substituting $\check{\theta}_i$ for θ_i in (3.5) and (3.6) exactly results in (3.4). The theorem is being developed for the controller for u_1. It can, due to the similarity, directly be implemented for u_2. Therefore $\bar{\theta_1}$ from (3.1) will from now on be referred to as $\bar{\theta}$, to simplify the syntax.

3.3.2 Performance requirement

Recall from section 2.5 that the region unfalsified by the performance requirement can be interpreted as two parallel half-spaces which can cut through the ellipsoid consisting of all unfalsified values. A new, smaller, ellipsoid is then defined by the minimal volume outer-bounding ellipsoid through the points of intersection (see Fig. 2.1 for a 2D interpretation)

$r_{1,fict}$ and $r_{2,fict}$ are both a function of θ and are used to define two separate performance requirements, which will be considered sequentially. Unfortunately, implementing (3.5) in this equation does not express two parallel five-dimensional half-spaces, because in (3.5), $\theta_1, \theta_2, \theta_3$ and θ_5 are divided by θ_4. The half-spaces are not defined in the same space spanned by the five parameters, in which the ellipsoid is defined. Eq. (3.5) must be rewritten so that

$$r_{1,\text{fict}} = \bar{w_1}^T \bar{\theta} \tag{3.7}$$

with

$$\bar{\theta} = [\theta_1, ..., \theta_5]^T$$

Therefore, we choose θ_4 to be constant ($\check{\theta}_4$) and define w_1 as stated below

$$w_1 = \frac{1}{\check{\theta}_4} \begin{bmatrix} -u_1(t_k) \\ -y_1(t_k) \\ -y_2(t_k) \\ 0 \\ -r_2(t_k) \end{bmatrix} \tag{3.8}$$

By doing this, θ_4 will not contribute in the evaluation. Parameter θ_4 will however be evaluated with the performance requirement for $r_{2,fict}$ as seen furtheron in (3.11). θ_5 on the other hand is not evaluated in that equation, but is already evaluated for $r_{1,fict}$. The value of this constant can be chosen anywhere in the ellipsoid. Because the algorithm that calculates the new ellipsoid already outputs a parameterset that is to be used in the controller ($\bar{\bar{\theta}}$), the value for the constant is taken from this set.

Notice that implementing (3.8) in (3.7) and substituting θ_4 for $\check{\theta}_4$, exactly results in (3.5). By substituting (3.8) in (2.5) and dividing by Δ we obtain

$$U_1 = \left\{ \bar{\theta} \mid -1 \leq \frac{G\mathrm{m} w_1}{\Delta_1(t_k)} \bar{\theta} - \frac{y_1(t_k)}{\Delta_1(t_k)} \leq 1 \right\} \tag{3.9}$$

For $r_{2,fict}$, we can develop a similar region of unfalsified parametervalues

$$U_2 = \left\{ \bar{\theta} \mid -1 \leq \frac{G\mathrm{m} w_2}{\Delta_2(t_k)} \bar{\theta} - \frac{y_2(t_k)}{\Delta_2(t_k)} \leq 1 \right\} \tag{3.10}$$

with

$$w_2 = \frac{1}{\check{\theta}_5} \begin{bmatrix} -u_1(t_k) \\ -y_1(t_k) \\ -y_2(t_k) \\ -r_1(t_k) \\ 0 \end{bmatrix} \tag{3.11}$$

Analog to $r_{1,fict}$, multiplying w_2 with $\bar{\theta}$, and substituting θ_5 for $\hat{\theta}_5$, exactly results in (3.6).

These two performance requirements are now both applicable on the ellipsoid of the chosen example of (3.4). So within one sample time, θ_1, θ_2 and θ_3 are evaluated twice, first for $r_{1,fict}$ and then for $r_{2,fict}$, while θ_4 and θ_5 are evaluated only once, resp. for $r_{2,fict}$ and $r_{1,fict}$.

A more general description of the region of unfalsified parametervalues is given by

$$U_i = \left\{ \bar{\theta} \mid -1 \leq \frac{G\mathrm{m} w_i}{\Delta_i(t_k)} \bar{\theta} - \frac{y_i(t_k)}{\Delta_i(t_k)} \leq 1 \right\}; i = 1, ..., NoR \tag{3.12}$$

Where NoR is number of references, with

$$w_i = \frac{1}{\hat{\theta}_j}\begin{bmatrix} -u_1 \\ \Lambda_u(z^{-1}) * u_1(t_k) \\ -y_1(t_k) \\ \Lambda_{y1}(z^{-1}) * y_1(t_k) \\ . \\ . \\ . \\ -y_i(t_k) \\ \Lambda_{yi}(z^{-1}) * y_i(t_k) \\ -r_1(t_k) \\ . \\ . \\ . \\ -r_i(t_k) \end{bmatrix} ; w(j) = 0; i = 1, ..., NoR, \hat{\theta}_j \neq 0 \tag{3.13}$$

where j is the element in w which represents r_i. In this formulation, Λ represents a vector of filters. That means the controller can have any possible structure. Equation (3.8) is a simplified version hereof. This formulation is general for the number of references and outputs (which are generally the same). Outputs that are not directly being referenced and controlled can still be used for control on an other output, for instance using knowledge about speed to control position. The formulation can also be used for every input, because as seen in (3.1) the structure 'K' is used for every input. It is thus applicable for an arbitrary number of inputs and outputs.

3.3.3 Use of the EUC algorithm

The EUC algorithm, as developed by (J. van Helvoort, 2005) uses $\frac{G_m w_i}{\Delta_i(t_k)} = \phi_k$ and $\frac{y_i(t_k)}{\Delta_i(t_k)} = y_k$ from eq. (3.12) and an initial ellipsoid as input, and outputs a new ellipsoid and the set controller parameters from this ellipsoid that can be implemented in the actual controller. The algorithm is then run again, but it now uses the new ellipsoid as input, and a different ϕ_k and y_k, which are calculated from the next reference. This is repeated for every reference (i times). This is all done within one sample time.

Chapter 4

Simulation results

4.1 Introduction

In the previous chapter, a theory is derived for EUC on MIMO systems. In this section, the developed theory will be applied to a simulated model, similar to fig. 3.1. The behavior of the controller for different 'unknown' plants is examined. The simulation model is created as a Simulink model in MATLAB.

4.2 The Simulink model

4.2.1 Controller structure

The controller structure which is used throughout the simulations is of the following form.

$$w = \begin{bmatrix} -u_1(t_k) \\ \frac{1}{z} - u_1(t_k) \\ -r_1(t_k) \\ -r_2(t_k) \\ -y_1(t_k) \\ \frac{1}{z} - y_1(t_k) \\ -y_2(t_k) \\ \frac{1}{z} - y_2(t_k) \end{bmatrix} \tag{4.1}$$

The constant $\frac{1}{\hat{\theta}_j}$ is put out, so it slightly differs from the original form as in for example (3.13). The constant can be placed anywhere in the corresponding term in (3.12) and for reasons of ease it is moved under the fraction bar and calculated together with the error bound.

4.2.2 The error bound

The error bound (first occuring in (2.5)) is chosen as

$$\Delta_i(t_k) = c_i + b_i e^{a_i t_k}, a_i < 0, b_i \geq 0, c_i \geq 0, i = 1, 2, ..., NoR \tag{4.2}$$

It can be chosen different for every reference. The parameter c_i represents $\Delta(\infty)$ and thus the desired final error bound. The parameters b_i and c_i added together results in the initial error bound. Parameter a_i characterizes the convergence speed to the final error bound.

4.2.3 Initial parameters

There are eight initial parameters for the chosen controller structure per controller. Initially they are zero by default, but the inverse of the parameters corresponding to u_1, r_1 and r_2 must exist for (3.4) and (3.13). Therefore, the initial parameters will be:

$$\theta_{init} = \begin{bmatrix} -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

which resembles $u_1 = r_1 + r_2$.

These parameters also define the initial center of the ellipsoid ($\theta_c(0)$). The initial Σ which defines the shape of the ellipsoid is an 8x8 diagonal matrix with 10^4 as diagonal elements.

4.2.4 Sequential update

As said in the previous chapter, a controller exists for every input. It is not desirable to evaluate all parametersets at the same time. An arbitrary controller (controller A) uses all system outputs as input. When another controller (controller B) changes its control parameters, controller A will see an output change while its own parameters are unchanged. It seems for controller A that the system itself changed. When controller A has evaluated, that evaluation is not of importance at the next time step, because controller B already has changed the system that controller A felt. The chances of the unfalsified region lying outside of the ellipsoid increase. The controller parametersets have to be evaluated sequentially, giving the parametersets time to stabilize and reduce errors.

In this simulation, no connection is yet implemented between the parametersets. This means that if controller A is evaluated to a certain performance bound (while controller B is held constant), the system might be changed in such a way that for controller B the parameterset can not be updated because the performance requirement can not be met (i.e. there is no intersection between the ellipsoid and the unfalsified region).

4.3 Results

4.3.1 Fixed controller

To get a first feel for the idea of EUC and to be able to draw some early conclusions, first a situation is considered where for the first input a EUC MIMO controller is implemented and for the second input a fixed controller (see Fig. 4.1.

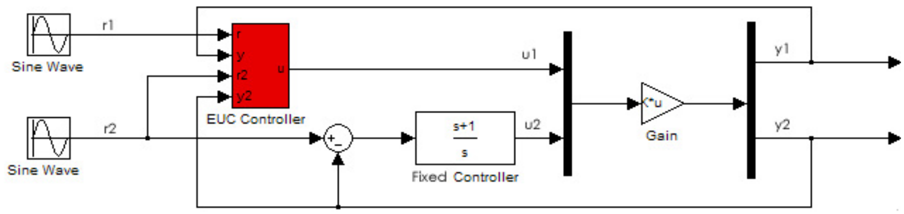

Figure 4.1: *The simulink model of section 4.3.1*

This setup is inadequate because the second controller can become instable for certain parametersets in the EUC MIMO controller. Also, a problem arises when input u_1 does not influence a certain output at all. In that case, when the parameterset fails to meet the performance requirement for that output, a new set is taken from the unfalsified set. This one will however also not be sufficient, because the controller can not control that output at all. Finally, no new unfalsified set can be found, and the algorithm is terminated. However a successful simulation can be run (here successful means that the performance requirement is met when $\Delta(t_k)$ is close to $\Delta(\infty)$). This is achieved partly by choosing the plant of the following form

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

Input u_1 influences y_1 and y_2 equally, while u_2 only influences the y_2. This is convenient for the EUC controller, because it is important to see how and if the algorithm successfully updates the parameterset for both references. The plant is chosen as a linear constant matrix, so the results can be analyzed with algebra.

The two references are chosen to be sinusoids. A sinusoid is a commonly used input signal, because its derivatives exist and are of similar form. A sinusoid with an amplitude of zero resembles a constant.

Table 4.1: Simulation settings

ref. 1	ref. 2	$\Delta(t_k)_1$	$\Delta(t_k)_2$	Fixed controller
0	$sin(1.3\pi t)$	$0.1 + e^{-0.02t}$	$0.1 + e^{-0.03t}$	$(s+1)/s$

Figure 4.2 shows the tracking errors e_1 and e_2 of both outputs. The tracking error stays within the error bound. It can be seen that while the parameterset is being updated for y_1, e_2 increases. This is not necessarily the case, but for these settings, it is expected. With the given initial parameters, y_1 (which is u_1 because of the structure of the plant (see (4.3)) will in fact initially resemble r_2, because $u_1 = r_1 + r_2$ and r_1 is zero. Output y_2 is also controlled by the error based fixed controller, so this controller will output zero, because the EUC controller is giving the perfect signal (reference 2). When y_1 hits the error bound, u_1 is decreased, which instantly results in an increased error in y_2, because u_1 does not equal reference 2 anymore.

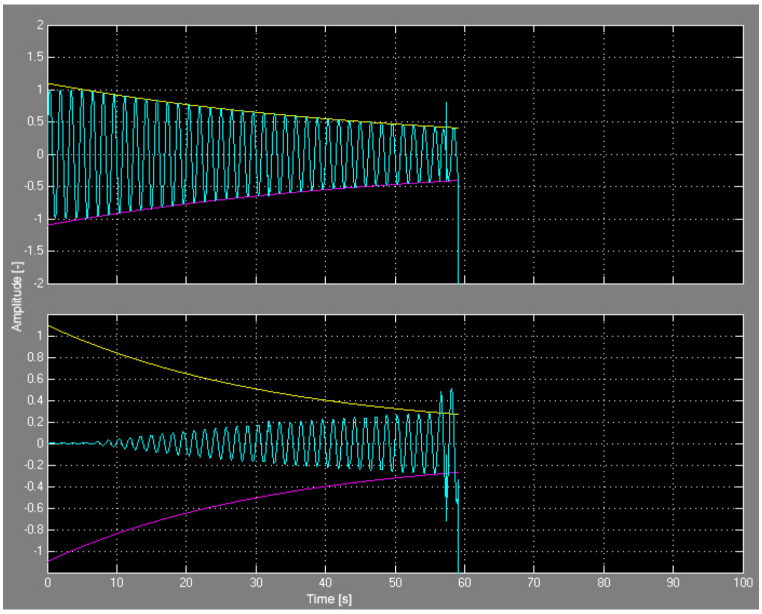

Figure 4.2: *Tracking error with error bound (bounding lines) for both outputs* y_1, y_2

It is also possible that the tracking error of the second output decreases while the parameterset is adjusted for the first output, for instance when the two inputs are the same signals (given the plant of 4.3).

It can also be seen that the simulation is stopped prematurely. This can have two reasons. Firstly, the region of unfalsified parametervalues lies completely outside of the ellipsoid and no intersection can be defined. In that case, no parameter sets can be found within the corresponding performance requirement with the current controller structure. This is a criterium for the simulation to stop. Secondly, MATLAB can have trouble solving the algebraic loop containing the fixed controller. Apparently a singularity occurs here for certain parametersets.

The first reason applies for this simulation. The contradictory demands of input 1 being the first reference for output 1 and being the second reference for output 2 make it impossible to find satisfactory parameters. Increasing the gain of the fixed controller (which is now very weak) will obviously help a lot, because then the error that input one makes on controlling output 2 is compensated better. Running a simulation with a higher gain on the fixed controller enables the simulation to finish. The end parameters are given by

$$\theta_{end} = \left[\begin{array}{cccccccc} -1.2077 & 0.1849 & 1.0000 & 0.1029 & -0.2077 & 0.1849 & -0.3803 & 0.4391 \end{array} \right]$$

which also shows that the third parameter (which is multiplied with the first reference) is in fact unchanged, because the first reference is zero.

From the previous simulation it can not be concluded if the EUC controller also successfully updates for the second output, because when the error bound hits e_2

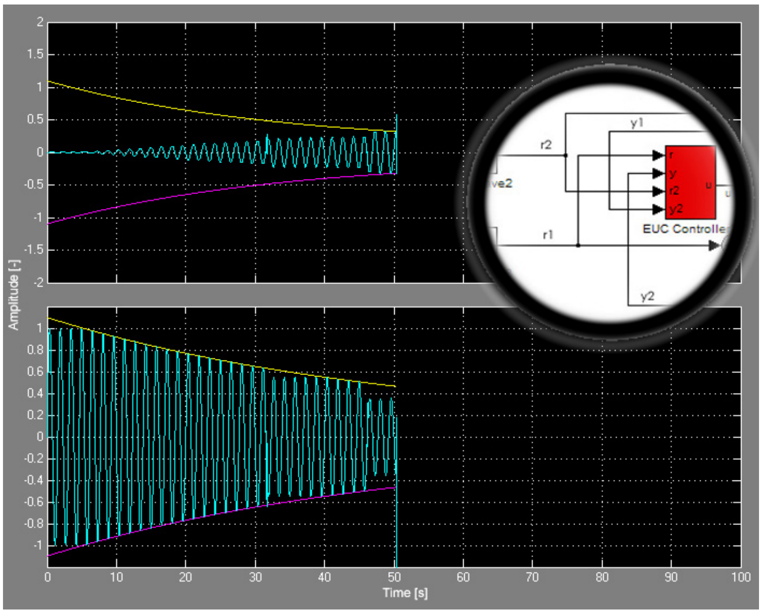

Figure 4.3: *Results when switching the inputs of the EUC controller*

the simulation is terminated. Therefore a new simulation is run with the same settings, but the connection of the two references and the two outputs to the EUC controller is now switched, which is emphasized in Fig. 4.3. The reference that is used for the first evaluation of the parameterset is now the reference 2. The second evaluation is now with regards to the first reference, which still hits the error bound first. It can now be seen that the second evaluation is executed correctly, resulting in a graph that is a mirrored version of Fig. 4.2. It can be concluded that the order in which the parameterset is evaluated is not of a noticeable influence.

4.3.2 MIMO Sequential EUC controller

The simulink model is extended with an EUC controller on the second input. The two controllers with two parametersets are evaluated in turns, for the reasons given in section 4.2.4. Initially, the controllers have the same given structure from (4.1) and the same initial parameters and are evaluated by the same algorithm. The amount of time that a certain parameterset is being updated before the model switches to the next controller is called the switching period. Every time the model switches, the ellipsoid of the parameterset that is about to be evaluated is given its initial dimensions back. The dimensions that the ellipsoid had at the last evaluation are not of importance for the new evaluation, because the system has changed during the evaluation of the other parameterset, as explained in section 4.2.4. This way, the chance of the region of unfalsified parameters to partly lie within the ellipsoid, is increased.

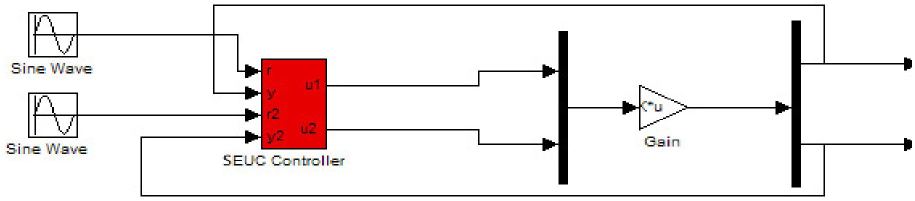

Figure 4.4: *The simulink model of section 4.3.2. The second input is also controlled by an EUC controller and the two controllers are merged in one box (named Sequential (S)EUC controller).*

Stop criterium

Given the new simulink model as in Fig. 4.4, the simulation of section 4.3.1 could be run successfully. Because the second controller is now also adaptable, the algorithm should be able to satisfy both error bounds. The stop criterium of the previous section (the simulation is stopped when no intersection between the ellipsoid and the unfalsified region can be defined), is altered. This stop criterium is now the switching criterium: when it is valid for one controller, the system switches to the next controller. The simulation is now stopped when successively for all controllers no intersection can be defined.

Switching period

An infinitesimal small switching period resembles a simultaneous parameterset update, a very large switching period (equal to the simulation time) resembles the single EUC controller version of section 4.3.1. A simulation is run similar to the simulation of section 4.3.1, as can be seen in Fig. 4.5. It can be seen that the simulation is stopped prematurely at 73,4 seconds, according to the stop criterium mentioned above. The switching period is chosen to be 0,01 seconds. A larger switching period causes the simulation to be stopped earlier. The longer a certain controller adapts its parameters, the more the system is changed for the other controller, and the greater the chance of not meeting the performance requirement. A smaller switching period is not of positive influence either, because then the algorithm doesn't have enough iterations to stabilize.

Figure 4.5 also shows a remarkable reversal of the direction in which the most parameters develop at approximately 50 seconds. At the same time, e_2 increases. The one parameters that doesn't show this behavior, is for both parametersets θ_4. This parameter, that is multiplied by r_2, decreases throughout the simulation. Before this point, the development of the parametersets is very similar. After the reversal, the two controllers are becoming increasingly different. The main difference is seen in the parameters which are multiplied by y_1 and y_2. The first controller becomes more dependent on y_2, while the second controller becomes more dependent on y_1.

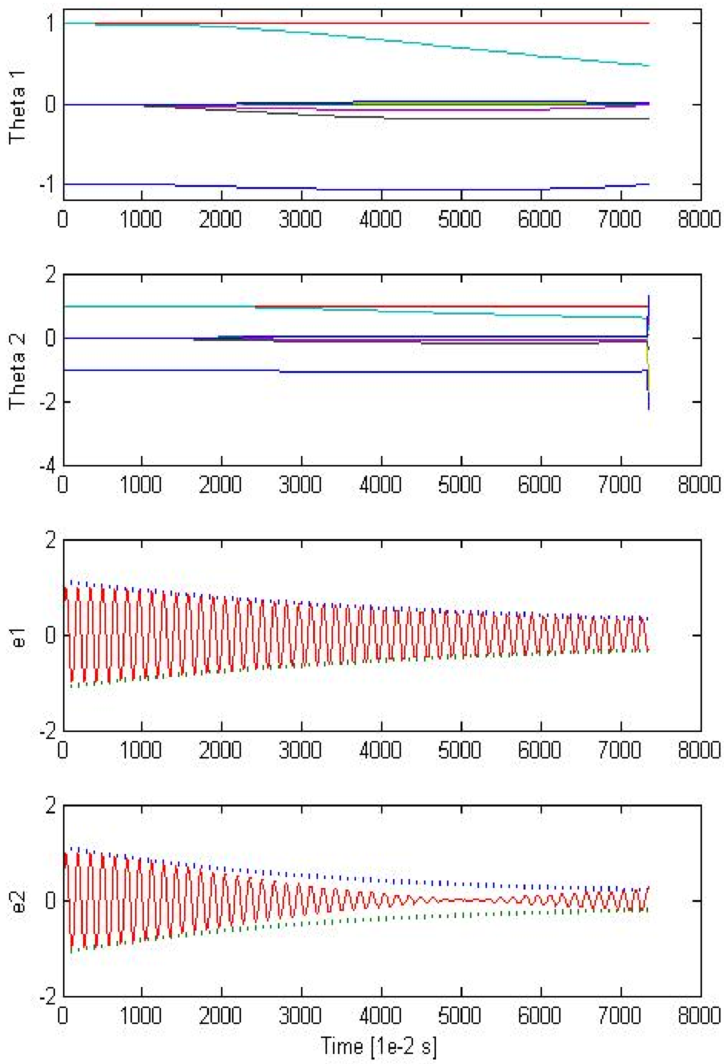

Figure 4.5: *Results of the simulation with the SEUC controller. The first two graphs show the development of the parametersets (Theta 1 ($\bar{\theta_1}$)and Theta 2 ($\bar{\theta_2}$)) and the other graphs show the tracking errors (e_1 and e_2) with error bounds (dashed lines).*

4.4 Possibilities for optimization

First of all, as already mentioned, if a certain input does not influence a certain output at all, the parameterset of the corresponding controller does not have to be evaluated for that output. Secondly, there exists a link between the parametersets belonging to different controllers, when they both influence the same output. If one of these controllers also controls another output, then the parameterset of that controller is also evaluated for that output. That evaluation could negatively influence the behavior of the output which the two controllers both control. By changing the corresponding parameters of the other controller, this influence can be compensated. Also the controller structure can be chosen freely and different from the choice made in chapter 4, giving the controller more dimensions of freedom to develop.

Chapter 5

Conclusion

A theory is developed for finding a controller for an unknown MIMO plant by using merely in- and outputdata. The control method is based on Ellipsoidal Unfalsified Control as developed by [1] and is called Sequential Ellipsoidal Unfalsified Control. It uses an adaptable controller for every input. The controllers use parametersets which are evaluated sequentially with respect to every reference. The controllers are adapted (by the sequential evaluations of one parameterset) in turn and are therefore also adapted sequentially. A drawback of this method is that there is no correlation implemented between controllers, while they have the same structure. A different approach of the problem would be to design one adaptable controller which makes a matrix-wise evaluation of a parameterset which controls every input at once, considering all references and all outputs as vectors.

The theory is implemented in a Simulink model to emphasize its use by simulations on a simple plant. The results imply a successful use of the theory on a real dynamic unknown plant with future optimizations and smart choices for the controller structure, the time-dependent maximum allowed tracking error and the switching period.

Bibliography

[1] J.J.M. van Helvoort, A.G. de Jager, M. Steinbuch, *Unfalsified control using an ellipsoidal unfalsified region applied to a motion system*, in 16th IFAC World Congress; Editors: IFAC, Prague, Czech Republic, 6 p.,(2005)