LABORATORY OF ROBUST IDENTIFICATION AND CONTROL

Lecture notes

Carlo Migliaccio

AY 2024/2025

Contents

Part I

Set-Membership System Identification

Chapter 1

Introduction

The *first step* of any control problem is tipically the derivatin of a **mathematical model** for the plant to be controlled that without loss of generality is a **mechatronic system**. This is the most critical step because, assuming that we use physics laws:

- We use *simplifying assumptions*;
- The value of the physical parameters involved in the equations (eg. mass, friction coefficients...) are not exactly known.

This fact is critical since *standard* approaches to controller design are **model based**, in the sense that the controller is designed by strongly relying on the mathematical model used to describe the mechatronic system under study. Clearly the neglection of some aspects, will result in a neglection of state variables! For example for certain problems the assumption of *rigid body* is satisfactory only under certain conditions. **How to solve this problem?** If we compare the two common approaches for designing a controller (state space or frequency description) the one based on frequency is much more able to face the problem of uncertainty.
In general, we can say that a controller is **robust** if it keeps good performances under the assumption of *uncertain description*. For this reason we need a **robust description** of the plant that, roughly speaking, is made up of a **nominal model** and by a model for the **uncertainty**[1]. Once such a model is derived we can apply some *robust control techniques* for designing a controller ($\mathcal{H}_\infty$, μ-synthesis...)

In order to deal with the presence of the uncertainty and to overcome the limitations related to the first principle modeling approach we will focus on **System Identification (SysId)** (Part I) and **Direct Data Driven Controller design (DDDC)** (Part II).

1.1 Mathematical modeling of dynamical systems

Since we have discussed about the importance of the mathematical model, now we can give an overview of the approaches one can track.

1.1.1 White-Box modeling *(first principles)*

The models deriving from **white-box approach** are obtained by applying the first principle of physics and all the physical phenomena involved in the equation, also all the *physical parameters*

[1] This can be modeled in an *unstructured* or *structured* way.

involved in the equation are **assumed to be exactly known**. The main idea which is useful to stress is that here **we know everything including the physical parameters**.

1.1.2 Gray-Box modeling

They are models based on equations obtained (again) by applying first principles, but this time the parameters entering the equations are not exactly known and so an estimation procedure from experimantally collected data is needed.

1.1.3 Black-Box modeling

In this case the structure of the equation is selected by the user on the basis of some "general" **a-priori information** on the system physics (eg. linearity). The parameters involved in the equation of the black-box model are then estimated/computed by using experimentally collected data. In general the parameter of a black box model do not have any *physical meaning.*

1.1.4 Some comments on the three approaches

White-box models are not very useful in practice, where is very irrealistic the fact of having the knowledge of everything! Instead, more interesting is the comparison between **Gray-box** and **Black-Box** models. In both cases we have *some information* (physical insights) and use the data in order to estimate the parameters themselves.

In **gray-box modeling** the structure of the equations is not selected by the user since it is forced by the first principle approach. For this reason in general the equations of a gray-box will depend in a **possibly complx nonlinear** way from the physical parameters to be estimated.

Example 1 (GRAY-BOX)

Let us assume that we know the plant to be modeled is an LTI one. You will remember that in the state-space representation, one can represent a dynamical system as:

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$y(t) = Cx(t) + Du(t)$$

If we look inside the A matrix for example we can find:

$$A = \begin{bmatrix} \frac{m}{k^2} & \sqrt{\beta} \\ \alpha^2 \frac{k^3}{\gamma} & \gamma^2 \end{bmatrix}$$

The mathematical procedure for the estimation of the parameters will be complex since they appear in the equation in a non linear way. Then, the modeling of such a plant will become very hard. This represents the main limitation of the gray-box approach.

On the other hand, in the **black-box approach**, we have more freedom to select the structure of the equations, especially because we do it in a *more convenient way* by only exploiting some general properties derived from our physical insights.

Coming back to the example we have just seen, the matrix A will be made up of four coefficients a_{ij} which appear linearly in the equation. The main difference is that such coefficients do not have a physical meaning.

1.2 Steps for mathematical modeling

The procedure we have for obtaining the mathematical model is the following:

1. **STEP 1** Exploit available *a-priori information* on the system under study to select the structure of the mathematical equations describing the Input-Output mapping. In the most general case we do not know all the state variables, from this fact we can understand that we derive an **input-output model** (equation) for the plant like the following:

$$\underbrace{y(t)}_{\text{output}} = f(\underbrace{u(t)}_{\text{input}}, \underbrace{\theta}_{\text{parameters}}) \tag{1.1}$$

2. **STEP 2** Collect Input-Output data representing the behaviour of the system under study by performing an (open-loop) experiment. In particular, we collect the output $\tilde{y}$ from the plant using as input the sequence $\tilde{u}$.

3. **STEP 3** To formulate a suitable **mathematical problem** to estimate/compute the values of parameter $\theta = [\theta_1 ... \theta_n]^T$ in such a way that our mathematical model is going to describe the behaviour of the real system, **as well as possible (in some sense)**. A common approach is to compute the parameter by solving the following problem:

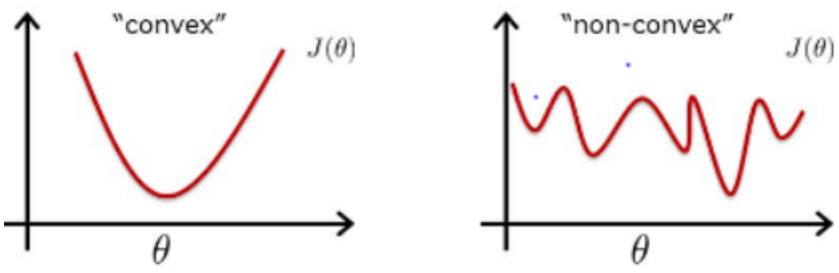
$$\hat{\theta} = \arg\min_{\theta} J(\theta) \tag{1.2}$$

where $\hat{\theta}$ is the vector of parameters to be estimated, while $J(\theta)$ is the functional to be minimized. Usually we take it as $J(\theta) = \|\tilde{y} - f(\tilde{u}, \theta)\|$. At this stage the difference between gray-box and black-box approach models comes into play, since:

- **Gray-box models** $\Longrightarrow$ $f(\tilde{u}, \theta)$ will depends by a complex nonlinear function from θ (parameters);
- **Black-box models** $\Longrightarrow$ $f(\tilde{u}, \theta)$ will be selected by the user in order to depend linearly from θ (if possible) or anyway in the *simplest possible way*.

1.3 Gray-Box vs Black-Box

We have said that in the case of **Gray-box models** in general $f(u, \theta)$ may be a *nonlinear* and *non convex* function of θ. This imply that the problem (1.2) is going to be a **non convex optimization problem**, and in this case it is not trivial to solve it, the best I can say is to find **local minima**. In some situations we could be particularly lucky, and choosing a particular initial point there is the possibility of finding the global minimum. However there is no way to certify it! Clearly a local minima, could correspond to a bad estimate of the parameter θ.



For **Black-box** models, by selecting the parametrization of f such that it could be a **convex function of** θ, the problem 1.2 becomes a **convex optimization problem**. In 1D the functional to be minimized is something similar to the one showed in the figure above. Convex functions have a unique **global minima**,there is no chance to be trapped in a local minima as in the non convex case. Clearly, waht is missed in this kind of approach is the *physical meaning* of the estimated parameters. Let us give an example to better clarify this aspect:

Example (Second-order LTI system)

Suppose that from first principles of physics we derive the following transfer function:

$$H(s) = \frac{\frac{p_1^2}{p_2}s + \frac{p_3}{\sqrt{p_4}}}{s^2 + \frac{p_1 p_4}{p_3^2}s + 1}$$

where $p_1, ..., p_4$ are physical (meaningful) parameters. What does we miss by modeling the transfer function by using the following model derived for example after a *black-box procedure*?

$$H(s) = \frac{\theta_1 s + \theta_2}{s^2 + \theta_3 s + \theta_4}$$

The physical meaning is clearly missed, but in many situation the objective is not to be grasped to the physical meaning of the parameters taken singularly, but (a) to derive an I/O model for the plant; (b) to use such a model for designing a (model-based) controller.

In order to conclude this discussion we can say that:

- The **black-box approach** is the *best choice* when we want either to simulate the I/O behaviour of the system or to design a **feedback control system**; an important remark to do is that the **structure** of a such a model must be selected by exploiting the most important a-priori information on the system (eg. **linearity**, **time invariance**...)
- The **gray-box approach** is the *best choice* when we want to estimate the values for some **physical parameters**.

It is true that – in all the approaches can be used for **System Identification** – the experimental data plays a crucial role, but also the *a-priori information* are of paramount importance. Besides, given the experimentally collected data there is an infinite number of functions which can interpolate those data. But if we apply an arbitrary input and then we compare the estimated output with the true one, we can confirm that the derived function just overfits the provided data. Conclusion: together with *a-posteriori information* (collected data), we need also the *a-priori information*, otherwise a well SysId procedure cannot be performed.

Chapter 2

System Identification: ℓ_p-norm estimators

We have introduced in the first chapter the concept of *System Identification* and we caught the importance of experimentally collected data, clearly in this more or less complex procedure one has to take into account that since that – the data are collected by performing experiments on the real plant, they can be affected by **uncertainty/measurement noise**.
Moreover, even if the system to be identified is a **continuous-time one** the most natural model for SysId is the discrete-time one, since samples of continuos time signals are collected.

2.1 Regression form for describing dynamical systems

There are evidences that – in a quite general manner – any dynamical system can be represented by using the so-called **regression form**, which stabilizes a relation between the current output, the input samples and the samples of the previous output. It is defined as follows:

$$y(k) = f(y(k-1), y(k-2), ..., y(k-n), u(k-1), ..., u(k-m), \theta) \tag{2.1}$$

For any physical system $m \leq n$ where n is the system order that is the number of *state variables*.

2.2 Error-in-variables(EIV): General setting for SysId

The most general setting describing an experiment performed on a plant to be identified is the **Error-in-variables (EIV)**, here both output $y(k)$ and input $u(k)$ are affected by measurement noise $\eta(k)$ and $\xi(k)$ respectively. Then the collected data can be represented by:

$$\tilde{u}(k) = u(k) + \xi(k) \tag{2.2}$$
$$\tilde{y}(k) = y(k) + \eta(k) \tag{2.3}$$

In some situation the sequence input $u(k)$ can be assumed to be perfectly known so $\xi(k) = 0$, because we build it in order to stimulate the system.[1] However, the **EIV** is more general and encapsulate also the situation in which the system to be identified is a subsystem from a more complex plant, then, since both $u(t)$ and $y(t)$ must be measured, both input and output are corrupted by uncertainty/measurement noise.

The following is a figure that shows schematically the setting we have just described:

Once we have fixed the setting, the a-priori information to provide are about:

[1] Later, when the concept on noise will be better formalized, we will give to such an approach the name of **output error (OE)**.

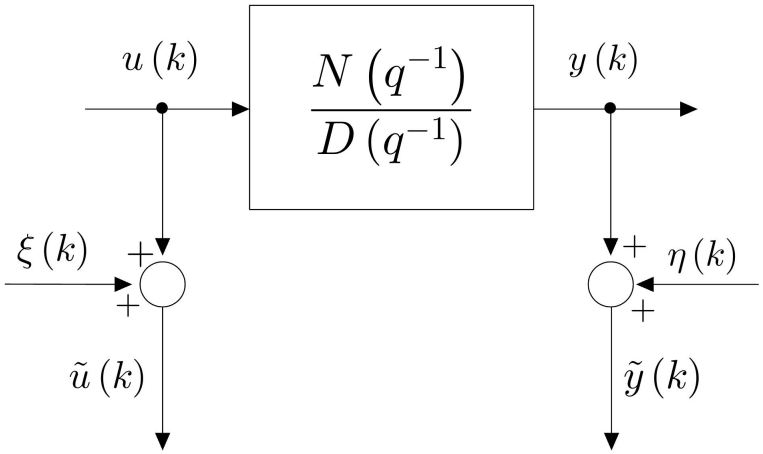


Figure 2.1: EIV SysId setting

- The **model** for example $f \in \mathcal{F}$ where $\mathcal{F}$ is associated with a certain class of systems (eg. LTI, nonlinear, stable...)
- The **noise** and in particular information reguarding the **statistical distribution** (white, gaussian...) or the **boundedness** (depending on the approach we are going to follow).

Now we are going to show an example which helps us to understand why the *regression form* is an effective model for describing in the most general manner a dynamical system.

Example (Regression form for a second order LTI system)

Let us consider a case in which we want to derive a model for a linear time-invariant system of the second order (a-priori assumption on the model: $f \in \mathcal{LTI}$, $n = 2$), furthermore suppose there is no noise in the data. The regression form $y(u(k), \theta)$ is:

$$y(k) = -\theta_1 y(k-1) - \theta_2 y(k-2) + \theta_3 u(k) + \theta_4 u(k-1) + \theta_5 u(k-2) \tag{2.4}$$

It is useful now to remind an important property (**backward-shift operator**) that tells us $s(k-r) = q^{-r}s(k)$, for this reason the 2.4 becomes:

$$\begin{aligned}
&y(k) = -\theta_1 q^{-1} y(k) - \theta_2 q^{-2} y(k) + \theta_3 u(k) + \theta_4 q^{-1} u(k) + \theta_5 q^{-2} u(k) \iff \\
&y(k)[1 + \theta_1 q^{-1} + \theta_2 q^{-2}] = u(k)[\theta_3 + \theta_4 q^{-1} + \theta_5 q^{-2}] \\
&\frac{y(k)}{u(k)} = \frac{\theta_3 + \theta_4 q^{-1} + \theta_5 q^{-2}}{1 + \theta_1 q^{-1} + \theta_2 q^{-2}} \iff H(z) = \frac{\theta_3 z^2 + \theta_4 z + \theta_5}{z^2 + \theta_1 z + \theta_2}
\end{aligned}$$

The last step comes up from the fact that can be dimostrated that it holds that $q^{-1} = z^{-1}$ and so from the regression form, passing through the backward-shift operator we can derive the transfer function of the system to be identified. Clearly a state-space description can be obtained once the parameters have been estimated by using the *realization theory* (transfer function $\rightarrow$ state space).

This example shows us in an inductive way that the regression form is the right one to use!

2.3 Least Squares estimation of the parameters θ_i

The objective of this paragraph is to show gradually – using significative examples – the problem of **parameter estimation** performed by using the **Least Squares (LS) model**, then we will analyze the pros and cons of such a method and some assumptions under which this kind of approach shows very nice properties (in a certain sense).

2.3.1 Estimation of parameters in the noise-free case

Let us consider again the case of a 2^{nd} order LTI system; we have seen it is characterized by the following regression form:

$$y(k) = -\theta_1 y(k-1) - \theta_2 y(k-2) + \theta_3 u(k) + \theta_4 u(k-1) + \theta_5 u(k-2)$$

The objective of the SysId procedure is to estimate the parameter $\theta = [\theta_1, \theta_2, \theta_3, \theta_4, \theta_5]$. We have to carry out an **open-loop experiment** on the plant by injecting the sequence $u(k)$ and collecting the output $y(k)$ for $k = 1, ..., H$. Since in the regression form we use samples till $k-2$ we have to start from $n+1 = 3$. Then:

$$\begin{aligned}
y(3) &= -\theta_1 y(2) - \theta_2 y(1) + \theta_3 u(3) + \theta_4 u(2) + \theta_5 u(1) \\
y(4) &= -\theta_1 y(3) - \theta_2 y(2) + \theta_3 u(4) + \theta_4 u(3) + \theta_5 u(2) \\
y(5) &= -\theta_1 y(4) - \theta_2 y(3) + \theta_3 u(5) + \theta_4 u(4) + \theta_5 u(3) \\
y(6) &= -\theta_1 y(5) - \theta_2 y(4) + \theta_3 u(6) + \theta_4 u(5) + \theta_5 u(4) \\
y(7) &= -\theta_1 y(6) - \theta_2 y(5) + \theta_3 u(7) + \theta_4 u(6) + \theta_5 u(5)
\end{aligned} \tag{2.5}$$

In this case $H = 3n+1 = 7$, it is quite evident we can express the equation 2.5 in matrix form as:

$$\underbrace{\begin{bmatrix} y(3) \\ y(4) \\ y(5) \\ y(6) \\ y(7) \end{bmatrix}}_{y} = \underbrace{\begin{bmatrix} -y(2) & -y(1) & u(3) & u(2) & u(1) \\ -y(3) & -y(2) & u(4) & u(3) & u(2) \\ -y(4) & -y(3) & u(5) & u(4) & u(3) \\ -y(5) & -y(4) & u(6) & u(5) & u(4) \\ -y(6) & -y(5) & u(7) & u(6) & u(5) \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \end{bmatrix}}_{\theta} \tag{2.6}$$

Then the five equations can be rewritten as $y = A\theta$, and in the case in which the matrix A is invertible ($\det(A) \neq 0$), the problem of estimating the θ parameters is simply:

$$\theta = A^{-1} y \tag{2.7}$$

The fact that must be $\det(A) \neq 0$ is not so hard to guarantee since the first two columns of the matrix A containing the samples of the output are likely to be very different! In order to prove it, it is sufficient to understand that for each LTI system there is a transient in which the output is not perfectly stabilized, even when the stimula assume very simple shapes (eg. step...). The matrix A is square by construction, then since we have $h = 2n+1 = 5$ parameters, we need 5 equations to obtain a unique solution to the problem. This approach is valid even with nonlinear functions which *depends linearly on the parameters*, we are sampling input and output, for this reason it is not important that the function f (of the regression form) is nonlinear. Important remarks:

- For a system of order n we need to estimate $h = 2n+1$ parameters;
- The minimal number of samples we need is $H = 3n+1$
- We can start applying the regression form function from the instant $k = n+1$, since it depends on both preceeding output and input.

2.3.2 Estimation of parameters in the noisy case

The fact that the collected samples were noisy free was only a simplificative assumption made up to introduce the problem. In real-world applications there is always uncertainty. Let us consider an example which will make necessary the collection of more and more data.
Let us consider a static system of the type[2]

$$y(k) = \theta u(k) \tag{2.8}$$

It seems that we can correctly estimate θ by just collecting a *simple pair* $(u(k), y(k))$ in order to obtain $\theta = \frac{y(1)}{u(1)}$. Now let us assume that the input data are exact, while the output sample $y(1)$ is corrupted by a noise $\eta(1)$. What is obtained is as follows:

$$\begin{cases} \tilde{u}(k) = u(k) \\ \tilde{y}(k) = y(k) + \eta(k) \end{cases}$$

Since $\tilde{y(k)} \neq y(k)$, the estimate given by $\frac{y(1)}{u(1)}$ is completely wrong, since:

$$\hat{\theta} = \frac{\tilde{y}(1)}{u(1)} = \frac{y(1) + \eta(1)}{u(1)} = \theta + \frac{\eta(1)}{u(1)} \neq \theta$$

What to do? The idea is to collect a number of data $H \gg 2n + 1$, in this case the matrix A becomes a tall matrix, there is not a unique solution as in the case of the noise-free example, but we can get an approximation $\hat{\theta}$ such that $\tilde{y} \approx y$. The following steps can be done:

$$\tilde{y} = A\theta \rightarrow A^T\tilde{y} = (A^TA)\theta \iff \theta = \underbrace{(A^TA)^{-1}A^T}_{A^*}\tilde{y} \qquad \text{(Normal Equations)}$$

where A^* is the Moore-Penrose pseudoinverse (generalization of the inverse for non-square matrices)[3]. It can be demonstrated[4] that the (Normal Equations) is the solution of the problem:

$$\theta_{LS} = \arg\min_{\theta} \|\tilde{y} - A\theta\|_2^2 \tag{LS}$$

that is the well-known **Least-Squares problem**, the deriving estimator is called the ℓ_2 estimator. This is a statistical approach to *parameter estimation* which has nice properties:

- The *computational burden* is very low! The only needed operation is the inversion of A^TA;
- There is a recursive way to solve it that reduces the work load in presence of big matrices (online computation).[5]
- The most important and 'powerful' property is the **consistency property**, which holds when two assumptions are satisfied. The next paragraph deals with the explanation of such assumptions.

[2]This could be for example a model for a resistor in which flows a certain current $(u(k))$ and we want to measure the voltage $(y(k))$.

[3]Keep in mind it is derived from the Singular Value Decomposition (SVD), which the generalization of the spectral Decomposition for non-symmetric matrices.

[4]The problem (LS) appears to be a convex quadratic unconstrained minimization problem. If the functional is explicitly written as a quadratic function, then after computing the gradient, its root raises the normal equations.

[5]See for more details: `https://en.wikipedia.org/wiki/Recursive_least_squares_filter`

2.3.3 ℓ_2-norm estimation (Least Squares): consistency property

Theorem 1 (Consistency Theorem). *If the following two assumptions are satisfied:*

1. *The noise can be considered as an additive term entering the problem that is*

$$\begin{aligned} y(k) = &- \theta_1 y(k-1) - \theta_2 y(k-2) - ... - \theta_n y(k-n) \\ &+ \theta_{n+1} u(k) + \theta_{n+2} u(k-1) + ... + \theta_{n+m+1} u(k-m) + \underbrace{e(k)}_{\text{EQUATION ERROR}} \end{aligned}$$

2. *The samples $e(k)$, $k = 1, ..., H$ are indipendent and identically distributed (white) random variables which can be modeled through a* ***zero-mean Gaussian noise***

Then, it holds that[6]*:*

$$\lim_{H \to \infty} \mathbb{E}[\theta_{LS}] = \theta \tag{2.9}$$

In a simplified way such a theorem states that under the two assumptions (satisfied), if you enlarge H, $\theta_{LS} \to \theta$.

2.3.4 Analysis of the assumptions

At this point, we wonder if the just exposed result, solved all of our problem for the parameter estimation, and then if the LS approach can be used in general. This is nothing but verifying if the two hypotesis are satisfied. For the sake of clarity let us take the most general setting for an experiment on a plant to identify (EIV).
Grasping on the a-priori information that the system is an LTI second-order one, let us take the associated regression form

$$y(k) = -\theta_1 y(k-1) - \theta_2 y(k-2) + \theta_3 u(k) + \theta_4 u(k-1) + \theta_5 u(k-2)$$

Since $u(k) = \tilde{u}(k) - \eta(k)$ and $y(k) = \tilde{y}(k) - \xi(k)$, we can substitute them obtaining:

$$\begin{aligned} \tilde{y}(k) = &- \theta_1 \tilde{y}(k-1) - \theta_2 \tilde{y}(k-2) + \theta_3 \tilde{u}(k) + \theta_4 \tilde{u}(k-1) + \theta_5 \tilde{u}(k-2) + \\ &\underbrace{+\theta_1 \eta(k-1) + \theta_2 \eta(k-2) - \theta_3 \xi(k) - \theta_4 \xi(k-1) - \theta_5 \xi(k-2)}_{e(k)} \end{aligned} \tag{2.10}$$

It is evident, I can envelope all the terms associated with the noise samples in a term which I call $e(k)$. Then, **the first assumption is satisfied**. What about the second? We have to check if the sequence

$$e(k) = \theta_1 \eta(k-1) + \theta_2 \eta(k-2) - \theta_3 \xi(k) - \theta_4 \xi(k-1) - \theta_5 \xi(k-2) \tag{EE}$$

is a white one (samples iid). Let us analyze a pair of samples:

$$\begin{aligned} e(3) &= \theta_1 \eta(2) + \theta_2 \eta(1) - \theta_3 \xi(3) - \theta_4 \xi(2) - \theta_5 \xi(1) \\ e(4) &= \theta_1 \eta(3) + \theta_2 \eta(2) - \theta_3 \xi(4) - \theta_4 \xi(3) - \theta_5 \xi(2) \end{aligned}$$

How it is highlighted, only by taking two of the $e(k)$ we can note they depend from common samples, for this reason the sequence $e(k)$ itself it is not white at all! They will provide an estimate θ_{LS} which is not going to enjoy of the consistency property. Even if the setting was OE instead of EIV, the same conclusion would have been drawn.

[6] We take the expected value of the estimate since random variables are introduced in the problem by adding $e(k)$, the estimate itself becomes a random variable.

2.4 The Equation Error (EE) noise structure

We have concluded in the former paragraph that the LS estimate is not suitable in the case we have either an EIV or an OE setting. Thus, what is the case in which the LS can be used? (again: that is, the two assumptions are verified). It is necessary to better dissect the properties of (2.10). In particular, it is useful (passing through the backward-shift operator) finding what is the relation between $\tilde{y}(k)$, $u(k)$ and $e(k)$. In order to discover such properties let us assume that the setting used is the Output Error (without loss of generality). Using $s(k-r) = q^{-r}s(k)$ we can write:

$$\tilde{y}(k)[1 + \theta_1 q^{-1} + ... + \theta_n q^{-n}] = u(k)[\theta_{n+1} + \theta_{n+2} q^{-1} + ... + \theta_{n+m+1} q^{-m}] + e(k) \iff \quad (2.11)$$

$$\tilde{y}(k) = \frac{[\theta_{n+1} + \theta_{n+2} q^{-1} + ... + \theta_{n+m+1} q^{-m}]}{[1 + \theta_1 q^{-1} + ... + \theta_n q^{-n}]} u(k) + \frac{1}{[1 + \theta_1 q^{-1} + ... + \theta_n q^{-n}]} e(k) = \quad (2.12)$$

$$= \frac{N(q^{-1})}{D(q^{-1})} u(k) + \frac{1}{D(q^{-1})} e(k) = \frac{N(z)}{D(z)} u(k) + \frac{1}{D(z)} e(k) \quad (2.13)$$

The deriving setting is represented in the figure below:

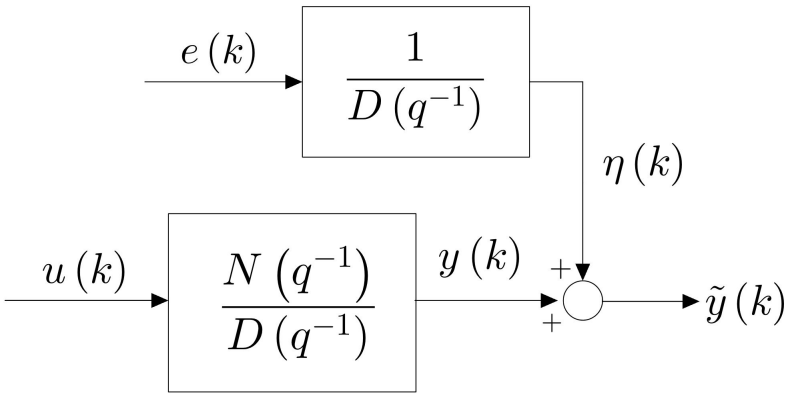


Figure 2.2: Equation Error (EE) noise structure

In this way we can conclude that the LS estimate makes sense if the date corrupted by a random sequence $e(k)$ (the noise) filtered by a system whose transfer function is the denominator of the system to be identified! No sense, since the sensor in general has nothing to share with the plant we want to identify. However there are some cases in which the LS approach can be used, we refer to the few cases in which the plant to be identified is such that $D(q^{-1}) = 1$. This occurs when I have to do:

- Identification of **FIR systems** (Finite impulse response);
- Identification of static systems;

When the denominator of the transfer function is equal to one, the Equation Error plays the role of the *output measurement error* $\eta(k)$. In this case also the **second assumption** is satisfied.

2.4.1 System Identification of Finite Impulse Response (FIR) systems

This of type of system has a transfer function which depends only on the samples of the input and not on on the previous output samples.

$$\tilde{y}(k) = \theta_1 u(k) + \theta_2 u(k-1) + ... + \theta_n u(k-n) + \underbrace{e(k)}_{\eta(k)} \tag{2.14}$$

Here also the second assumption is satisfied since the error samples are iid. In the case of FIR the property (2.9) is fulfilled.

2.4.2 System Identification of Static systems

Here the output is a function of parameters θ_i and of the input $u(k)$ at the current instant.

$$y(k) = \theta_1 g_1(u(k)) + \theta_2 g_2(u(k)) + ... + \theta_n g_n(u(k)) + e(k) \tag{2.15}$$

The functions $g_i(u(k))$ can be trigonometric functions, polynomial or anyway any other basic function.

Beyond the ℓ_2 (Least-Squares) estimator, there are other estimators that under the assumption for the noise of satisfying some statistichal properties, are consistent ones. The **consistency** deals with the property for the expected value of an estimator to converge to the real value θ of the parameter vector with the increasing number of data. In the following the ℓ_∞-norm and ℓ_1-norm estimators are presented with their features. Moreover a transformation of such problems into Linear Programs is sketched.

2.5 ℓ_∞-norm parameter estimation

The ℓ_∞-norm parameter estimation is issued by properly solving the following optimization problem:

$$\theta_{\ell_\infty} = \arg\min_{\theta \in \mathbb{R}^p} \|y - A\theta\|_\infty \tag{2.16}$$

It can be demonstrated that under the assumption that the uncertainty enters into the identification problem as an *equation error* and $e(k) \sim U([-a, a])$[7], the estimator in (2.16) is consistent. This is equivalent to say:

$$\lim_{N\to\infty} \mathbb{E}[\theta_{\ell_\infty}] = \theta$$

It is remarkable that using the **epigraphic formulation** in (2.16) the following problem can be recasted into a **Linear Program** (LP). In particular, keeping in mind that

$$\|y - A\theta\|_\infty \doteq \max_i |y_i - a_i^T \theta|$$

the problem can be rewritten, introducing a **scalar slack variable** t and pushing the objective in the constraints[8]:

$$\begin{aligned} &\min_{\theta \in \mathbb{R}^p, t \in \mathbb{R}} \; t \\ &\text{s.t. } |y_i - a_i^T \theta| \le t \quad \forall i \end{aligned} \tag{2.17}$$

[7] Uniform probability distribution

[8] Note that: $\max_i |y_i - a_i^T \theta| \le t \iff |y_i - a_i^T \theta| \le t \quad \forall i$

An *augmented optimization variable* $x = [\theta, t] \in \mathbb{R}^{p+1}$ can be used in order to transform the original problem into:

$$\begin{aligned} &\min_x c^T x \\ &\text{s.t. } \tilde{A}\theta \leq \tilde{b} \end{aligned} \tag{2.18}$$

where

$$\tilde{A} = \begin{bmatrix} -A & -\mathbf{1} \\ A & -\mathbf{1} \end{bmatrix}, \quad \mathbf{1} = [1\ 1\ ...\ 1]^T, \quad \tilde{b} = \begin{bmatrix} -y \\ y \end{bmatrix} \tag{2.19}$$

2.6 ℓ_1-norm parameter estimation

The ℓ_1-norm parameter estimation is performed by solving the following optimization problem:

$$\theta_{\ell_1} = \arg\min_{\theta \in \mathbb{R}^p} \|y - A\theta\|_1 = \arg\min_{\theta \in \mathbb{R}^p} \sum_{i=1}^{m} |y_i - a_i^T \theta| \tag{2.20}$$

In a similar way we have seen for the ℓ_2 and ℓ_1 estimators, can be demonstrated that under the assumption of the uncertainty entering the problem as an equation error and the noise samples $e(k) \sim \mathcal{L}(\mu, b)$[9], the ℓ_1-norm estimator is consistent, in the sense that

$$\lim_{N \to \infty} \mathbb{E}[\theta_{\ell_1}] = \theta \tag{2.21}$$

Also in this case the problem (2.20) can be recasted as an LP one, introducing some additional slack variables t_i for each one of the terms of the terms in the summation:

$$\begin{aligned} &\min_{\theta \in \mathbb{R}^p, t \in \mathbb{R}^m} \sum_{i=1}^{m} t_i \\ &\text{s.t. } |y_i - a_i^T \theta| \leq t_i \quad i = 1, ..., m \end{aligned} \tag{2.22}$$

Defining $x = [\theta, t] \in \mathbb{R}^{p+m}$ as the augmented optimization variable, the standard form of a *Linear Program* defined as in (2.18), by putting:

$$\tilde{A} = \begin{bmatrix} -A & -I \\ -A & I \end{bmatrix}, \ b = \begin{bmatrix} -y \\ y \end{bmatrix} \tag{2.23}$$

where I is the identity matrix $m \times m$.

2.7 Final remarks

Real experimets have data characterized by noise and in general the consistency property does not hold, in some situations in which the problem has a particular structure the assumptions required are perfectly fulfilled. We have understood that the most critical problem to manage is the second assumption which require the error to be white, zero mean and Gaussian (very strong assumptions!), moreover the resulting noise structure reaches a quite strange conclusion in which it is required that the system and the sensor share a part of their model.
In the following we will see another approach that, differently from LS, replaces the Assumption (2) with something that is significantly *less strong*.

[9] Laplacian probability distribution

Chapter 3

Set-Membership Identification: an introduction, Equation-Error noise structure

The objective of this chapter is to introduce an approach for the parameter estimation which requires to do less strong assumption on the noise affecting the experimentally collected data. After a brief introduction with the crucial ingredients, we will go on with some instructive examples which will bring us to the complete formulation of the **Set-Membership System Identification procedure**.

3.1 Ingredients for Set-Membership System Identification

As usual in order to perform correctly the procedure of System Identification we need some crucial ingredients:

❶ **A-priori assumption on the system:**

- ✓ We use the general **regression form**

$$y(k) = f(y(k-1), y(k-2), ..., y(k-n), u(k-1), ..., u(k-m), \theta) \tag{3.1}$$

- ✓ The **class of function** $\mathcal{F}$ and the order of the system n;

❸ **A-priori information of the noise** and in particular:

- ✓ **Noise structure**: is referred to the way the uncertainty enters into the problem.
- ✓ **Characteristic of the signal**, it is remarkable that here we assume something different and weaker. We will assume that the noise sequence/sequences (depending on the noise structure) belongs to a certain bounded set $\mathcal{B}$.

3.2 Set-Membership Identification of LTI system with EE noise structure

In this paragraph we will show what is obtained in term of parameter estimation, when we have that the a-priori information on the noise are the following:

✓ The uncertainty enter in the problem as an additive term which we call $e(k)$ (the same of the first assumption of the theorem), that is:

$$\begin{aligned} y(k) = &- \theta_1 y(k-1) - \theta_2 y(k-2) - ... - \theta_n y(k-n) \\ &+ \theta_{n+1} u(k) + \theta_{n+2} u(k-1) + ... + \theta_{n+m+1} u(k-m) + \underbrace{e(k)}_{\text{EQUATION ERROR}} \end{aligned}$$

✓ We suppose on the sequence characterizing the error is **bounded** (this is the crucial difference with respect to what requires the *consistency theorem*), that is:

$$e(k) \in \mathcal{B}_e \iff |e(k)| \leq \Delta_e, \ k = 1, ..., H \tag{3.2}$$

3.2.1 Feasible Parameter Set $\mathcal{D}_\theta$

In this paragraph by using some examples, we will define the *feasible parameter set* and its fundamental properties, in particular, it is useful to derive a **mathematical formulation of such a set** in order to explore its *usefulness* and *boundedness*

The set of solutions for the identification problem is implicitly described on what is called the **Feasible Parameter Set (FPS)**, we will indicate it with $\mathcal{D}_\theta$.

Definition 3.2.1 (FEASIBLE PARAMETER SET). *The **Feasible Parameter Set** $\mathcal{D}_\theta$ is the set of all the values of the parameter $\theta = [\theta_1 ... \theta_n]^T$ which are consistent (coherent) with all the available a-priori information (system and noise) and all the collected data (a-posteriori information).*

In order to better understand the meaning of $\mathcal{D}_\theta$ let us assume we are collecting data in the OE setup, and we know that $|\eta(k)| \leq \Delta_\eta$. The input sequence, then, is perfectly known while the output is corrupted by the noise $\eta(k)$.

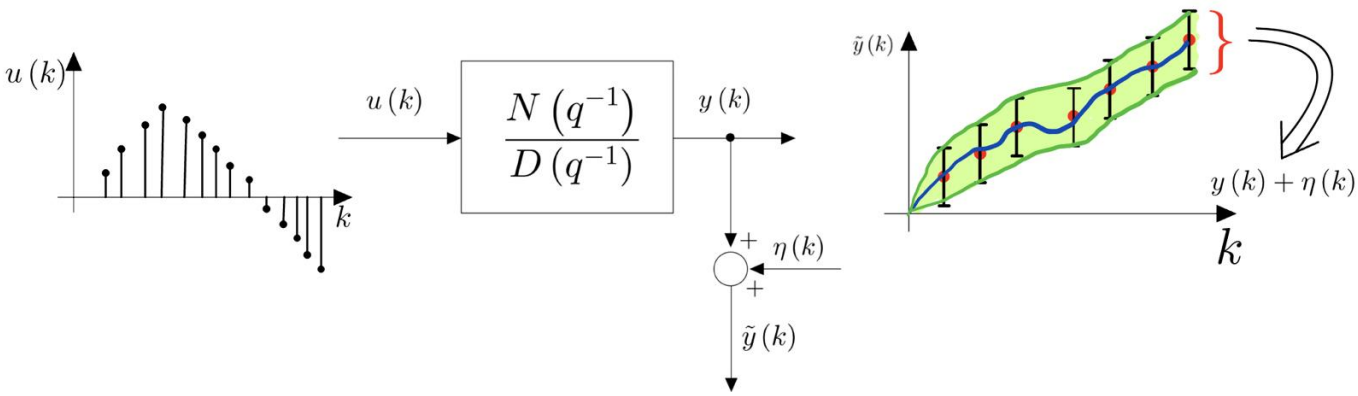


Figure 3.1: OE set-up experiment

The question is: Is the SM approach providing us a pointwise estimate for the parameter θ? We will show more or less formally in the following that the answer is NO, but we can track an intuitive reasoning which will bring us to the same conclusion.
For this aim, given the sequence $u(k)$, we collect an output $\tilde{y}(k)$, $k = 0, 1, ...$ Moreover, let us suppose that for such collected data the obtained model is giving us some parameter θ such that the I/O mapping is the one indicated with the blue line. Now, since the output measurements

are affected by noise, all of the samples between $[\tilde{y}(k) - \eta(k), \tilde{y}(k) + \eta(k)]$[1] is providing us an information which is coherent with the a-priori assumption on the noise itself. Besides, another couple of parameter θ_1, θ_2 derived from such samples, can be also the result of our identification problem. From the reasoning we have just presented we can intuitively understand that:

> probably the parameter θ_i are provided with an **uncertainty interval** since also the experimental data are provided with uncertainty.

3.2.2 Mathematical formulation of the FPS

We know that the system is a **second order**, **LTI one** (*a-priori assumption on the system*), the uncertainty/noise enters the identification problem as additive term $e(k)$ (equation error) and it is **bounded** (that is $|e(k)| \leq \Delta_e,\ k = 0, 1, ..., H$) (*a-priori assumption on the noise*), after having collected the data $\tilde{y}(k)$, after having stimulated the system to be identified with a sequence $\tilde{u}(k)$ (*a-posteriori information*)[2], we can define the Feasible Parameter Set as follows:

$$\begin{aligned}\mathcal{D}_\theta = &\{\theta \in \mathbb{R}^p :\ \tilde{y}(k) = -\theta_1\tilde{y}(k-1) - \theta_2\tilde{y}(k-2) + \\ &+ \theta_3\tilde{u}(k) + \theta_4\tilde{u}(k-1) + \theta_5\tilde{u}(k-2) + e(k), \quad k = 3, ..., H \\ &|e(k)| \leq \Delta_e, \quad k = 1, ..., H\}\end{aligned} \tag{3.3}$$

The set (3.3) is made up of both inequality and equality constraints, moreover it appears clear that it is a *subset of* $\mathbb{R}^p$ with p the number of parameters. There is a problem: the set $\mathcal{D}_\theta$ is defined using some inequality constraints on the noise samples $e(k)$ which are not part of the parameter space. In the following a way to eliminate such a dependence is shown, without adding any approximation or conservativeness.

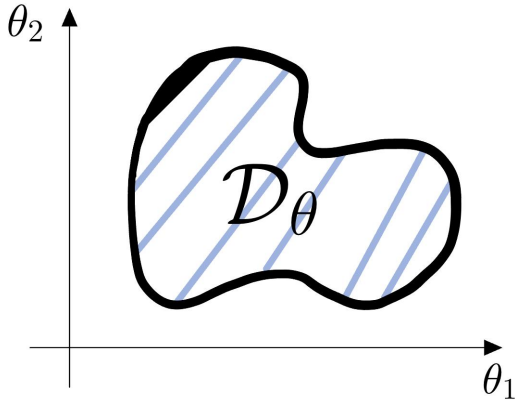
$$\begin{aligned}\mathcal{D}_\theta = &\{\theta \in \mathbb{R}^p :\ \tilde{y}(k) + \theta_1\tilde{y}(k-1) + \theta_2\tilde{y}(k-2) + \\ &- \theta_3\tilde{u}(k) - \theta_4\tilde{u}(k-1) - \theta_5\tilde{u}(k-2) = e(k), \quad k = 3, ..., H \\ &|e(k)| \leq \Delta_e, \quad k = 1, ..., H\} = \\ &\{\theta \in \mathbb{R}^p :\ |\tilde{y}(k) + \theta_1\tilde{y}(k-1) + \theta_2\tilde{y}(k-2) + \\ &- \theta_3\tilde{u}(k) - \theta_4\tilde{u}(k-1) - \theta_5\tilde{u}(k-2)| \leq \Delta_e, \quad k = 3, ..., H\}\end{aligned}$$

In this way, we have obtained an implicit description of the **set of all the feasible solution of our identification problem** in term of a *set of inequality contraints only involving* θ.

A graphical representation of such a set in a 2-dimensional parameter space is shown in the figure on the side. Here the objective is to analyze two main features of the Feasible Parameter Set by formulating the following two questions:

(Q1) **Boundedness** Is this set a bounded one? Under which conditions?

(Q2) **Usefulness** What is the relation between $\mathcal{D}_\theta$ and θ_{true}?



[1] See the black vertical bars

[2] An EIV(Errors-in-variables) set-up is assumed, then, in order to be more precise the input is given by a subsystem, this is the reason why we have to measure it.

Main features of $\mathcal{D}_\theta$

Question Q1 The boundedness of $\mathcal{D}_\theta$ depends on the way we collect the data (in general) ($\Rightarrow$ this concept has to be better specified in the next slides). *For the moment let us assume that such a set is bounded.*

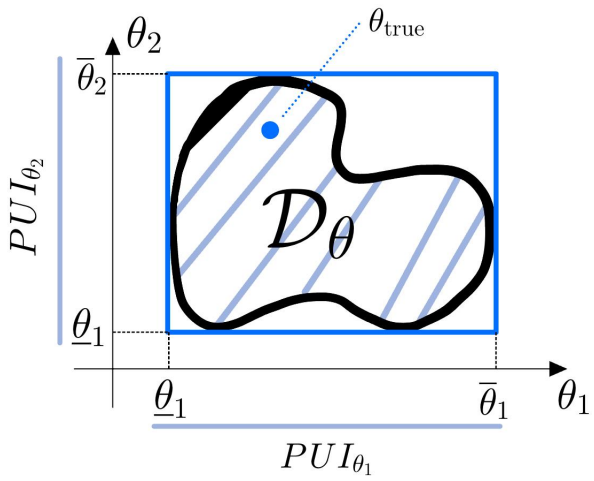
Question Q2 Assuming that the a-priori assumptions on the system and on the noise are correct, then θ_{true} is guaranteed to belong to $\mathcal{D}_\theta$ (this is very important for further theory development).

Once that we have obtained an implicit description of $\mathcal{D}_\theta$, *how can we extract a useful model from it,either for simulating the system or designing a feedback controller for such a system?* Before saying it, we have to distinguish *two different classes* of SM estimation algotrithms:

(E1) **Set-valued estimators**, defined as estimation algorithms which provides a (possibly) conservative estimate of $\mathcal{D}_\theta$ in a **simplified geometrical form** that can be easily used to simulate or control the system;

(E2) **Pointwise Estimators**, defined as estimation algorithms that provides a single value of θ which is an optimal estimate of θ_{true} in some sense.

Here, among all the possible estimator in the class (E1), we consider the algorithm which is providing the **minimum volume box outerbounding** $\mathcal{D}_\theta$. Such an estimator is implicitly providing what we will call **Parameter uncertainty Intervals (PUIs)**.



The PUI are defined as follows:

$$PUI_{\theta_j} = [\underline{\theta}_j, \overline{\theta}_j] \tag{3.4}$$

where the extrema of the interval are:

$$\underline{\theta}_j \doteq \min_{\theta \in \mathcal{D}_\theta} \theta_j \tag{3.5}$$

$$\overline{\theta}_j \doteq \max_{\theta \in \mathcal{D}_\theta} \theta_j = \min_{\theta \in \mathcal{D}_\theta} -\theta_j \tag{3.6}$$

It is remarkable that each PUI is providing the **minimum uncertainty interval** for each parameter θ_j. In the figure are shown the PUI for the feasible parameter set presented before.

Note the that, in the case of $\mathcal{D}_\theta \subseteq \mathbb{R}^2$ the minimum volume containing $\mathcal{D}_\theta$ is a rectangular shape whose sides are the parameter uncertainty intervals associated to θ_1 and θ_2.

Usefulness of the PUIs

Suppose you have for the following model, the corrensponding PUIs:

$$G(z) = \frac{\theta_2}{z + \theta_1} \quad \theta_2 \in [\underline{\theta}_2,\ \overline{\theta}_2] = PUI_{\theta_2}, \quad \theta_1 \in [\underline{\theta}_1,\ \overline{\theta}_1] = PUI_{\theta_1}$$

Moreover, you assume you want to derive a Lead/Lag controller for such a system. It is well known that the main idea behind this approach is having a representation of the loop function $L(s)$, and adding a certain number of dynamic networks in order to change the shape of $L(s)$ itself, in order to obtain a certain $\omega_{c,des}$ In this framework, we have not a single loop function

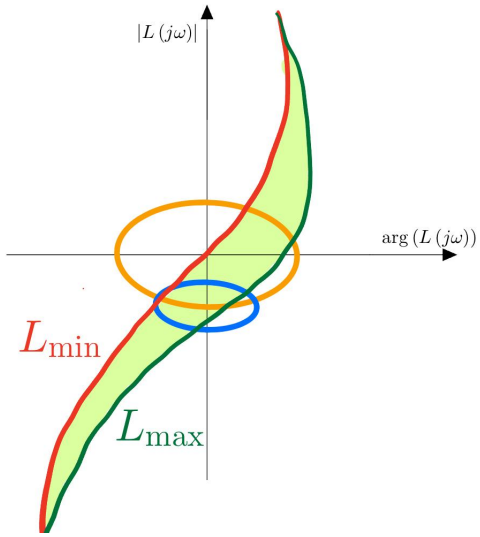


Figure 3.2: Nichols plot of the loop function

but a cloud of them, from which we can retrieve a bound of some type in order to derive a controller in the **limit cases**.

At this aim, the following figure for example shows the *frequency response* on the Nichols' Plot of the loop function. According to the choice of the parameters one can obtain a different plot in the range of curves delimited by $[L_{min}, L_{max}]$.

3.2.3 Geometric shape of the Feasible Parameter Set

In the case we have a linear time-invariant system where the uncertainty can be described by means of an unknown but bounded *equation error* $e(k)$, we can obtain a particular shape of the FPS which makes the problem particularly 'simple' to solve.
For sake of simplicity, the description of such a property by using a first order system ($n = 1$) is done. At this aim, we know that the transfer function has the shape:

$$G(z) = \frac{\theta_2 z}{z + \theta_1} \longrightarrow G(q^{-1}) = \frac{\theta_2}{1 + \theta_1 q^{-1}} = \frac{y(k)}{u(k)} \tag{3.7}$$

From which you can find that the regression form is:

$$y(k) + \theta_1 y(k-1) = \theta_2 u(k) \iff y(k) = -\theta_1 y(k-1) + \theta_2 u(k) \tag{3.8}$$

This result has been issued by using the **a-priori information on the system**: LTI and $n = 2$. On the other hand, we know that the noise enters the equation as an *unknown but bounded* equation error $e(k)$, that is

$$|e(k)| \leq \Delta_e \quad k = 1, ..., N$$

Finally, the **a-posteriori information** are the experimentally collected data, which both are corrupted by measurement noise.

$$\tilde{u}(k) = u(k) + \xi(k), \quad \tilde{y}(k) = y(k) + \eta(k) \quad \forall k = 1, ..., N$$

Putting all together, we can find the implicit mathematical formulation of the FPS as follows:

$$
\begin{aligned}
\mathcal{D}_\theta =& \{\theta \in \mathbb{R}^2 : \ \tilde{y}(k) = -\theta_1 \tilde{y}(k-1) + \theta_2 \tilde{u}(k) + e(k), \quad k = 2, ..., N, \\
& |e(k)| \leq \Delta_e \quad k = 1, ..., N\} = \\
& = \{\theta \in \mathbb{R}^2 : \ |\tilde{y}(k) + \theta_1 \tilde{y}(k-1) - \theta_2 \tilde{u}(k)| \leq \Delta_e, \quad k = 2, ..., N\} = \\
& = \{\theta \in \mathbb{R}^2 : \ -\Delta_e \leq \tilde{y}(k) + \theta_1 \tilde{y}(k-1) - \theta_2 \tilde{u}(k) \leq \Delta_e, \quad k = 2, ..., N\} = \\
& = \{\theta \in \mathbb{R}^2 : \ \theta_1 \tilde{y}(k) - \theta_2 \tilde{u}(k) \leq \Delta_e - \tilde{y}(k) \quad (\mathsf{C1}) \\
& \qquad\qquad -\theta_1 \tilde{y}(k) + \theta_2 \tilde{u}(k) \leq \Delta_e + \tilde{y}(k) \quad (\mathsf{C2}), \quad k = 2, ..., N\}
\end{aligned}
\tag{3.9}
$$

For each k, from $\mathcal{D}_\theta$ we obtain a couple of straight lines. Let us represent them by computing the constraints (C1) and (C2) for $k = 2, 3$

Constraints for $k = 2$

$$
\begin{aligned}
& \tilde{y}(2) + \theta_1 \tilde{y}(1) - \theta_2 \tilde{u}(2) \leq \Delta_e \\
& \theta_2 \geq \frac{\tilde{y}(1)}{\tilde{u}(2)} \theta_1 + \frac{\tilde{y}(2) - \Delta_e}{\tilde{u}(2)} \qquad (\mathsf{C1}) \\
& \tilde{y}(2) + \theta_1 \tilde{y}(1) + \theta_2 \tilde{u}(2) \leq -\Delta_e \\
& \theta_2 \leq \frac{\tilde{y}(1)}{\tilde{u}(2)} \theta_1 + \frac{\tilde{y}(2) + \Delta_e}{\tilde{u}(2)} \qquad (\mathsf{C2})
\end{aligned}
$$

We can note that such lines have the **same angular coefficient** but different **intercept**. Roughly speaking they are parallel straight lines, that in form of inequalities define some *half-planes*. Since in the FPS the constraints must be satisfied **for each** k, we take the intersection of such constraints. In the specific case they result in a **strip**. Such a set is unbounded, this is related to the fact we used a single constraints against two degree of freedom (that is 2 parameters).

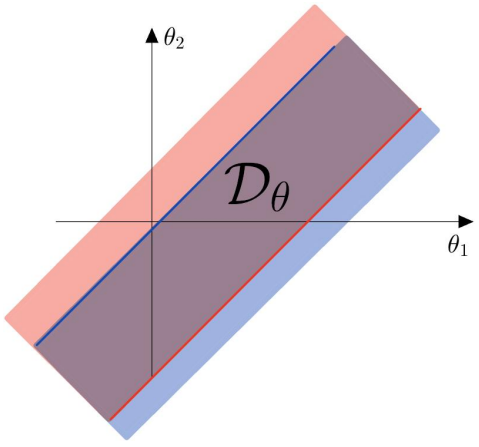


Figure 3.3: $\mathcal{D}_\theta$ resulting from a single equation

A possible representation is given in the figure above. Now, in order to obtain a bounded set we must collect $k = 3$ data while using at least 2 equations resulting in 2 pairs of constraints C1-C2.

Constraints for $k = 3$

Following the same reasoning we obtain:

$$\tilde{y}(3) + \theta_1 \tilde{y}(2) - \theta_2 \tilde{u}(3) \leq \Delta_e$$
$$\theta_2 \geq \frac{\tilde{y}(2)}{\tilde{u}(3)}\theta_1 + \frac{\tilde{y}(3) - \Delta_e}{\tilde{u}(3)} \quad \text{(C1)}$$
$$\tilde{y}(3) + \theta_1 \tilde{y}(2) + \theta_2 \tilde{u}(3) \leq -\Delta_e$$
$$\theta_2 \leq \frac{\tilde{y}(2)}{\tilde{u}(3)}\theta_1 + \frac{\tilde{y}(3) + \Delta_e}{\tilde{u}(3)} \quad \text{(C2)}$$

Combining the two halfplanes with the ones obtained before, we obtain that **intersection is a convex set**, and in particular is a **polytope**. Note that the angular coefficient for the couple of lines derived putting $k = 3$, clearly is different than the one we have obtained for $k = 2$.

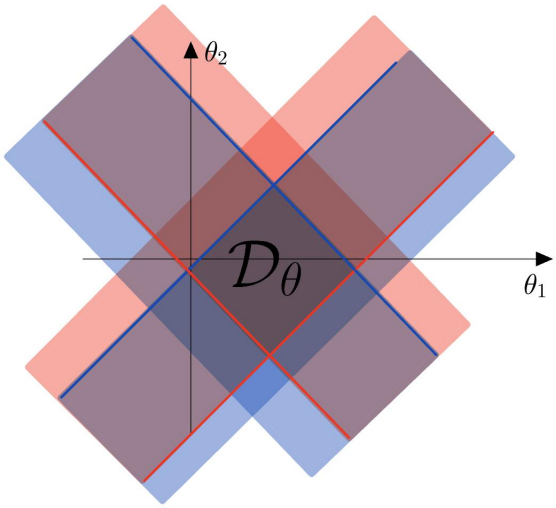


Figure 3.4: $\mathcal{D}_\theta$ resulting from a pair of equations

Let us remind that the FPS for sure contains θ_{true} if and only if all the a-priori assumptions are correct, otherwise the FPS will be **empty**!
It is remarkable, that this was a case in which by inspection you can read the PUI, however we need to generalize. In particular the problem of finding a PUI in the hypotesis of LTI systems with equation error noise structure can be recasted in the solution of **two standard Linear Programming (LP) problems**: minimization of a linear (convex) objective under constraints representing a polytope.[3]

[3] A **Linear Programming (LP) problem** is a convex optimization problem of the form

$$\min_{x \in \mathbb{R}^n} c^T x \quad \text{subject to } Ax \leq b$$

This takes into account all the situations when the problem can be *exactly written* as the minimization of a **linear function** of the optimization variable subject to **a set of linear inequalities and/or equalities constraints**.

3.3 PUIs computation by mean of LP problems solution

We know that for a certain parameter θ_j, $j = 1, ..., p$, the related interval is $PUI = [\underline{\theta}_j, \overline{\theta}_j]$. Under assumption of having a LTI system with Unknown But Bounded error.

$$\underline{\theta}_j \doteq \min_{\theta \in \mathcal{D}_\theta} \theta_j = \min_{\theta \in \mathbb{R}^n} \theta_j \qquad \text{s.t.} \begin{cases} \tilde{y}(k) + ... \leq \Delta_e \\ \tilde{y}(k) + ... \leq -\Delta_e \end{cases} \quad k = n+1, ..., N \tag{3.10}$$

$$\overline{\theta}_j \doteq \max_{\mathcal{D}_\theta} \theta_j = \max_{\theta \in \mathbb{R}^n} \theta_j = \min_{\theta \in \mathbb{R}^n} -\theta_j \quad \text{s.t.} \begin{cases} \tilde{y}(k) + ... \leq \Delta_e \\ \tilde{y}(k) + ... \leq -\Delta_e \end{cases} \quad k = n+1, ..., N \tag{3.11}$$

Note that both the problems (3.10) and (3.11) can be rewritten in the form of an LP problem by suitably choosing c and by arranging the set of constraints in suitable matrices A and b. For example when we want to compute the PUI for the θ_1 parameter for a first order system, the direction c will be $c = [1 \quad 0]^T$, then for θ_2, $c = [0 \quad 1]^T$. In `MATLAB` once you have formulated the problem in the LP form, you can use the command `[th, opt_val]=linprog(c,A,b)`, in which `th` is the optimal solution, while `opt_val` is the optimization variable.
In conclusion, we have understood that the problem of finding the PUIs for p parameters in the described setting, is leading to the solution of $2p$ LP problems, whose solutions are **global ones**, the problem in this case can be exactly solved without adding any conservativeness!

The most critical issue of using such an approach is that, despite any type of experimental set-up can be recasted into a model having an equation error noise structure, I rarely am able to retrieve a bound Δ_e.
More in particular, this approach can be applied all the times you have that the model is linearly parametrized, that is *the parameters appears linearly in the equation*, since from the regression form you have samples from the input and output which can be for sure non linear ones!

3.4 Final remarks

In this section we have introduced the *Set-Membership approach* for the model parameters estimation. Next, we have analysed the main ingredients, and introducing the concept of **Feasible parameter set** we have understood that such an approach leads with itself a robust way for describing the parameters, embedding the uncertainty which comes from the data collection. This has been carried out by properly defining the *Parameter Uncertainty Interval*.

In the last part we have applied all of the features of the described approach starting from the simplest case in which the model to be identified was LTI and the noise samples unknonwn but bounded. By suitably arranging the mathematical constraints derived by putting together all the a-priori and a-posteriori information, we have recasted the problem of finding the PUIs in the solution of a couple of LP problems which leads to a global minima. Finally, analyzing the problem of SM Identification in this way was mainly of theoretical and conceptual interest since we have seen that retrieving a bound Δ_e on the error is practically impossible. Other approaches embedding a complete description of the noise samples are needed.

References

✍ Milanese et al., Bounding approaches to system identification, 2013

✍ Milanese and Belforte, *"Estimation theory and uncertainty intervals evaluation in presence of unknown but bounded errors: Linear families of models and estimators"*, 1982

Chapter 4

Set-Membership SysId of LTI systems with Errors-In-Variables (EIV) noise structure

We have seen in the last chapter that an equation-error noise structure leads to a solution of LP problems for obtaining the Parameter Uncertainty Intervals, however, we have seen that there are some drawbacks. First of all the way we can find a bound Δ_e on the noise samples. Then, the objective here is to find a way for dealing with the 'original' problem, that is the one using the output and input samples corrupted by noise. In order to gradually present all the needed ingredients for properly solving the problem, we show in turn general teoretical results and examples in which such results are applied to our problem of *System Identification*.

4.1 Feasible Parameter Set in the EIV set-up

In order to define also for this type of set-up the feasible parameter set, we have to follow the same road as we have done before. In particular we have to put together:

A-priori information on the system We know that the system belongs to a certain class $\mathcal{F}$, moreover we know the order n of the system itself.

A-priori information on the noise In particular the way the uncertainty enters the identification problem (we assume here the most general case when both input and output are corrupted by noise) and the boundedness of the noise samples, in particular

$$|\eta(k)| \leq \Delta_\eta \quad |\xi(k)| \leq \Delta_\xi \quad \forall k = 1, ..., N \tag{4.1}$$

A-posteriori information They are nothing but the experimentally collected data

$$\tilde{y}(k) = y(k) + \eta(k), \quad \tilde{u}(k) = u(k) + \xi(k)$$

For an **LTI system of order** n the feasible parameter set is defined as follows:

$$\begin{aligned} \mathcal{D}_\theta = \{\theta \in \mathbb{R}^p : (\tilde{y}(k) - \eta(k)) + \sum_{i=1}^{n} \theta_i \ (y(k-i) - \eta(k-i)) = \\ \sum_{j=0}^{m} \theta_j \ (u(k-j) - \xi(k-j)), \quad k = n+1, ..., N\} \end{aligned} \tag{4.2}$$

In this context the definition of PUI is always the same, and the extrema of such an interval are defined as before by solving the optimization problems:

$$\underline{\theta}_j = \min_{\theta \in \mathcal{D}_\theta} \theta_j, \quad \overline{\theta}_j = \max_{\theta \in \mathcal{D}_\theta} \theta_j \Longrightarrow PUI_{\theta_j} = [\underline{\theta}_j, \overline{\theta}_j]$$

At this point the question is: **what type of $\mathcal{D}_\theta$ I obtain?**

4.2 Extended Feasible Parameter Set $\mathcal{D}_{\theta,\eta,\xi}$

How we are able to see in the (FPS) the set definition depends also on the noise samples. The difference here is that I cannot eliminate them without adding any approximation. Substancially, I am introducing a non trivial number of new unknown in the description of the set: for N collected samples, $2N$ new variables are added, which cannot be eliminated. For this reason we have to *enlarge* the set $\mathcal{D}_\theta$ involving also the new variables. In this way we introduce the so-called **Extended Feasible Parameter Set (EFPS)** which we indicate with $\mathcal{D}_{\theta,\eta,\xi}$.
In order to better understand this aspect, let us consider a toy-example in which we have a single parameter $\eta(1)$ which is added to the pair θ_1, θ_2. The EFPS in this case – how the figure below shows – is a subset of $\mathbb{R}^3$.

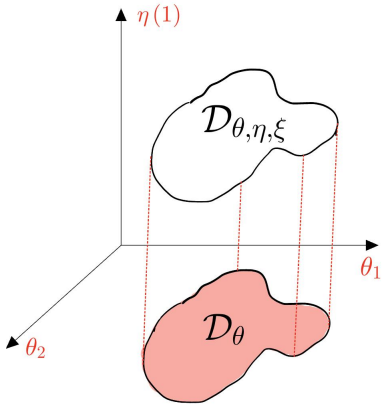


Figure 4.1: Example in $\mathbb{R}^3$ of the Extended feasible parameter set

The FPS $\mathcal{D}_\theta$ is nothing but the projection on the (θ_1, θ_2) plane of the set $\mathcal{D}_{\theta,\eta}$, in this case no parameter ξ is present.
The definition of the **Extended Feasible Parameter Set** becomes the following:

$$\begin{aligned}\mathcal{D}_{\theta,\eta,\xi} = \{&\theta \in \mathbb{R}^p, \eta \in \mathbb{R}^N, \xi \in \mathbb{R}^N : \tilde{y}(k) - \eta(k) + \theta_1(y(k-1) - \eta(k-1)) + \\ &+ \theta_2(y(k-2) - \eta(k-2)) + \cdots + \theta_n(y(k-n) - \eta(k-n)) = \\ &= \theta_{n+1}(u(k) - \xi(k)) + \theta_{n+2}(u(k-1) - \xi(k-1)) + ... + \\ &\theta_{n+m+1}(u(k-m) - \xi(k-m)), \quad k = n+1, ..., N \\ &|\xi(k)| \leq \Delta_\xi, \quad |\eta(k)| \leq \Delta_\eta, \quad k = 1, ..., N\}\end{aligned} \tag{4.3}$$

Such a set is defined by **nonlinear** and **non-convex** constraints and then the set is non-convex. In the specific case, the constraints that arises are **bilinear ones** which are a particular class of **polynomial constraints**. In general we know that is very hard to obtain a global minimum from a non-convex optimization problem, in this case using some tools for *polynomial optimization* it is possible to reach a global minimum.

In this framework the problem of finding the PUIs becomes:

$$PUI_{\theta_j} = [\underline{\theta}_j, \overline{\theta}_j] \Longrightarrow \underline{\theta}_j = \min_{\theta,\eta,\xi \in \mathcal{D}_{\theta,\eta,\xi}} \theta_j, \quad \overline{\theta}_j = \max_{\theta,\eta,\xi \in \mathcal{D}_{\theta,\eta,\xi}} \theta_j \tag{4.4}$$

4.3 Convex relaxation for Polynomial Optimization Problems (POPs)

In this section we will introduce some general theoretical results on polynomial optimization problems, that – how we have seen – are the ones arising in the definition of the extended feasible parameter set (EFPS). We will start by formulating them, we will analyze the type of sets they produce and finally we will introduce the concept of **convex relaxation**.

Let us consider the following general optimization problem:

$$\begin{aligned} &\min_x f_0(x) \\ &\quad \text{s.t. } f_k(x) \leq 0 \quad k = 1, ..., l \\ &\quad f_k(x) = 0 \quad k = l+1, ..., m \end{aligned} \tag{4.5}$$

where f_0 and $f_k,\ k = 1, ..., m$ are **multivariate polynomials** in the optimization variable x. Giving an example if $x = [x_1\ x_2\ x_3]^T$, an example of f_0 is

$$f_0(x) = x_1^2 + x_2 x_3^3 + x_1^5 x_2^3 + 7x_3^2$$

All POPs can be written, for sure, in the so-called **epigraphic form** by introducing a scalar (slack) variable γ.

Original formulation

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} f_0(x) \\ &\quad \text{s.t. } f_k(x) \leq 0 \quad k = 1, ..., l \\ &\quad f_k(x) = 0 \quad k = l+1, ..., m \end{aligned} \tag{4.6}$$

Epigraphic formulation

$$\begin{aligned} &\min_{x \in \mathbb{R}^n, \gamma \in \mathbb{R}} \gamma \\ &\quad \text{s.t. } f_0(x) \leq \gamma \\ &\quad f_k(x) \leq 0 \quad k = 1, ..., l \\ &\quad f_k(x) = 0 \quad k = l+1, ..., m \end{aligned} \tag{4.7}$$

By rewriting a generic POP in the epigraphic form, it becomes a problem of **minimizing a linear function over a non-convex set described by polynomial constraints** (this is also true for the unconstrained case).

The figure (4.3) shows the general idea. We are moving a linear objective (thin lines in black) over the set in sky-blue derived by the polynomial (so non-convex) constraints. The thin lines are the so-called *level-curves* in order to minimize θ_1. Moreover the orange point is a **local minimum**, while the red one is the **global minimum**.

Important remark: since a generic POP can written in epigrapigh form, we can note that the non-convexity of the problem is now **completely embedded** in the description of the set of constraints → whether we want to compute the **global optimal solution** we have to deal with the non-convexity of the set of constraints. It is very high the probability of getting stuck in local minima.

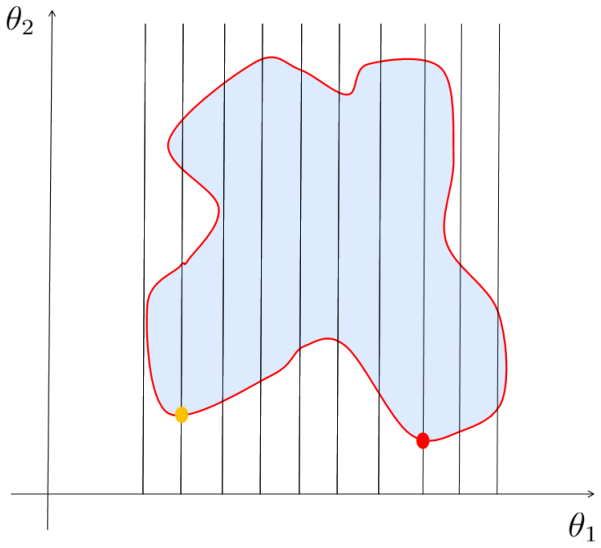


Figure 4.2: Example of min a linear objective (thin line) over a non-convex set (sky-blue)

Definition 4.3.1 (Convex hull of a non-convex set)**.** Given a non-convex set $\mathcal{S}$, the **convex hull** for it is the smallest convex-set including $\mathcal{S}$.

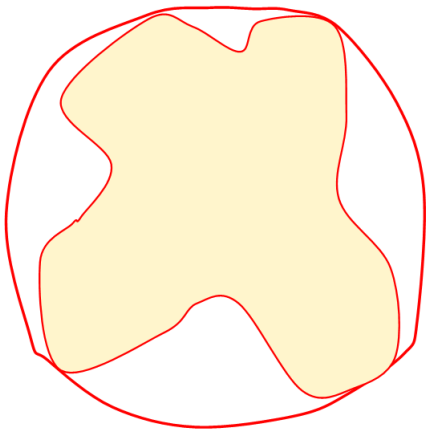
A very-nice property is that the **min/max** computed on the convex hull is equal to the one computed on the original set. In this way we gain the convexity of the problem.

In other words,if we are able to write down the equations describing the convex hull $C_{\mathcal{S}}$ of the non-convex set $\mathcal{S}$ we have that:

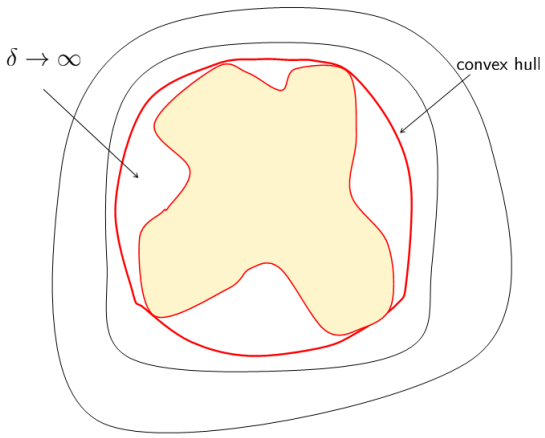
$$\min_{x \in C_{\mathcal{S}}} f(x) = \min_{x \in \mathcal{S}} f(x) \tag{4.8}$$

The drawback here is that, in general, obtaining a mathematical description of $C_{\mathcal{S}}$ is a quite difficult problem.

In the particular case of POPs, it is possible to compute a **convex relaxation os $\mathcal{S}$** depending on a parameter called the **order of relaxation** δ.



(a) Convex hull $C_{\mathcal{S}}$



(b) Relaxing the convex hull by δ

For a given δ we have different relaxed sets as you can see in the figure, moreover for $\delta \to \infty$ the relaxed set coincides with the convex hull in red.
All of this concepts are then applied to our optimization problem which is aimed to find the extrema of the Parameter Uncertainty Intervals for our problem of System Identification. In this case comes up an interesting property, that is, for sure whatever is the order δ, we are including the real PUI. More specifically, we have that for each k the relaxed solutions $\underline{\theta}_k^\delta$ and $\overline{\theta}_k^\delta$ are such that:

$$\underline{\theta}_k^\delta \leq \underline{\theta}_k \quad \overline{\theta}_k^\delta \geq \overline{\theta}_k \tag{4.9}$$

moreover it holds that:

$$\lim_{\delta\to\infty} \underline{\theta}_k^\delta = \underline{\theta}_k \quad \lim_{\delta\to\infty} \overline{\theta}_k^\delta = \overline{\theta}_k \tag{4.10}$$

In order to solve a POP by means of *Lassere convex relaxation approach* in MATLAB we use:

1. SparsePOP that given a polynomial optimization problem and the order of relaxation δ provides a *Semidefinite relaxed problem (SDP)*;
2. Finally SparsePOP calls the optimization toolbox SeDuMi which solves the given SDP.

4.4 Choosing the order of relaxation δ

Let us call x^* the global optimal solution of a given POP, and x^δ the solution of the corresponding convex relaxed solution of order δ. **How to select δ?** From the theory we know mainly **three results**:

Result 1 (R1) It holds that:

$$\lim_{\delta\to\infty} x^\delta \to x^* \tag{4.11}$$

this result is not in the most rigorous form since has been proved the convergence for the optimal value, not for the optimal solution. In our case, we are lucky since we are minimizing/maximizing the identity function.

Result 2 (R2) The *order of relaxation* δ must be such that:

$$\delta \geq \left\lceil \frac{n_{max}}{2} \right\rceil = \delta_{min} \tag{4.12}$$

where n_{max} is the maximum degree of the polynomials related to the objective and to the constraints;

Result 3 (R3) the complexity of the SDP problem obtained by applying to the original POP *convex relation techniques* grows exponentially in the order of relaxation δ and grows exponentially in the number of optimization variables (decision variables) of the original POP.

At this point we can note that such results are, on a certain extent, negative for us since: only making grow δ we can obtain a good solution, but the complexity of the problem grows exponentially with respect to δ! However there is evidence that:

Result 4 (R4) For a **large class of optimization problems** (including the one treated in this course) it is possible to prove that the convergency to the **global optimal solution** of the original POP is very fast and it is achieved for a finite value of δ. Since we are not able to increase a lot δ, this result helps us a lot. However, keep in mind that there is always the lower bound given by the (4.12).

4.5 Our case: SM SysId with EIV noise structure

What is the impact in the case we are treating POP arising from SM SysId with EIV noise structure? Since we have seen there are **bilinear constraints** (order 2 polynomials) arising, we have that $\delta_{min} = 1$; furthermore the number of optimization variables, how we have seen in previous paragraph is going to include the parameters, and the samples of the noise, which are in number of $2N$, with N being the number of experimentally collected I/O pairs.
Now, since in SysId problems N is going to be quite large in many applications, then the computational complexity is going to be **untractable**! But we can exploit the following result:

Result 5 (R5) If the original POP satisfies a property called

running intersection property[1]

it is possible to build a sequence of convex SDP relaxation involving much less variables. The problem use a lot of matrix with a lot of zeros (*sparse matrices*). This technique is called **sparse convex relaxation** and it is the one implemented in SparsePOP.

SparsePOP software is able to automatically check if the original POP satisfies such a property and if this is the case, it applies the **sparse convex relaxation**. This allows us to formulate another important result:

Result 6 (R6) The complexity of the SDP problem obtained by applying the *sparse convex relaxation* to the original POP:

- grows exponentially with the order of relaxation δ (this, still is a problem);
- grows **linearly** with the number of optimization variables

The 6th result tells us that the problem *SM-ID for LTI systems with EIV noise structure* is computationally tractable for a rather large number of I/O experimentally collected data since the problem satisfies the *running intersection property* and it is characterized by bilinear constraints.

4.6 SM SysId of LTI system with EIV using `SparsePOP`

First of all let us write down the FPS and EFPS (respectively what we called $\mathcal{D}_\theta$ and $\mathcal{D}_{\theta,\eta,\xi}$). In order to simplify the notation, but without loss of generality we consider the case where the sytem to be identified is of order $n = 2$. The definition of FPS and EFPS are as follows:

[1] Roughly speaking, even if there a lot of optimization variables, only a small number of them are appearing at the same time in a certain constraint.

$$\begin{aligned}\mathcal{D}_\theta = \big\{\theta \in \mathbb{R}^5 :\ & y(k) + \theta_1 y(k-1) + \theta_2 y(k-2) - \theta_3 u(k) + \\ & - \theta_4 u(k-1) - \theta_5 u(k-2) = 0 \quad k = 3, ..., N \\ & \tilde{y}(k) = y(k) + \eta(k), \quad \tilde{u}(k) = u(k) + \xi(k), \quad k = 1, ..., N \\ & |\eta(k)| \leq \Delta_\eta, \quad |\xi(k)| \leq \Delta_\xi,\ k = 1, ..., N\big\}\end{aligned} \tag{FPS}$$

$$\begin{aligned}\mathcal{D}_{\theta,\eta,\xi} = \big\{\theta \in \mathbb{R}^5, \eta \in \mathbb{R}^N, \xi \in \mathbb{R}^N :\ & \tilde{y}(k) - \eta(k) + \theta_1(\tilde{y}(k-1) - \eta(k-1)) \\ & + \theta_2(\tilde{y}(k-2) - \eta(k-2)) - \theta_3(\tilde{u}(k) - \xi(k)) + -\theta_4(\tilde{u}(k-1) - \xi(k-1)) \\ & - \theta_5(\tilde{u}(k-2) - \xi(k-2)) = 0,\ k = 3, ..., N \\ & |\eta(k)| \leq \Delta_\eta, \quad |\xi(k)| \leq \Delta_\xi,\ k = 1, ..., N\big\}\end{aligned} \tag{EFPS}$$

Now, we have to solve the problems in (4.4), for all of the parameters θ_j, $k = 1, ..., 5$. Then, if we have p parameters we have to solve $2p$ optimization problems which are nothing but POPs. Then, we have to properly build data structures `objPoly` and `ineqPolySys` containing respectively the information about the *objective function* and the *constraints* of the optimization problem under study.

4.6.1 Data structure `objPoly`

As far as the PUI are concerned, we know that the objective function is simply, for example:

$$f_0(\theta) = \theta_1 \tag{4.13}$$

for the 1^{st} parameter. The `objPoly` structure must be built as follows:

```
objPoly.typeCone=1;                 %no use for this field
objPoly.noTerms=1;                   %number of rows of 'supports' matrix
objPoly.dimVar=5+2*N;                %number of columnns of 'supports' matrix
objPoly.degree=1;                    %degree for the objective function
support=zeros(objPoly.dimVar,1);
support(1)=1;                        %for the first parameter
objPoly.supports=support;            %[1 0 0 ... 0]
objPoly.coef=[1;0;0;0;0;...;0]
```

4.6.2 Data structure `ineqPolySys`

In the following we are building the part of `ineqPolySys` related to the first constraint ($k = 3$). Here the `supports` matrix is not so easy to show, since it is very big! We will use dots in order to have a sort of 'contraction' of such a matrix, with the aim to understand what are its features. The fields to fill in are exactly the same with respect to `objPoly`. The constraint for which the data structure is built is:

$$\begin{aligned}&\tilde{y}(3) - \eta(3) + \theta_1\tilde{y}(2) - \theta_1\eta(2) + \theta_2\tilde{y}(1) - \theta_2\eta(1) - \theta_3\tilde{u}(3) + \theta_3\xi(3) + \\ &- \theta_4\tilde{u}(2) + \theta_4\xi(2) - \theta_5\tilde{u}(1) + \theta_5\xi(1) = 0\end{aligned} \tag{4.14}$$

```
ineqPolySys{1}.typeCone=-1;      %equality --> -1  | inequality --> 1
ineqPolySys{1}.noTerms=12;
ineqPolySys{1}.dimVar=5+2*N;
ineqPolySys{1}.degree=2;
ineqPolySys{2}.supports=support;
ineqPolySys{2}.coef=coef;
```

The field `support` and `coef` are defined as follows:

$$\texttt{support} = \begin{bmatrix}
 & \theta_1 & \theta_2 & \theta_3 & \theta_4 & \theta_5 & \eta(1) & \eta(2) & \eta(3) & \dots & \eta(N) & \xi(1) & \xi(2) & \xi(3) & \cdots & \xi(N) \\
\tilde{y}(3) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-\eta(3) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\theta_1\tilde{y}(2) & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-\theta_1\eta(2) & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\theta_2\tilde{y}(1) & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-\theta_2\eta(1) & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-\theta_3\tilde{u}(3) & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\theta_3\xi(3) & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\
-\theta_4\tilde{u}(2) & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\theta_4\xi(2) & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\
-\theta_5\tilde{u}(1) & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\theta_5\xi(1) & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0
\end{bmatrix}$$

As you can note such a matrix is a **sparse** one, since it has few non-zero elements. Finally the `coef` vector is:

$$\texttt{coef} = \begin{bmatrix} \tilde{y}(3) & -1 & \tilde{y}(2) & -1 & \tilde{y}(1) & -1 & -\tilde{u}(3) & 1 & -\tilde{u}(2) & 1 & -\tilde{u}(1) & 1 \end{bmatrix}^{\mathsf{T}}$$

Such a procedure must be repeated for all of the equality/inequality constraints of the problem. It is clear that a piece of code using a `for` cycle can help significantly in building such data structures!

4.6.3 Data structures `lbd`,`ubd`, `param`

Other data structures to be provided to the `sparsePOP()` command are `lbd`, `ubd`, `param`. As far as the first pair (lower and upper bounds on the optimization variables), you have to use $\pm 10^{10}$ in order to indicate $\pm\infty$ for the parameters θ_i, for the samples η and ξ the bounds $\pm\Delta_\eta$ and $\pm\Delta_\xi$ must be used. As far as `param` is concerned:

```
param.relaxOrder=1;                    %order of relaxation (>=delta_min)
param.POPsolver='active-set';          %type of solver to be used
```

4.6.4 Retrieving the solution of the problem

Once all of the data structures have been built, we can call the `sparsePOP` command as follows:

```
[param,SDPobjValue,POP,cpuTime,SDPsolverInfo,SDPinfo] = ...
        sparsePOP(objPoly,ineqPolySys,lbd,ubd,param);
```

In order to retrieve the (refined) solution of the optimization problem:

- `POP.objValueL` contains the **optimal solution** of the optimization problem[2];
- `POP.xVectL` contains the **optimizer** (minimizer in our case);

[2] `theta_min(i)=POP.objValueL` if you use a vector to store the $\underline{\theta}_i$. **Important:** when you are computing $\overline{\theta}_i$ you must write `theta_max(i)=-POP.objValueL`. Due to the structure of the problem also `POP.xVectL` can be used, it is not necessary the change of sign that instead is needed in the `coef` vector

References

- ✍ Cerone, Piga, and Regruto, *"Set-membership error-in-variables identification through convex relaxation techniques"*, 2011
- ✍ Cerone, Piga, and Regruto, *"Improved parameter bounds for set-membership EIV problems"*, 2011

Chapter 5

Set-Membership Identification of MIMO LTI systems with EIV noise structure

Till now we have introduced fundamental aspects about Set-Membership Identification, we have understood why it is the most realistic approach to the SysId, and – in order to go step by step – we focused our attention on a specific case on which we developed the theory: *SISO LTI systems with Errors-in-variables noise structure*. Our objective now is trying to generalize such results, including the case in which the system to identify is multi-input multi-output (MIMO) or nonlinear. In this chapter we treat the former topic.

Let us start introducing some general concepts and definition useful for the comprehension of the incoming topics. A **Multi-Input Multi-Output (MIMO)** Linear Time-Invariant system with p inputs and q outputs can be described by means of a *matrix transfer function* $G(q^{-1})$ where each element represents a SISO transfer function

$$G_{ij}(q^{-1}) = \frac{N_{ij}(q^{-1})}{D_{ij}(q^{-1})} \tag{5.1}$$

between the j-th input and the i-th output.
The I/O relationship of the system we want to study is defined by the following equation:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_q \end{bmatrix} = \underbrace{\begin{bmatrix} G_{11}(q^{-1}) & G_{12}(q^{-1}) & \dots & G_{1p}(q^{-1}) \\ G_{21}(q^{-1}) & G_{22}(q^{-1}) & \dots & G_{2p}(q^{-1}) \\ \vdots & \vdots & \ddots & \vdots \\ G_{q1}(q^{-1}) & G_{q2}(q^{-1}) & \dots & G_{qp}(q^{-1}) \end{bmatrix}}_{G(q^{-1})} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_p \end{bmatrix} \tag{5.2}$$

A block diagram representation of a MIMO system is reported below:
The reason why I use directly a transfer function description for the system I want to identify is the same for which we use it in the simple case of a SISO LTI system: since we are able to do some open-loop experiments on the plant collecting I/O data, an I/O description such as the *regression form* (that leads to a transfer function) is the most convenient way to mathematically describe the system itself!
However, we can have an objection in the sense that we can find an (apparently) **useful insight** if we start from a state-space description, roughly speaking using the matrices A, B, C, D. Indeed, recalling what is the definition of (matrix) transfer function $H(s)$ obtained starting

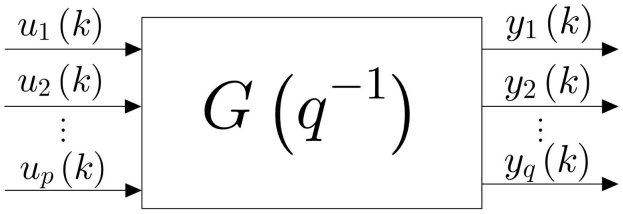


Figure 5.1: Block diagram for a MIMO system

from the state space description we know that:

$$\underbrace{H(s) = C(sI - A)^{-1}B + D}_{\text{continuous time}}, \quad \underbrace{H(z) = C(zI - A)^{-1}B + D}_{\text{discrete time}}$$

(From now on without loss of generality, for sake of simplicity we will use $(sI - A)$ for the explanation of what follows). By computing the inverse of the matrix $sI - A$ we have to divide it by its determinant $\det(sI - A)$ (which is also the *characteristic polynomial*). For this reason all of the elements of $H(s)$ have the same common denominator, resulting into the same parameters to be estimated!
Now, at the end of the day our problem is *estimating some parameters* exploiting I/O experimental data. The approaches I can use in order to continue developing the theory are:

1. Considering the denominators of the transfer functions **identical** how the state-space insight suggests us (this plays the role of an additive a-priori information);
2. Considering the transfer functions as having **different denominators** resulting in a greater number of parameters to be estimated.

It would be better to analyze the features for one approach and for the other. We can say that the most evident advantage in considering the same all of the denominators is that we have **less parameters** entering the identification procedure. On the other hand, in the second case we could have **some physical a-priori information** that suggest us something about the order of each transfer function, the order n in general can be different from one transfer function to another. In such a case we know something directly related to the number of parameters to be estimated[1]. Apparently, we are anyway tempted to say that such an information is not lost in the first case, since there could be **zero-pole cancellations** which makes also very different the several transfer functions. However this is not true, since our collected data are affected by noise and then the parameters related to the zeros/poles are not exact. It is sufficient to think about the fact we retrieve some PUIs from the Set-Membership procedure, uncertainty is embedded into the problem. This evidence suggests us that maybe the best path to be followed is not the one that apparently makes the identification problem simpler. *We will going on discussing the problem following the second approach.*

Starting from the (5.2), a generic output of the system y_i is given by:

$$y_i(k) = G_{i1}(q^{-1})u_1(k) + G_{i2}(q^{-1})u_2(k) + \cdots + G_{ip}(q^{-1})u_p(k), \quad i = 1, ..., q \tag{5.3}$$

From this follows an **important remark**: each output y_i depends on the past samples of the output itself and the samples of the inputs $u_1, ..., u_p$ not by the other outputs. In other words the

[1]For a system of order n, I have $2n + 1$ parameter to estimate.

y_i behaviours are **independent**. Therefore, a first conclusion can be drawn: *the identifiction of a MIMO LTI system with q outputs is equivalent to the identification of q **MISO** (Multi-input single-output) systems.* Now, the sub-problem to be faced is:

5.1 SM Identification of MISO LTI systems

As usual in a set-membership identification procedure we have to list all of the ingredients of our problem and then put them together to obtain the feasible parameter set. This is the what we are going to do.

5.1.1 A-priori information on the system

The system order n is assumed to be known moreover the I/O mapping can be expressed as follows:

$$y = G(q^{-1}) \begin{bmatrix} u_1 \\ \vdots \\ u_p \end{bmatrix} = \begin{bmatrix} G_1(q^{-1}) & G_2(q^{-1}) & \dots & G_p(q^{-1}) \end{bmatrix} \begin{bmatrix} u_1 \\ \vdots \\ u_p \end{bmatrix} \tag{5.4}$$

where the function $G_i(q^{-1})$ is:

$$G_i(q^{-1}) = \frac{\beta_0^i + \beta_1^i q^{-1} + ... + \beta_{n_i}^i q^{-n_i}}{1 + \alpha_1^i q^{-1} + ... + \alpha_{n_i}^i q^{-n_i}}, \quad i = 1, ..., p \tag{5.5}$$

$n_i \leq n$ is the dynamical order of $G_i(q^{-1})$. The fact that n_i could be smaller is derived from some other a-priori information, otherwise we put always $n_i = n$ for that function we do not know other insights.

5.1.2 A-priori information on the noise

With the objective of being as more general as possible, we analyze the case **errors-in-variables** (EIV) where both the inputs and the output are affected by measurement noise. A block diagram showing this set-up is given here:

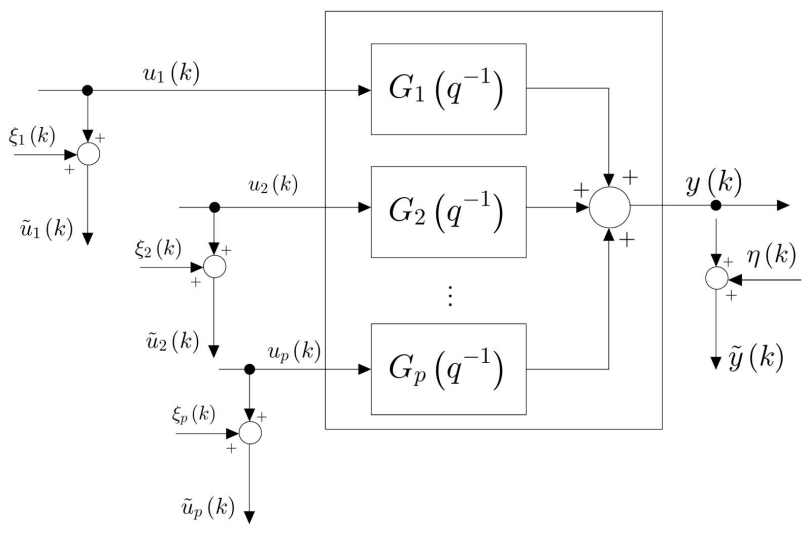


Figure 5.2: Block diagram of a MISO LTI system with EIV noise structure

As in the SISO case we make some quite realistic assumptions on the boundedness of the noise samples, that is:

$$\begin{aligned} &|\eta(k)| \leq \Delta_\eta, \quad k = 1, ..., N \\ &|\xi_i(k)| \leq \Delta_{\xi_i} \quad k = 1, ..., N \quad i = 1, ..., p \end{aligned} \tag{5.6}$$

where N as usual is the number of I/O collected data, and Δ_η, Δ_{ξ_i} are the only available information on the noise.

5.1.3 A-posteriori information: experimental data

In order to perform the identification N samples of the inputs $\tilde{u}_1(k), ..., \tilde{u}_p(k), \quad k = 1, ..., N$ and the output $\tilde{y}(k)$ must be collected by doing an *open-loop experiment* on the plant to be identified.

5.1.4 Feasible Parameter set $\mathcal{D}_\theta$

The next step is to put everything together in order to define the **feasible parameter set** $\mathcal{D}_\theta$:

$$\begin{aligned} \mathcal{D}_\theta = \big\{\theta \in \mathbb{R}^{\sum_{i=1}^{p} 2n_i+1} : \quad & y(k) = G_1(q^{-1})u_1(k) + ... + G_p(q^{-1})u_p(k), \quad k = 2n+1, ..., N \\ & y(k) = \tilde{y}(k) - \eta(k), \quad u_i(k) = \tilde{u}_i(k) - \xi_i(k), \quad i = 1, ..., p \\ & |\eta(k)| \leq \Delta_\eta, \quad |\xi_i(k)| \leq \Delta_{\xi_i} \quad k = 1, ..., N \quad i = 1, ..., p\big\} \end{aligned} \tag{5.7}$$

More explicitly we can substitute the transfer functions $G_i(q^{-1})$ with the definition we gave in (5.5), the set becomes:

$$\begin{aligned} \mathcal{D}_\theta = \big\{\theta \in \mathbb{R}^{\sum_{i=1}^{p} 2n_i+1} : \quad & y(k) = \frac{\beta_0^1 + \beta_1^1 q^{-1} + ... + \beta_{n_1}^1 q^{-n_1}}{1 + \alpha_1^1 q^{-1} + ... + \alpha_{n_1}^1 q^{-n_1}} u_1(k) + \\ & + \cdots + \frac{\beta_0^p + \beta_1^p q^{-1} + ... + \beta_{n_p}^p q^{-n_p}}{1 + \alpha_1^p q^{-1} + ... + \alpha_{n_p}^p q^{-n_p}} u_p(k), \quad k = n+1, ..., N \\ & y(k) = \tilde{y}(k) - \eta(k), \quad u_i(k) = \tilde{u}_i(k) - \xi_i(k), \quad i = 1, ..., p \\ & |\eta(k)| \leq \Delta_\eta, \quad |\xi_i(k)| \leq \Delta_{\xi_i} \quad k = 1, ..., N \quad i = 1, ..., p\big\} \end{aligned} \tag{5.8}$$

Starting from this point, how can we going on? Also in this case we could be tempted in modeling the MISO as a set of SISO, but there are some drawbacks:

- Keep in mind that for a system of order n we have to solve $2(2n+1)$ optimization problems for finding the parameter uncertainty intervals, then this will result in a high computation load;
- The temptation raises up from the moment that we grasp to the *superposition principle for LTI systems* according which if we apply the inputs one at a time, the output will show a behaviour which is related only to that input itself. Unfortunately, in real-world applications is not always possible to "turn-off" some inputs while keeping unchanged the system, this is also because the great majority of the plants are *only approximatively linear*.

In order to better understand the limitations behind the form (5.7), let us try going through the development of it. In particular, the (5.7) can be rewritten as:

$$
\begin{aligned}
\mathcal{D}_\theta =& \Big\{ \theta \in \mathbb{R}^{\sum_{i=1}^{p} 2n_i+1} : \\
& y(k) = \frac{N_1(D_2 \cdot D_3 \cdot \ldots D_p)u_1(k) + \cdots + N_p(D_1 \cdot D_2 \cdot \ldots D_{p-1})u_p(k)}{D_1 \cdot D_2 \ldots D_p}, \;\; k = n+1, ..., N \\
& y(k) = \tilde{y}(k) - \eta(k), \quad u_i(k) = \tilde{u}_i(k) - \xi_i(k), \quad k = 1, ..., N \;\; i = 1, ..., p \\
& |\eta(k)| \leq \Delta_\eta, \quad |\xi_i(k)| \leq \Delta_{\xi_i} \quad k = 1, ..., N \quad i = 1, ..., p \Big\} = \\
=& \Big\{ \theta \in \mathbb{R}^{\sum_{i=1}^{p} 2n_i+1} : \\
& (D_1 \cdot D_2 \ldots D_p)y(k) = N_1(D_2 \cdot D_3 \cdot \ldots D_p)u_1(k) + \cdots + \\
& + N_p(D_1 \cdot D_2 \cdot \ldots D_{p-1})u_p(k), \;\; k = n+1, \ldots, N \\
& y(k) = \tilde{y}(k) - \eta(k), \quad u_i(k) = \tilde{u}_i(k) - \xi_i(k), \quad k = 1, ..., N \quad i = 1, ..., p \\
& |\eta(k)| \leq \Delta_\eta, \quad |\xi_i(k)| \leq \Delta_{\xi_i} \quad k = 1, ..., N \quad i = 1, ..., p \Big\} = \\
=& \Big\{ \theta \in \mathbb{R}^{\sum_{i=1}^{p} 2n_i+1} : \\
& \Big[\underbrace{(1 + \alpha_1^1 q^{-1} + \cdots + \alpha_{n_1}^1 q^{-n_1})}_{D_1} + \cdots + \underbrace{(1 + \alpha_1^p q^{-1} + \cdots + \alpha_{n_p}^p q^{-n_p})}_{D_p} \Big] y(k) = \\
& \underbrace{(\beta_0^1 + \beta_1^1 q^{-1} + ... + \beta_{n_1}^1 q^{-n_1})}_{N_1} \underbrace{(1 + \alpha_1^2 q^{-1} + \cdots + \alpha_{n_2}^2 q^{-n_2})}_{D_2} \underbrace{(1 + \alpha_1^3 q^{-1} + \cdots + \alpha_{n_3}^3 q^{-n_3})}_{D_3} \ldots \\
& \underbrace{(1 + \alpha_1^p q^{-1} + \cdots + \alpha_{n_p}^p q^{-n_p})}_{D_p} u_1(k) + \cdots + \underbrace{(\beta_0^p + \beta_1^p q^{-1} + ... + \beta_{n_p}^p q^{-n_p})}_{N_p} \\
& \underbrace{(1 + \alpha_1^1 q^{-1} + \cdots + \alpha_{n_1}^1 q^{-n_1})}_{D_1} \underbrace{(1 + \alpha_1^2 q^{-1} + \cdots + \alpha_{n_2}^2 q^{-n_2})}_{D_2} \ldots \\
& \underbrace{(1 + \alpha_1^{p-1} q^{-1} + \cdots + \alpha_{n_{p-1}}^{p-1} q^{-n_{p-1}})}_{D_{p-1}} u_p(k) \quad k = n+1, ..., N \\
& y(k) = \tilde{y}(k) - \eta(k), \quad u_i(k) = \tilde{u}_i(k) - \xi_i(k), \quad k = 1, ..., N \quad i = 1, ..., p \\
& |\eta(k)| \leq \Delta_\eta, \quad |\xi_i(k)| \leq \Delta_{\xi_i} \quad k = 1, ..., N \quad i = 1, ..., p \Big\}
\end{aligned}
\tag{5.9}
$$

where $N_i = N_i(q^{-1})$, $D_i = D_i(q^{-1})$. The notation is not so easy to read, however we have written it rigorously to understand that at the end, we can obtain a 'standard' regression form as in the SISO case. Then, what is the problem? Explicit computation will lead to **high order polynomial constraints** in the unknown variables:

$$
\begin{aligned}
& \alpha_j^i, \quad i = 1, ..., p \quad j = 1, ..., n_i \\
& \beta_j^i, \quad i = 1, ..., p \quad j = 0, ..., n_i \\
& \eta(k), \xi_i(k), \quad k = n+1, ..., N \quad i = 1, ..., p
\end{aligned}
\tag{5.10}
$$

In general the computation of PUI_{θ_j} will lead us to POPs of order n, and whether p would be high, the order of relaxation δ is going to be *large* $\rightarrow$ bad news, since there will be the explosion of computational complexity and/or memory overflow because the size of data structures involved in the SDP problem – obtained by applying convex relaxation techniques – depends **exponentially** on the order of relaxation itself! This approach cannot be used neither for normal or large dimension identification problems. Something alternative is needed.

5.1.5 Reducing the computational complexity: partial outputs z_i

In order to reduce the computational complexity let us reformulate the problem by explicitly exploiting the fact that *a MISO system is nothing but a collection of SISO systems*. Let us call $z_1, \dots, z_p$ the **partial outputs**, that is the output of the transfer functions $G_1, G_2, \dots, G_p$ before entering into the summing junction of the output $y(k)$:

$$z_1(k) = G_1(q^{-1})u_1(k) \quad z_2(k) = G_2(q^{-1})u_2(k) \quad \dots \quad z_p(k) = G_p(q^{-1})u_p(k)$$

Now we can redefine the FPS by using such an observation obtaining:

$$\begin{aligned}\mathcal{D}_\theta = \big\{\theta \in \mathbb{R}^{\sum_{i=1}^{p} 2n_i+1} : \; & \tilde{y}(k) - \eta(k) = z_1(k) + \cdots + z_p(k) \;\; k = 1, ..., N \\ & z_i(k) = \frac{N_i(q^{-1})}{D_i(q^{-1})}(\tilde{u}_i(k) - \xi_i(k)), \;\; i = 1, \dots, p \quad k = n_i + 1, ..., N \\ & |\eta(k)| \leq \Delta_\eta, \quad |\xi_i(k)| \leq \Delta_{\xi_i} \quad k = 1, ..., N \quad i = 1, ..., p\big\}\end{aligned} \tag{5.11}$$

We cannot measure the samples $z_i(k)$ of the partial outputs, then such variables are unknowns. We have to look the problem into the **extended space** which includes all of the unknown variables (for the noise samples the same conclusion can be drawn). Then, let us define the...

5.1.6 Extended feasible parameter set $\mathcal{D}_{\theta,z,\eta,\xi}$

The **Extended feasible parameter set (EFPS)** has the following definition:

$$\begin{aligned}\mathcal{D}_{\theta,z_1,z_2,...,z_p,\eta,\xi_1,\xi_2,...,\xi_p} = \big\{&(\theta, z_1, z_2, ..., z_p, \eta, \xi_1, \xi_2, ..., \xi_p) \in \mathbb{R}^{\sum_{i=1}^{p} 2n_i+1+(2p+1)N} : \\ & \tilde{y}(k) - \eta(k) = z_1(k) + \cdots + z_p(k) \;\; k = 1, ..., N \\ & z_i(k) = -\alpha_i^i z_i(k-1) - \alpha_2^i z_i(k-2) - ... - \alpha_{n_i}^i z_i(k-n_i) + \\ & + \beta_0^i \tilde{u}_i(k) - \beta_0^i \xi(k) + ... + \beta_{n_i}^i \tilde{u}_i(k-n_i) - \beta_{n_i}^i \xi_i(k) \\ & i = 1, ..., p \quad k = n_i + 1, ..., N \\ & |\eta(k)| \leq \Delta_\eta, \quad |\xi_i(k)| \leq \Delta_{\xi_i} \quad k = 1, ..., N \quad i = 1, ..., p\big\}\end{aligned} \tag{5.12}$$

Extending the parameter space we can note that the EFPS involves only *bilinear constrints*, then we can pick again as minimum order of relaxation $\delta = 1$, that is crucial from a computational point of view.
Now, we are adding all the samples of $z_1, ..., z_p$ into the definition of the set. In principle we know that for a generic POP, the computational complexity depends *exponentially also on the optimization variables*. However, even in this case can be proved that the problem of *SM-ID of LTI MIMO systems with EIV noise structure* has the nice *running intersection property*. This is the reason why we can use *sparse convex relaxation techniques* which leads the computational complexity depending only **linearly** on the number of optimization variables.

5.1.7 PUI computation for MISO LTI systems

We have to solve a number $n_\theta = 2\big(\sum_{i=1}^{p} 2n_i + 1\big)$ optimization problems of the form:

$$PUI_j = [\underline{\theta}_j, \; \overline{\theta}_j], \quad \underline{\theta}_j = \min_{\theta,z,\eta,\xi \in \mathcal{D}_{\theta,z,\eta,\xi}} \theta_j, \quad \overline{\theta}_j = \max_{\theta,z,\eta,\xi \in \mathcal{D}_{\theta,z,\eta,\xi}} \theta_j, \quad j = 1, ..., n_\theta \tag{5.13}$$

The problem can be properly formulated using suitable data structures of `SparsePOP`. This is the end of the fundamental theoretical aspects of SM-ID of MIMO systems with EIV noise structure.

References

✍ Cerone, Razza, and Regruto, *"Set-membership errors-in-variables identification of MIMO linear systems"*, 2018

Chapter 6

Set-membership identification of nonlinear systems

6.1 Introduction

Let us continue to generalize our Set-Membership identification framework introducing the theoretical concepts useful to identify **SISO nonlinear systems** in the regression form we have seen in the very first part of these notes:

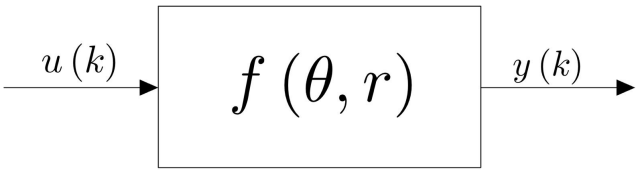


Figure 6.1: Nonlinear SISO system (regression form)

$$y(k) = f(\theta, r(k)) = f(\theta, [y(k-1), y(k-2), \ldots, y(k-n), u(k), u(k-1), \ldots, u(k-m)]) \quad (6.1)$$

where θ is the vector with the parameters of the model, while $r(k)$ is the so-called **regressor**. We specialize the class f of functions we deal with, in particular they are **multivariate polynomial function of the regressor** $r(k)$. In this context we propose some examples which, as usual, are useful to both introduce and better clarify the setting to which we are approaching. Through all of the examples we do two fundamental quantities are:

- The **dynamical order of the system** n; this is related to the number of past samples we need in order to describe the generic system output $y(k)$;
- The **degree** (or **order**) $\deg(f)$ of the multivariate polynomial f describing the nonlinear system in the parametrized form.

Example 1

Suppose that from the physical insights we know that the system has $n = 1$ and $\deg(f) = 2$. This is the same to state that the equation describing the I/O of the system is:

$$y(k) = f(\theta, r(k)) = f(\theta, [y(k-1), u(k), u(k-1)]) = \quad (6.2)$$
$$= \theta_1 y(k-1) + \theta_2 u(k) + \theta_3 u(k-1) + \theta_4 y^2(k-1) + \theta_5 u^2(k) + \theta_6 u^2(k-1) + \quad (6.3)$$
$$+ \theta_7 y(k-1)u(k) + \theta_8 y(k-1)u(k-1) + \theta_9 u(k)u(k-1) \quad (6.4)$$

Since $\deg(f) = 2$, in order to avoid loosing of generality we have to consider also the mixed monomial terms of order 2.

Example 2

Let us consider, now, the case in which both the dynamical order and multivariate polynomial degree are the same as before, on the other hand from the physical insights we know that the structure of the I/O equation is:

$$y(k) = \theta_1 y(k-1) + \theta_2 u(k) + \theta_3 u^2(k-1) + \theta_4 y(k-1)u(k)$$

It is not said that all of the terms are present, in this case there is some reason for which this happens.

Example 3

Finally, here we consider the case in which, again, n, $\deg(f)$ are equal while not having any useful physical insights which can allow us make a guess on the shape of $f(\theta, r)$. In this particular case some general assumptions on the continuity[1] of f are needed, otherwise we cannot say anything and the identification problem is not tractable. This example gives the possibility to introduce to cite an important result.

6.2 Dealing with nonlinearity of the problem

Whether one can assume that the function f can leverage on some continuity properties,the **Stone-Weirstrass theorem** can be used. It allows us with the possibility to approximate as well as we want the true nonlinear function with a multivariate polynomial over a *compact set*. In spite this is a nice result, it is not going to provide a way to build such polynomial, since it is an *existency theorem*.

6.2.1 On the choice of the polynomial structure

When we are not provided with physical insights making us to neglect some terms of the multivariate polynomial, we can always start from a *complete description*, next when the identification procedure is performed, the terms which are not included into the model description will have an uncertainty interval whose central estimate is approximatively equal to zero.

Having said that, the choice of $\deg(f)$ can be done by using the following **procedure**:

0. We can start from [2]$\deg(f) = 2$ and we try to solve the problem going through the computation of the feasible parameter set $\mathcal{D}_\theta$ putting together all the available information;
1. **Is the $\mathcal{D}_\theta$ empty?**[3]
   - NO In this case we have to check what are the (relaxed) PUI, in particular we have to check that

$$\text{sign}(\underline{\theta}_i) = \text{sign}(\overline{\theta}_i) \quad i = 1, \dots, n_\theta \tag{6.5}$$

[1] for example $f \in \mathcal{C}^0$ or $f \in \mathcal{C}^1$

[2] If there was $\deg(f) = 1$ the system under study is linear

[3] Practically speaking, using `SparsePOP` a symptom of the fact that $\mathcal{D}_\theta = \varnothing \iff$ POP infeasible can be read in the output variable `exitflag`, in particuar its negativeness implies the infeasibility of the POP.

Whether this is the case we can accept our solution and use the retrieved interval in order to simulate control the identified system. On the other hand, we continue to investigate in the fact that, maybe the the length of the PUI

$$|\underline{\theta}_i - \overline{\theta}_i| \leq \delta \tag{6.6}$$

for some small δ. In pratice this is the case when, if for the other parameters the property (6.5) is fulfilled, the procedure is saying us that $\theta_i = 0$. On the contrary if (6.5) is not satisfied and the noise bound is big[4], it would be better if the sensor is changed because it is not properly measuring the data to be used in the identification process. Finally on the other side when the *sign concordance property* is not fulfilled and the noise relative bound is acceptable, **the order of the polynomial ought to be increased** $\rightarrow$ try $\deg(f) = \deg(f) + 1$.

1. **YES** Also in this case the $\deg(f)$ is too **low**, then you have to increase it.

Remark (**Unknown dynamical order** n). A similar *trial and error* approach can also be applied to the case (both linear and nonlinear systems) when the *dynamical order* n is not exactly known from the physical insights. This will be useful especially when the Direct Data-driven control technique will be explained.

6.2.2 Additional comments on the PUIs extrema sign concordance

After this discussion, let us give further attention on the property (6.5). The reason why we must reject the PUIs with non-concordant sign is that we cannot even know the sign for a certain parameter. Whether we want to design a simple stabilizing *feedback controller*, the fact that we do not have the information on the sign does not protect us from building a positive feedback(!), making unstable the closed loop system. We conclude this part saying that the **relative size of the uncertainty** is given by:

$$\Delta_i = \left| \frac{\overline{\theta}_i - \underline{\theta}_i}{\theta_i^c} \right|, \qquad \theta_i^c = \frac{\overline{\theta}_i + \underline{\theta}_i}{2} \tag{6.7}$$

where θ_i^c is the so-called **central estimate** which can leverage on some nice properties how we will see in one of the next chapters. When the sign of the extrema is not the same, the uncertainty relative size is greater than the 1 (that is 100%).

6.3 Toward the computational load reduction

We have seen in the previous sections a trial and error procedure for choosing either the dynamical order or the order of the multivariate polynomial f. The drawback of such an approach is that the computational complexity explode when the such a polynomial have high order. Practical examples are the best way to understand things. Let us do, then, an example.

[4]We can quantify it in percentage terms with respect to the measured input $\tilde{u}(k)$ or output $\tilde{y}(k)$.

Example

Suppose we want to estimate the parameters of the following nonlinear MISO system:

$$y(k) = \theta_1 u_1(k) + \theta_2 u_1(k)u_2(k) + \cdots + +\theta_p u_1(k)\dots u_p(k)$$

If we add the fact that both the inputs and the output are corrupted by bounded measurement noise. The feasible parameter set is defined as:

$$\begin{aligned}\mathcal{D}_\theta = \{\theta \in \mathbb{R}^p : \tilde{y}(k) - \eta(k) = \theta_1(\tilde{u}(k) - \xi_1(k)) + \theta_2(\tilde{u}_1(k) - \xi_1(k))(\tilde{u}(k) - \xi_2(k)) + \dots \\ \theta_p(\tilde{u}_1(k) - \xi_1(k))\dots(\tilde{u}_p(k) - \xi_p(k)) \quad k = n+1, ..., N \\ |\eta(k)| \leq \Delta_\eta \ k = 1, ..., N \\ |\xi_i(k)| \leq \Delta_x i \ k = 1, ..., N \ i = 1, ..., p\}\end{aligned} \tag{6.8}$$

Now, considering the problem of computing the PUIs, we have to solve a POP whose constraints are at most of order $p+1$. Moreover, if p is high, the minimum order of relaxation δ_{min} is high. Bad symptom! In order to **reduce the computational complexity** of the POP we can reformulate it by introducing new variables z_j as follows:

$$\begin{aligned}&\min_{\theta,\xi,\eta,z} \theta_1 \\ &\text{s.t.} \quad z_1(k) = \xi_1(k)\cdot\xi_2(k),\ z_2(k) = z_1(k)\cdot\xi_3(k),\ \dots,\ z_{p-1}(k) = z_{p-2}(k)\cdot\xi_p(k) \\ &\tilde{y}(k) - \eta(k) = \theta_1 u_1(k) + \theta_1\xi_1(k) + \theta_2 + \cdots + \theta_p z_{p-1}\end{aligned}$$

All of the constraints entering the optimization problem are now bilinear, then we can pick as minimum order of relaxation $\delta_{\min} = 1$.

6.4 Block-oriented nonlinear systems

The SysId procedure for nonlinear system is quite challenging. When we have addressed the problem of identifying an LTI system (both SISO and MIMO), we had the information about the transfer function that was of crucial importance. For nonlinear system we cannot exploit such an information. In the most general case, assuming we can do some continuity assumptions, we can go through the steps we have described in order to identify the system, otherwise we have to exploit as much as possible the *a-priori information* available on that system. In many situations we can know something about the **internal structure of the nonlinear system**. To this aim we propose the so-called **block-structured nonlinear systems**.

Definition 6.4.1 (Block oriented nonlinear systems)**.** Block oriented nonlinear systems are nonlinear dynamical systems which are obtained by **connecting together** a number of subsystems such that each one of the subsystems can either be:

- a *dynamical LTI system*
- a *static nonlinear system*

6.4.1 Hammerstein system

This structure is characterized by the following features:

1. Internal signal $x(k)$ cannot be measured;
2. It is useful to describe physical systems which are essentially LTI systems but with a **significant input nonlinearity** such as input saturation, dead-zone and any other non linear effect in the actuation part.

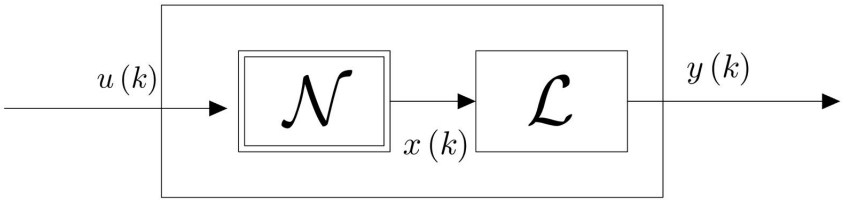


Figure 6.2: Hammerstein system (block diagram)

6.4.2 Wiener system

The **Wiener system** has the nonlinear part $\mathcal{N}$ toward the output. $\mathcal{N}$ is a *linear static map* from the output of the linear system to the output of the nonlinear block. Practical examples of such a system are: (i) electrical circuits with nonlinear components (e.g., diodes, transistors), (ii) mechanical systems with nonlinear friction or backlash. Biological systems where feedback mechanisms include saturation effects.

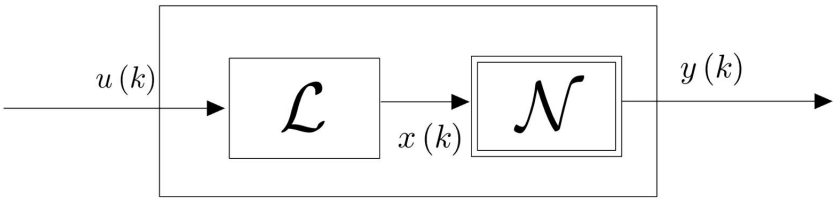


Figure 6.3: Wiener system

Remark (**Features of the Wiener system**). (i) the $x(k)$ signal cannot be measured; (ii) such a type of system is useful to describe systems which are essentially LTI but with a significant **ouput nonlinearity**.

6.4.3 Lur'e system

Lur'e system is a type of block-oriented nonlinear system where the nonlinear static map is in the feedback. Real-wolrd examples in which such systems can occur are: (i) electrical circuits with nonlinear components (e.g., diodes, transistors). (ii) mechanical systems with nonlinear friction or backlash. (ii) biological systems where feedback mechanisms include saturation effects. Parameters bound estimation for the linear and nonlinear blocks are treated in Cerone, Piga, and Regruto, "Bounding the parameters of block-structured nonlinear feedback systems" (2013).

When dealing with **block-structured** systems, we assume that the nonlinear static subsystems can be described by means of polynomial functions, so that all the unmeasurable inner signal will be considered as optimization variables in the SysId problem.

6.5 SM-ID of an Hammerstein system

6.5.1 More details on the blocks

In the Hammerstein system,the nonlinear part is often referred as $\mathcal{N}(\cdot)$ and it is a function of $u(k)$, while the linear part $\mathcal{L}$ is represented by a transfer function $G(q^{-1})$. Then:

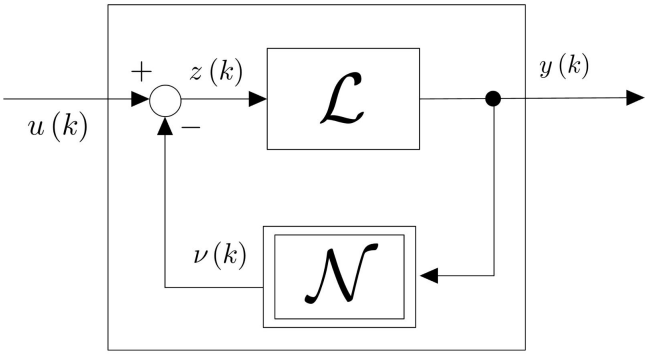


Figure 6.4: Lur'e system

$$\begin{cases} y(k) & = -\alpha_1 y(k-2) - \alpha_2 y(k-2) - ... - \alpha_n y(k-n) + \\ & \beta_0 x(k) + \beta_1 x(k-1) + ... + \beta_n x(k-n) \\ x(k) & = \mathcal{N}(u(k)) \end{cases} \tag{6.9}$$

where $\mathcal{N}(\cdot)$ is a **static linear map** between the input $u(k)$ and the internal output $x(k)$.
As an *additional a-priori information* about the structure of the static map, we can saym that:

$$\begin{aligned} x(k) = \mathcal{N}(u(k)) &= \sum_{j=1}^{m} \gamma_j \ \phi_j \ (u(k)) = \\ &= \gamma_1 \ \phi_1 \ (u(k)) + \gamma_2 \ \phi_2 \ (u(k)) + ... + \gamma_m \ \phi_m \ (u(k)) \end{aligned} \tag{6.10}$$

$\{\phi\}_j$, $j = 1, ..., m$ is a **set of basis function** selected on the base of our physical insights. In the case we do not have any other specifical information on the form of $\mathcal{N}$, we can surely selected (keeping in mind the Stone-Weirstrass theorem) the *standard polynomial basis function*:

$$\phi_j = u^j(k) \quad \forall j = 1, ..., m$$

6.5.2 System Identification procedure

As usual at this stage, when we want to formulate the problem of SysId we havo to put together a-priori and a-posteriori information.

Prior information on the system Since we have said that the system is an *Hammerstein one* we know that:

1. Nonlinear part ($\mathcal{N}$) can be espressed how we have just seen. Moreover, note that there is no constant term (in other words there is no term with the power 0), since it is of little practical interest.
2. Linear part ($\mathcal{L}$) is described by $y(k)$ as in equation (6.9)

Prior information on the noise Here we assume an **Output Error (OE)** setting for sake of simplicity. We assume as always the boundedness of the noise samples

$$|\eta(k)| \leq \Delta_\eta \quad k = 1, ..., N$$

A-posteriori information: collected data Are the N pairs $(u(k), \tilde{y}(k))$ experimentally collected by performing an open-loop experiment on the plant.

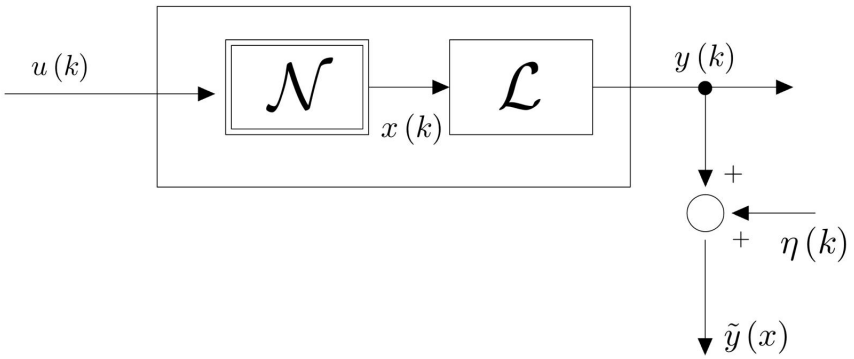


Figure 6.5: Hammerstein system (OE setting)

Putting all together we can retrieve the **Feasible parameter Set** where the parameters are $\theta = [\alpha_1 \dots \alpha_n \ \beta_0 \dots \beta_n \ \gamma_1 \dots \gamma_m]$[5]:

$$\begin{aligned}\mathcal{D}_\theta = \big\{\theta \in \mathbb{R}^{2n+1+m} : \ & y(k) = -\alpha_1 y(k-1) + ... + \beta_n x(k-n) \quad \forall k = n+1, ..., n \\ & x(k) = \gamma_1 u(k) + \gamma_2 u^2(k) + ... + \gamma_m u^m(k) \ k = 1, ..., n \\ & \tilde{y}(k) = y(k) + \eta(k), \quad |\eta(k)| \leq \Delta_\eta \quad k = 1, ..., N\big\}\end{aligned} \tag{6.11}$$

We know that we have to go on by computing the minimum for each theta in the derived set, that must be extended. There are mainly two approaches by which we can go on. In the former we *eliminate the dependence on the inner variable* x by substituting it in the expression defining $\mathcal{L}$ while the former is about considering also the samples of the inner signal $x(k)$ as optimization variables. In the following we will consider separately the parameters of the static and the dynamic block of the overall Hammerstein system.

6.5.3 First approach: toward the definition of $\mathcal{D}_{\theta,\gamma,\eta}$

By using this first approach, we can eliminate the dependence of the set on $x(k)$ by substituting it in the first equation. The resulting **Extended feasible parameter set** is:

$$\begin{aligned}\mathcal{D}_{\theta,\gamma,\eta} = \{\theta \in \mathbb{R}^{2n+1}, \gamma \in \mathbb{R}^m, \eta \in \mathbb{R}^N : \ & \tilde{y}(k) - \eta(k) = -\alpha_1[\tilde{y}(k-1) - \eta(k-1)] - \dots - \\ & - \alpha_n[\tilde{y}(k-n) - \eta(k-n)] + \beta_0\gamma_1 u(k) + \beta_0\gamma_2 u^2(k) + \dots + \beta_0\gamma_m u^m(k) + \\ & + ... + \beta_n\gamma_1 u(k-n) + ... + \beta_n\gamma_m u^m(k-n) \quad k = n+1, ..., N \\ & |\eta(k)| \leq \Delta_\eta \quad k = 1, ..., N\}\end{aligned} \tag{6.12}$$

This set is defined by bilinear constraints; the PUIs can be computed as usual as:

$$PUI_{\theta_j} = [\underline{\theta}_j, \overline{\theta}_j] \qquad \underline{\theta}_j = \min_{\theta,\gamma,\eta \in \mathcal{D}_{\theta,\gamma,\eta}} \theta_j, \quad \overline{\theta}_j = \max_{\theta,\gamma,\eta \in \mathcal{D}_{\theta,\gamma,\eta}} \theta_j \tag{6.13}$$

Remark. **Approach 1** leads to POPs with:

1. $2n + 1 + m + N$ variables;
2. max degree of the constraints equal to 2;
3. N-n *bilinear* polynomial constraints of order 2;
4. 2N linear constraints ($|\eta(k)| \leq \Delta_\eta$)

[5]We can also split them into two parts: γ that are the ones related to $\mathcal{N}$ and θ which are the ones related to $\mathcal{L}$. In the following we will use this approach.

6.5.4 Second approach: toward the definition of $\mathcal{D}_{\theta,\gamma,x,\eta}$

Here we are not going to replace the $x(k)$ samples, we consider them as optimization variables. This leads to the following description of the EFPS:

$$\begin{aligned}\mathcal{D}_{\theta,\gamma,x,\eta} = \big\{&\theta \in \mathbb{R}^{2n+1}, \gamma \in \mathbb{R}^m, x \in \mathbb{R}^N, \eta \in \mathbb{R}^N : \\ &y(k) = -\alpha_1 y(k-1) + ... + \beta_n x(k-n) \quad \forall k = n+1, ..., N \\ &x(k) = \gamma_1 u(k) + \gamma_2 u^2(k) + ... + \gamma_m u^m(k) \; k = 1, ..., N \\ &\tilde{y}(k) = y(k) + \eta(k), \quad |\eta(k)| \leq \Delta_\eta \quad k = 1, ..., N\big\}\end{aligned} \tag{6.14}$$

Again, we have bilinear constraints and the problem of finding the *parameter uncertainty interval* can be formulated as usual:

$$PUI_{\theta_j} = [\underline{\theta}_j, \overline{\theta}_j] \qquad \underline{\theta}_j = \min_{\theta,\gamma,x,\eta \in \mathcal{D}_{\theta,\gamma,x,\eta}} \theta_j, \quad \overline{\theta}_j = \max_{\theta,\gamma,x,\eta \in \mathcal{D}_{\theta,\gamma,x,\eta}} \theta_j \tag{6.15}$$

Remark. **Approach 2** leads to POPs with:

1. $2n + 1 + m + 2N$ optimization variables
2. max degree of the polynomial constraints equal to 2
3. $N - n$ polynomial constraints of order 2
4. $2N$ linear constraints (deriving from η, x)

The problem we have just presented satisfy the *running intersection property* so that the *Sparse convex relaxation* can be applied. Moreover the fact that we have bilinear constraints allows us to pick as minimum order of relaxation $\delta_{min} = 1$.
At this point it seems that all clear and suitable to solve the problem by using `sparsePOP` (for example). There is another problem to face related to the **well-posedness** of such a problem.

6.6 Identifiability for Hammerstein system

Let us consider the following Hammerstein systems:

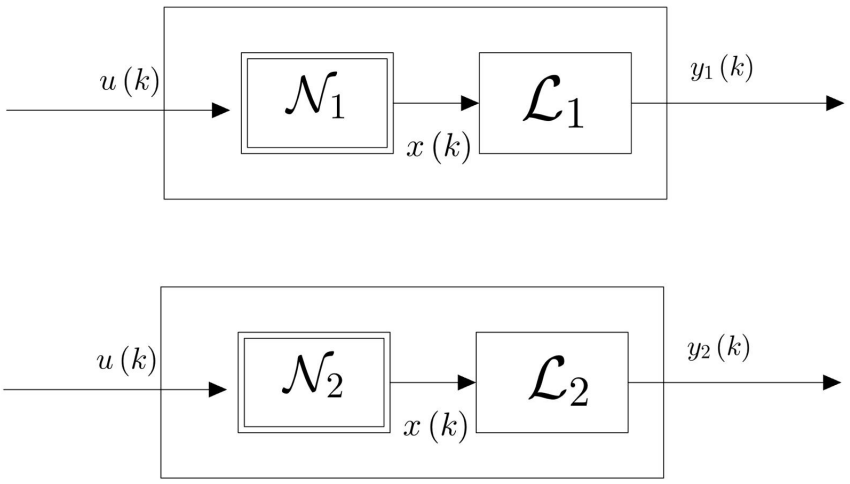


Figure 6.6: Two Hammerstein systems identical from the Input/Output point of view

It can be easily proved that $y_1(k) = y_2(k)$ provided that $\mathcal{N}_2 = \mathcal{N}_1/\alpha$ and $\mathcal{L}_2 = \alpha\mathcal{L}_1 \; \forall \alpha \in \mathbb{R}$. The Hammerstein system is not identifiable because we have an infinite number of solutions of

the identification problem which are perfectly equivalent from the input/output point of view. We are able to draw two important conclusions:

Result 1 No matter what is the technique that we are going to use we will not be able to estimate the exact parameters of the Hammerstein system generating the data, even if we are collecting *noiseless data* and $N \to \infty$. This leads to an intrinsic **unidentifiability**.

Result 2 Since we have an infinite number of *Hammerstein model* which are able to provide exactly the same I/O behaviour of the true one, our numerical procedure for the computation of the PUI **is going to fail**. This is due to the fact the FPS will be **structurally unbounded**.

From such results we understand that we need an **additional constraint**, in principle any constraint so that the I/O behaviour is not changed. For example:

1. We can fix the DC-gain of $G(q^{-1})$ to a given value;
2. We can fix one of the parameter of $\mathcal{N}$ to a fixed.

In general: any normalization of either G or N is fine! The easiest approach that is not leading to any loss of generality is to set the DC-gain of $G(q^{-1})$ to 1. If we impose such a condition we will obtain another constraint:

$$\frac{\beta_0 + \beta_1 + \beta_2 + ... + \beta_n}{1 + \alpha_1 + \alpha_2 + ... + \alpha_n} = 1 \tag{6.16}$$

This is nothing but a **linear additional constraint** on the system parameters. The additive condition can be also a different one, for example that one of the γ_i coefficients is fixed. For example $\gamma_1 = 1$.

6.7 Extension for non-polynomial $\{\phi\}_j$ basis function

At this stage we wonder if the presented SM-ID method for the Hammerstein system can be extended to the case where $\mathcal{N}$ is no more polynomial. The answer is positive provided that $\mathcal{N}$ is polynomially parametrized that is:

$$\mathcal{N}(\cdot): \; x(k) = \gamma_1\varphi_1(u(k)) + \gamma_2\varphi_2(u(k)) + ... + \gamma_m\varphi_m(u(k)) \tag{6.17}$$

This is quite useful when either the nonlinearity is **difficult** to be approximated by polynomials (hard nonlinearity such as **input saturation**), or the application explicitly suggests a precise choice of $\varphi_1, \; ... \; , \varphi_m$. This last case arises when we have for example a **dead zone**. The following figure is taken from [20]:

6.8 SM-ID of a Wiener and Lur'e system

6.8.1 Wiener SysId

The SM-ID procedure for the *Wiener system* is practically speaking the same with respect to the one we have seen for the *Hammerstein system* including the discussion about the Identifiability. What changes is the way the nonlinearity enters into the block-oriented system: indeed, here the nonlinear block is in the output of the system, then after the linear block $\mathcal{L}$.

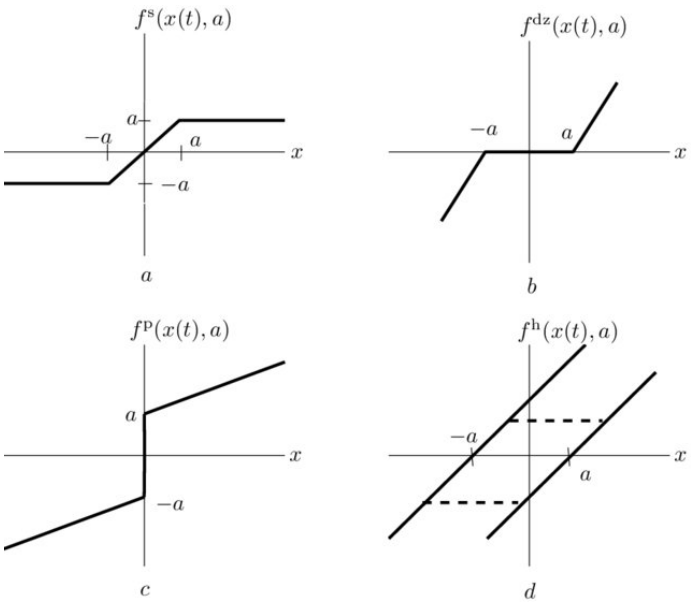


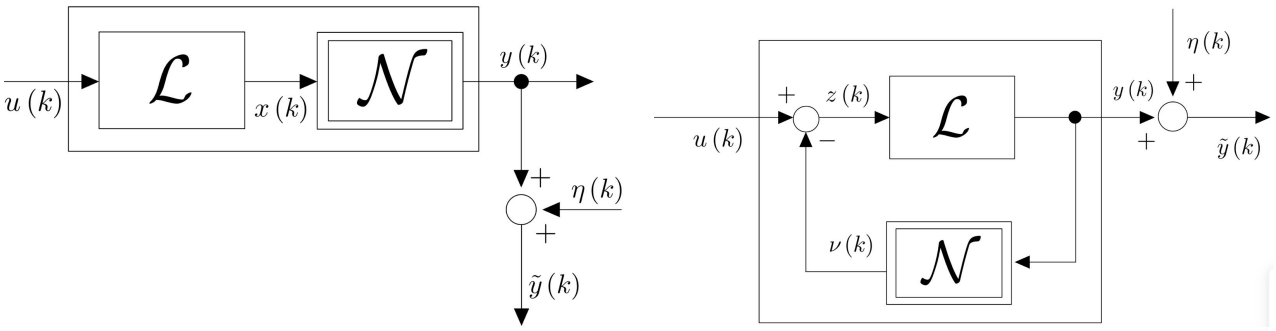
Figure 6.7: Examples of hard nonlinearities: (a)Saturation, (b)Dead zone, (c) Preload, (d)Hysteresis

6.8.2 Lur'e SysId

As far as the identification of the *Lur'e system* is concerned, there is a more tricky three-stage to follow. The main reference is Cerone et al. [5]. Just for mention, we say that such a procedure is done by the following steps:

1. Estimation of the parameter γ_i of the nonlinear static map $\mathcal{N}$ while the linear part is at steady state;
2. Estimation of a bound for the inner unmeasurable variable z and ν;
3. Using the obtained information, and assuming that the output of the system is affected by noise η, we estimate bounds on the parameters of the linear model $\mathcal{L}$ using an *Error-in-variables* (EIV) approach.

Moreover, equivalently than the other cases, also here a condition on the DC gain of $\mathcal{L}$ is needed in order to make the Lur'e system identifiable. The experimental settings to employ in the case of SM-ID for Wiener and Lur'e model is reported in the Figure 6.8.



(a) Wiener OE (b) Lur'e OE

Figure 6.8: Wiener and Lur'e systems' experimental settings

6.9 SM Identification of LFT Block-structured nonlinear systems

We have introduced some basic block-structured nonlinear systems configuration such as *Hammerstein*, *Wiener* and *Lur'e*, however in real-world applications, frequently more complex structures can appear.
In recent years a **more general structure** based on a block-structured linear system in linear fractional transformation form (LFT) has been used. This is characterized by the *feedback interconnection* of a *multivariable linear system* $\mathcal{L}$ and a *static multivariable nonlinear map* $\mathcal{N}$. This way of formulating the problem remains rather rich including also NARX models. In this section we call such a description **General Block-structured configuration (GBSC)**. For sake of generality we assume that both input and output of such a system are corrupted by Unknown but bounded (UBB) measurement noise. The main reference for this section is the article *"A unified framework for the identification of a general class of multivariable nonlinear block-structured systems"* (Cerone, Razza, and Regruto [10]).

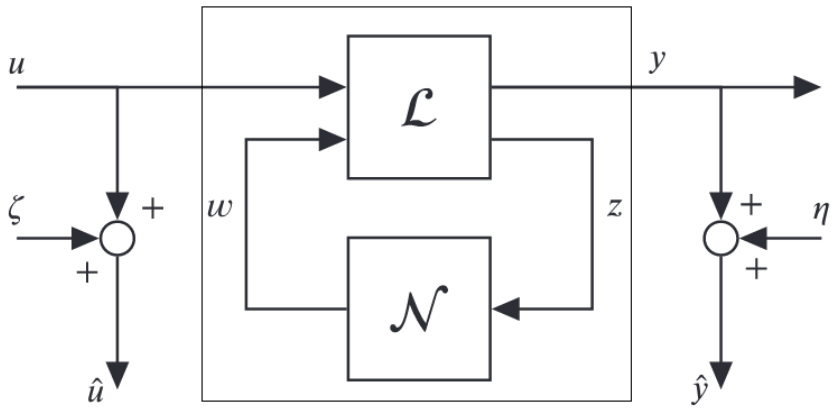


Figure 6.9: General block-structured configuration (GBSC)

Let us analyze more in details the structure of $\mathcal{L}$ and $\mathcal{N}$ in order to better understand the generality gained by using such a LFT configuration.

6.9.1 Multivariable linear block $\mathcal{L}$

$\mathcal{L}$ is a multivariable linear synamic system described by a MIMO transfer matrix mapping the noiseless inputs $u_k \in \mathbb{R}^{n_u}$ and $w_k \in \mathbb{R}^{n_w}$ into the outputs $y_k \in \mathbb{R}^{n_y}$ and $z_k \in \mathbb{R}^{n_z}$. For simplicity and clarity we denote the generic sample $u(k)$ with u_k. Then, we can write:

$$\begin{bmatrix} y_k \\ z_k \end{bmatrix} = \mathcal{L} \begin{bmatrix} u_k \\ w_k \end{bmatrix} = \begin{bmatrix} \mathcal{L}_{yu} & \mathcal{L}_{yw} \\ \mathcal{L}_{zu} & \mathcal{L}_{zw} \end{bmatrix} \begin{bmatrix} u_k \\ w_k \end{bmatrix} \tag{6.18}$$

where the terms $\mathcal{L}_{yu}$, $\mathcal{L}_{yw}$ $\mathcal{L}_{zu}$, $\mathcal{L}_{zw}$ are submatrices of suitable dimensions. The generic (i,j)-th block of the block $\mathcal{L}$ is – as you can imagine – a SISO transfer function of the form:

$$\mathcal{L}^{(i,j)} = G^{(i,j)}(q^{-1}) = \frac{\sum_m^{n_b^{(i,j)}} b_m^{(i,j)} q^{-m}}{1 + \sum_{k=1}^{n_a^{(i,j)}} a_k^{(i,j)} q^{-k}} \tag{6.19}$$

the dynamical order $n_b^{(i,j)}$ and $n_a^{(i,j)}$ are assumed to be known.

6.9.2 Static map $\mathcal{N}$

The nonlinear block $\mathcal{N}$ is an **array of nonlinear static functions** where the generic i-th entry is defined as:

$$w_t^{(i)} = f^{(i)}(z_k), \quad i = 1, ..., n_w \tag{6.20}$$

$$f^{(i)}(z_k) = \sum_{k=1}^{n_\varphi^{(i)}} \varphi_k^{(i)} \Phi_k^i(z_k) \tag{6.21}$$

where $\Phi_k^i : \mathbb{R}^{n_z} \to \mathbb{R}$ are known **multivariate polynomial basis functions** in the variable z. How we mentioned in the introduction both inputs u_k and output y_k are affected by bounded noise while the inner variables z_k and w_k are not measurable. Formally:

$$\hat{u_k} = u_k + \zeta_k \quad \hat{y_k} = y_k + \eta_k \tag{6.22}$$

$$|\zeta_k| \leq \Delta_\zeta \quad |\eta_k| \leq \Delta_\eta \tag{6.23}$$

6.10 SM Identification setting

Our objective is the estimation of the **parameter vector** θ that in our context is done by two subvectors $\theta_\mathcal{L}$ and $\theta_\mathcal{N}$ related to – respectively – the linear and nonlinear part of the LFT block. Then, in formulas:

$$\theta = [\theta_\mathcal{L}, \theta_\mathcal{N}]^T \quad \theta_\mathcal{L} \in \mathbb{R}^{p_l} \quad \theta_\mathcal{N} \in \mathbb{R}^{p_n} \tag{6.24}$$

$$\theta_\mathcal{L} = [\theta_\mathcal{L}^{(1,1)}, ..., \theta_\mathcal{L}^{(1,n_u+n_w)}, ..., \theta_\mathcal{L}^{(n_y+n_z,1)}, ..., \theta_\mathcal{L}^{(n_y+n_z,n_u+n_w)}]^T \tag{6.25}$$

$$\theta_\mathcal{L}^{(i,j)} = [a_1^{(1,1)}, ..., a_{n_a^{(i,j)}}^{(i,j)}, b_0^{(i,j)}, ..., b_{n_b^{(i,j)}}^{(i,j)}]^T \tag{6.26}$$

$$\theta_\mathcal{N} = [\theta_\mathcal{N}^{(1)}, ..., \theta_\mathcal{N}^{(n_w)}]^T, \quad \theta_\mathcal{N}^i = [\varphi_1^{(i)}, ..., \varphi_{n_\varphi^{(i)}}^{(i)}]^T \tag{6.27}$$

where $p = p_n + p_l$ is the number of parameters to be estimated.

As we discussed in the case of *Hammerstein system*, also the LFT-GBSC is non-identifiable. For a complete proof of this fact see [10]. We can force again, without loss of generality that the DC-gain of each single SISO transfer function of the $\mathcal{L}$ block is equal to one, that is:

$$\frac{\sum_m^{n_b^{(i,j)}} b_m^{(i,j)}}{1 + \sum_{k=1}^{n_a^{(i,j)}} a_k^{(i,j)}} = 1 \quad \forall (i,j) \tag{6.28}$$

We can write now the expression of the **extended feasible parameter set (EFPS)**. In order to make more readable the formulas, we can group in a single vector r_k the inputs and in a single vector s_k the outputs where

$$r_k = [u_k, w_k]^T \in \mathbb{R}^{n_r} \quad s_k = [y_k, z_k]^T \in \mathbb{R}^{n_s} \tag{6.29}$$

As we did in the case of SM-ID of MIMO LTI systems, is useful to introduce *partial outputs* $v_k^{(i,j)}$ and make the assumption that each MIMO system can be written as the superposition of more MISO systems $s_k^{(i)}$. In the following, for clarity we call simply $\mathcal{D}$ the EFPS $\mathcal{D}_{\theta,v^{(i,j)},\eta,\zeta,w,z}$:

$$
\begin{aligned}
\mathcal{D} = \Big\{ & (\theta, v^{(i,j)}, \eta, \zeta, w, z) \in \mathbb{R}^{p+(n_r+n_s)N} : \\
& v_k^{(i,j)} = \sum_{m=0}^{n_b^{(i,j)}} b^{(i,j)} r_{k-m}^{(j)} - \sum_{h=1}^{n_a^{(i,j)}} a^{(i,j)} v_{k-h}^{(i,j)} \\
& s_k^{(i)} = \sum_{h=1}^{n_r} v_h^{(i,h)}, w_t^{(l)} = \sum_{h=1}^{n_\varphi^{(l)}} \varphi_k^{(l)} \Phi_k^l(z_k) \\
& \hat{y}_k^{(o)} = y_k^{(o)} + \eta_k, \quad \|\eta\| \leq \Delta_\eta \\
& \hat{u}_k^{(i)} = u_k^{(i)} + \zeta_k, \quad \|\zeta\| \leq \Delta_\zeta \\
& k = 1, ..., N, \ i = 1, ..., n_s, \ j = 1, ..., n_s \\
& l = 1, ..., n_w, \ o = 1, ..., n_y, \ i = 1, ..., n_u \Big\}
\end{aligned} \tag{6.30}
$$

As you know, in the SM framework you define the Parameter Uncertainty Interval of the generic parameter θ_k of θ as:

$$PUI_k = [\underline{\theta}_k, \overline{\theta}_k], \ i = 1, ..., p$$

where the extrema are:

$$\underline{\theta}_k = \min_{(\theta, v^{(i,j)}, \eta, \zeta, w, z) \in \mathcal{D}} \theta_k = \min_{\theta \in \mathcal{D}} \theta_k \tag{6.31}$$

$$\overline{\theta}_k = \max_{(\theta, v^{(i,j)}, \eta, \zeta, w, z) \in \mathcal{D}} \theta_k = \max_{\theta \in \mathcal{D}} \theta_k \tag{6.32}$$

In general the functional $J(\theta)$ of the optimization problem could be also something different[6]. Focusing on the case we have presented the optimization problems to be solved become:

$$\underline{\theta}_k = \min_{\theta, v^{(i,j)}, \eta, \zeta, w, z} \theta_k$$

subject to:

$$
\begin{cases}
v_k^{(i,j)} = \sum_{m=0}^{n_b^{(i,j)}} b^{(i,j)} r_{k-m}^{(j)} - \sum_{h=1}^{n_a^{(i,j)}} a^{(i,j)} v_{k-h}^{(i,j)} \\
s_k^{(i)} = \sum_{h=1}^{n_r} v_h^{(i,h)} \\
w_t^{(l)} = \sum_{h=1}^{n_\varphi^{(l)}} \varphi_k^{(l)} \Phi_k^l(z_k) \\
\hat{y}_k^{(o)} = y_k^{(o)} + \eta_k, \quad \|\eta\| \leq \Delta_\eta \\
\hat{u}_k^{(i)} = u_k^{(i)} + \zeta_k, \quad \|\zeta\| \leq \Delta_\zeta \\
k = 1, ..., N, \ i = 1, ..., n_s, \ j = 1, ..., n_s \\
l = 1, ..., n_w, \ o = 1, ..., n_y, \ i = 1, ..., n_u
\end{cases} \tag{6.33}
$$

$$\overline{\theta}_k = \max_{\theta, v^{(i,j)}, \eta, \zeta, w, z} \theta_k$$

subject to:

$$
\begin{cases}
v_k^{(i,j)} = \sum_{m=0}^{n_b^{(i,j)}} b^{(i,j)} r_{k-m}^{(j)} - \sum_{h=1}^{n_a^{(i,j)}} a^{(i,j)} v_{k-h}^{(i,j)} \\
s_k^{(i)} = \sum_{h=1}^{n_r} v_h^{(i,h)} \\
w_t^{(l)} = \sum_{h=1}^{n_\varphi^{(l)}} \varphi_k^{(l)} \Phi_k^l(z_k) \\
\hat{y}_k^{(o)} = y_k^{(o)} + \eta_k, \quad \|\eta\| \leq \Delta_\eta \\
\hat{u}_k^{(i)} = u_k^{(i)} + \zeta_k, \quad \|\zeta\| \leq \Delta_\zeta \\
k = 1, ..., N, \ i = 1, ..., n_s, \ j = 1, ..., n_s \\
l = 1, ..., n_w, \ o = 1, ..., n_y, \ i = 1, ..., n_u
\end{cases} \tag{6.34}
$$

What about the solution of the problems we have just presented? It is not a surprise the fact that they are hardly non-convex, gradient based algorithms fail in computing a global minimum. In [10], Cerone et al. demonstrate that the *Lassere's Moment theory* and *convex LMI-based relaxation techniques* can be used, and the running intersection property can be used if we are able to obtain bilinear constraints for the EFPS.
Hammerstein, Lur'e and Wiener systems can be included in the GBSC, the same holds SISO and MIMO LTI systems and even more interesting, *general polynomial nonlinear systems in regression form are special cases of GBSC.*

References

✍ Cerone, Piga, and Regruto, *"Bounded error identification of Hammerstein systems through sparse polynomial optimization"*, 2012

[6] In order to enforce *Sparse SM Identification* you can choose for example $J(\theta) = \|\theta\|_1$. In fact it is known in literature that the minimization of the ℓ_1-norm promotes sparse solution of the identification problem. This approach is useful when $n_b^{(i,j)}$, $n_a^{(i,j)}$ and $n_\varphi^{(i)}$ are not known. The choice of an l_1-norm objective function force the solution to be of reduced complexity.

- ✍ Cerone, Piga, and Regruto *"Bounding the parameters of block-structured nonlinear feedback systems"*, 2013
- ✍ Cerone, Piga, and Regruto *"Computational load reduction in bounded error identification of Hammerstein systems"*, 2012
- ✍ Cerone, Razza, and Regruto *"A unified framework for the identification of a general class of multivariable nonlinear block-structured systems"*, 2021
- ✍ Cerone and Regruto *"Parameter bounds evaluation of Wiener models with noninvertible polynomial nonlinearities"*, 2006

Chapter 7

Set-Membership identification of continuous time systems

In this chapter we are dealing with System Identification of continuous time systems. After an introduction to the problem in which we talk about identification in general, we move to the discussion about the identification of a continuous time model from noisy data (SM-Identification). Different approaches are analyzed including model transformation and Tustin discretization.

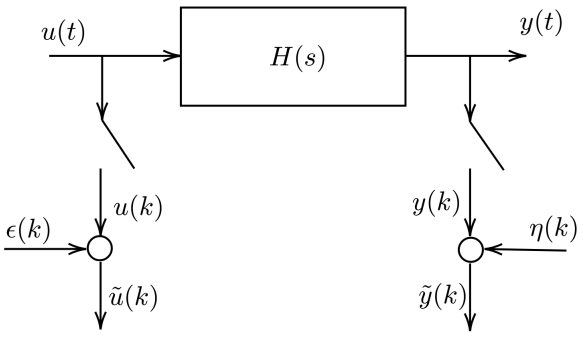
In this chapter we are considering the problem of estimating the parameters for a SISO LTI system described by a transfer function of the type:

$$H(s,\theta) = \frac{\beta_{n-1}s^{n-1} + \beta_{n-2}s^{n-2} + \cdots + \beta_1 s + \beta_0}{s^n + \alpha_{n-1}s^{n-1} + \alpha_{n-2}s^{n-2} + \cdots + \alpha_1 s + \alpha_0} \tag{7.1}$$

Where θ are the parameters of the system defined by:

$$\theta = [\alpha_0 \ldots \alpha_{n-1}, \beta_0, \ldots, \beta_{n-1}] \tag{7.2}$$

This problem must be solved by using **sampled** and **noisy** data. The experiment can be performed in the setup in Figure 7.1 in by assuming that both input and output sampled data are affected by Unknown-but-bounded (UBB) noise.



Here we assume that:

$$y(k) = y(t = kT_s) \quad u(k) = y(t = kT_s)$$

The noise samples are such that

$$|\epsilon(k)| \leq \Delta_\epsilon \quad |\eta(k)| \leq \Delta_\eta$$

Figure 7.1: SM-ID of SISO LTI CT system

7.1 Motivations

Here there are some motivations about the fact we could be interested in identifying a continuous time model. At first we can say that the parameters are not dependent on the sampling time,

while in discrete time the smaller T_s the higher is the probability to push all of the parameters toward 1[1]. Moreover continuous time model are clonser to physical descriptions of physical systems. Finally the great majority of *robust control techniques* (see $\mathcal{H}_\infty$, μ-synthesis...) are in continuous time in their original formulation.

7.2 Solution to the problem

There are several approaches for solving the problem, each one of them having advantages and disadvantages. However, our main focus in on the set-membership approach to which we devote the great majority of this chapter.

7.2.1 Indirect approach

This is the case in which, first we estimate a discrete time model from noisy sampled data $\tilde{u}(k)$ and $\tilde{y}(k)$, then we use some discretization method to transform such a model into a continuous time one. This approach is very easy BUT:

- Does not solve the problem related to the parameters that goes to 1 with the decreasing sampling time T_s.
- Inverse discretization methods do not preserve important property of the identified system, first among the other, **stability**.

Anyway, we can use for example *Least Squares* (taking care of all its disadvantages) and in MATLAB the command `d2c` can be using with the objective of obtaining a continuous time transfer function.

7.2.2 Direct approach

In order to better understand the features and the issues raised up from such a **Direct approach**, we analyze an example.

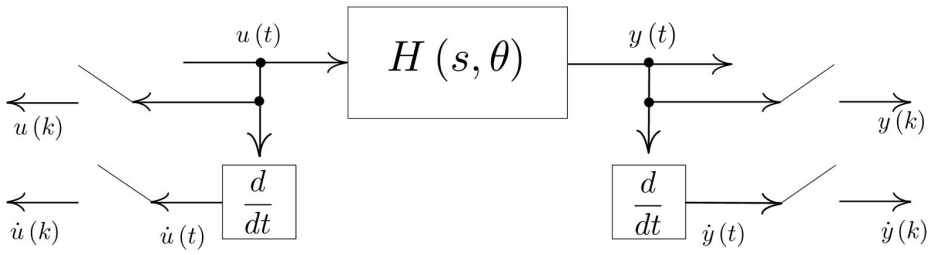


Figure 7.2: Experimental settings using derivators

Suppose you want to identify the parameter of a model whose a-priori assumptions are saying us that the shape of its transfer function is

$$H(s) = \frac{\beta_0 + \beta_1 s}{s + \alpha_0} \tag{7.4}$$

[1]The mapping between s-plane and z-plane is dictated by the **sampling transformation**

$$z = e^{sT_s} \tag{7.3}$$

Here we have that for $T_s \to 0$, $z \to 1$

By applying the definition of transfer function relating the $\mathcal{L}$-transform of the input and the output, we can write down:

$$y(s) = H(s)u(s) \iff sy(s) + \alpha_0 y(s) = \beta_1 su(s) + \beta_0 u(s) \tag{7.5}$$

We want to go back in time domain (data are sampled in time) by inverse transforming the Equation (7.5), since we assume to start with a transfer function model, the initial conditions related to the input and the output are null. Then, in time domain we have

$$\dot{y}(t) + \alpha_0 y(t) = \beta_1 \dot{u}(t) + \beta_0 u(t) \tag{7.6}$$

Note that if we sample the input and the output at $t = kT_s$ we can arrange the Equation (7.6) in a matrix form, where the coefficient matrix contains the sample of $y(t)$, $\dot{y}(t)$, $u(t)$, $\dot{u}(t)$, then we can use the Least-Squares to obtain the parameters with the assumption that the noise entering the data is modeled as a *white gaussian noise*. How can such samples be obtained? We can think to introduce some derivators in the experimental setting with the aim to sample the input/output of the system and its derivatives. However, there is a big problem: derivators are anticausal filters, though not phisically realizable!
Another idea could be the one of **estimating the derivatives** by introducing a sort of approximation in discrete time of the transfer function $H(s) = s$ representing a derivative in the Laplace domain. So we have that:

$$\dot{y}(k) \simeq D(q^{-1})\tilde{y}(k) = D(q^{-1})y(k) + D(q^{-1})\eta(k)$$

Here we have to keep in mind that the sampled data are noisy! If such noise has high frequency components, this is hugely amplified, since $D(q^{-1})$ being an approximation of s will have a shape similar to an *High pass filter*. To conclude this discussion, if we had second derivatives, even worse.

In conclusion, also the idea to estimate the samples of the derivatives must be thrown out since even if the consistency property is perfectly fulfilled and data are noiseless, we have the non-negligible problem of the high pass filtering effect introduced by $D(q^{-1})$.

7.2.3 System identification by using model transformation

A possible way for going on about this topic is replacing the derivative operator s with a low-pass filter of the type

$$\lambda(s) = \frac{1}{1 + s\tau} \tag{7.7}$$

Then, we want to estimate instead of $\dot{y}(t)$ and $\dot{u}(t)$ the following quantities based on such a type of *model transformation*:

$$y^{(1)}(t) = \mathcal{L}^{-1}\{y^{(1)}(s)\}, \quad y^{(1)}(s) = \lambda(s)y(s) \tag{7.8}$$

By inverting the expression of Equation (7.7), we obtain another expression to substitute the Laplace variable s:

$$s = \frac{1 - \lambda}{\lambda\tau}$$

Now, we obtain

$$\frac{1-\lambda}{s\tau}y(s) + \alpha_0 y(s) = \beta_1 \frac{1-\lambda}{s\tau}u(s) + \beta_0 u(s) \tag{7.9}$$

$$y(s) - \lambda y(s) + \alpha_0 \lambda \tau y(s) = \beta_1 u(s) - \beta_1 \lambda u(s) + \beta_0 \lambda \tau u(s) \tag{7.10}$$

$$y(s) = (1 - \alpha_0 \tau)\lambda y(s) + \beta_1 u(s) + (\beta_0 \tau - \beta_1)\lambda u(s) \tag{7.11}$$

At this point we have to go back to *time domain*. In particular the steps are the following:

1. Substitute λ with the Equation (7.7);
2. Compute the inverse Laplace transform $\mathcal{L}^{-1}$

Doing such steps the Equation (7.11) becomes

$$y(t) = \underbrace{(1 - \alpha_0 \tau)}_{\gamma_1} y^{(1)}(t) + \underbrace{\beta_1}_{\gamma_2} u(t) + \underbrace{(\beta_0 \tau - \beta_1)}_{\gamma_3} u^{(1)}(t) \tag{7.12}$$

where γ_1, γ_2, γ_3 are the parameters of the **transformed model**. You can arrange the Equation (7.12) in a matrix form by computing the function at time $t = kT_s$, $k = 1, 2, ...$ and by using the Least-Squares the system can be (pseudo)inverted. The samples $y^{(1)}$ and $u^{(1)}$ can be obtained in simulation without amplifying the noise as in the case of an high pass filter approximating a derivator. The last step is obtaining the original parameters θ starting from the ones of the transformed model. Since

$$\begin{cases} \gamma_1 = 1 - \alpha_0 t \\ \gamma_2 = \beta_1 \\ \gamma_3 = \beta_0 \tau - \beta_1 \end{cases} \iff \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \end{bmatrix} = \begin{bmatrix} -\tau & 0 & 0 \\ 0 & 0 & 1 \\ 0 & \tau & -1 \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \beta_0 \\ \beta_1 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = M\theta - v \tag{7.13}$$

We can invert the system in order to retrieve from γ the parameters θ

$$\theta = M^{-1}(\gamma - v) \tag{7.14}$$

7.3 Set-Membership identification

Here the objective is to obtain a model for the system using noisy data and embedding the information on the noise directlty into the problem. As usual when you want to perform System Identification, we have to collect all of a-priori and a-posteriori information.

1. A-PRIORI INFORMATION ON THE SYSTEM the system is continuous-time, described by the equation Equation (7.1);

2. A-PRIORI ASSUMPTION ON THE NOISE for sake of generality we assume that both input and output are affected by unkown but bounded noise $\epsilon(k)$ and $\eta(k)$, the bounds Δ_η and Δ_ϵ are assumed to be known.

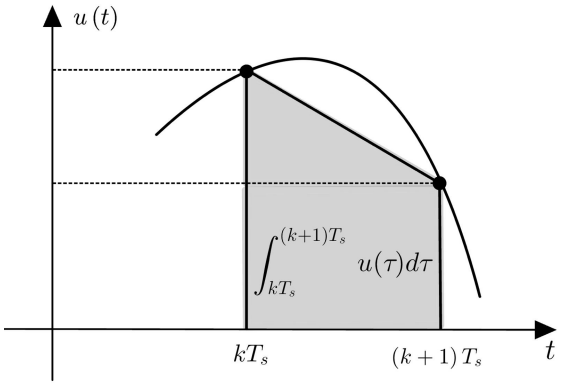
At this point two approaches can be exploited in order to going on:

1. Model transformation, this is not very used, however for further details you can consider the reference Johansson "Identification of continuous-time models", 1994 [17];

2. Tustin Discretization, for this approach the main ideas are taken from Cerone et al. "Set-membership identification of continuous-time systems through model transformation", 2022 [16]

7.3.1 Tustin discretization

The *Tustin discretization* is a **discretization method** than can be used when the function to be discretized has some regularity properties. The fundamental thing to be understand is: **what is the Tustin discretization of** $1/s$? In this way, after having retrieved such an expression a mapping between z and s using the Tustin method can be found. We know that given an integrator we have that its output can be computed as:



$$y(t) = y(t_0) + \int_{t_0}^{t} u(\tau)d\tau$$

In the case of evaluating this expression at the instant $t = kT_s$ we have that

$$y(k+1) = y(k) + \int_{kT_s}^{(k+1)T_s} u(\tau)d\tau \tag{7.15}$$

Figure 7.3: Idea behind Tustin Discretization

The second term of $y(k+1)$ is nothing but the area under the graph of $u(t)$. If we have the assumption that the signal $u(t)$ has a shape such this area can be approximated by using a trapezoid, we can write that

$$\int_{kT_s}^{(k+1)T_s} u(\tau)d\tau = \frac{(u(k+1)+u(k))T_s}{2} \tag{7.16}$$

which is the formula for computing the area of a trapezoid and $u(k)$, $u(k+1)$ are the basis, while T_s (sampling time) is the height. Then we have:

$$y(k+1) = y(k) + \frac{T_s}{2}(u(k+1)+u(k)) \tag{7.17}$$

$$y(k+1) - y(k) = \frac{T_s}{2}(u(k+1)+u(k)) \tag{7.18}$$

$$qy(k) - y(k) = \frac{T_s}{2}(qu(k)+u(k)) \tag{7.19}$$

$$(q-1)y(k) = \frac{T_s}{2}(q+1)u(k) \iff y(k) = \underbrace{\frac{T_s}{2}\frac{(q+1)}{(q-1)}}_{1/s}u(k) \tag{7.20}$$

Having the expression for $1/s$ we can find that

$$s = \frac{2}{T_s}\frac{(z-1)}{(z+1)} \tag{7.21}$$

where we have replaced the q of the backward shift operator with z. Once we have such a discretization formula, you have to replace it the continuous time expression.

7.3.2 Applying Tustin discretization to SM-ID

As usual, is very effective giving an example by which we can discover the main properties of the approach of using the Tustin Discretization, the Figure 7.4 shows the experimental settings. Suppose you have a continuous time model described by:

$$H(s) = \frac{\beta_1 s + \beta_0}{s^2 + \alpha_1 s + \alpha_0} \tag{7.22}$$

By applying Equation (7.21) the Equation (7.22) becomes a discrete time one discribed by:

$$H(z) = \frac{\beta_1 s \frac{2}{T_s}\frac{(z-1)}{(z+1)} + \beta_0}{\left(\frac{2}{T_s}\frac{(z-1)}{(z+1)}\right)^2 + \alpha_1 \frac{2}{T_s}\frac{(z-1)}{(z+1)} + \alpha_0} \tag{7.23}$$

The output $y^d(k)$ of the model $H(z)$ when the sequence $u(k)$ is applied, in general is different than the sampled output of the real system $y(k)$ if all of the assumption are not fulfilled. More clearly:

$$y(k) = y(t = kT_s),\ k = 1, 2, ... \tag{7.24}$$
$$y^d(k) = H_{\text{Tustin}}(q^{-1})u(k) \tag{7.25}$$
$$\delta(k) = y^d(k) - y(k) \tag{7.26}$$

where $\delta(k)$ denotes the gap between the two sequences for each k and so it plays the role of a **discretization error**. By assumption we assume that such a difference is bounded by a constant Δ_δ assumed to be known.
Since, we measure the output affected by noise, we can write that

$$\delta(k) = y^d(k) - \tilde{y}(k) + \eta(k) \iff \delta(k) - \eta(k) = \eta'(k) = y^d(k) - \tilde{y}(k) \tag{7.27}$$

Since we know also the bound for η we have that

$$|\eta'(k)| \leq \Delta_\eta + \Delta_\delta \tag{7.28}$$

Now we can define the feasible parameter set by putting together all of the a-priori and a-posteriori information:

$$\begin{aligned}\mathcal{D}_\theta = \big\{ & \theta \in \mathbb{R}^4 :\ y(t) = H(s,\theta)u(t) \\ & H(z) = \frac{\beta_1 s \frac{2}{T_s}\frac{(z-1)}{(z+1)} + \beta_0}{\left(\frac{2}{T_s}\frac{(z-1)}{(z+1)}\right)^2 + \alpha_1 \frac{2}{T_s}\frac{(z-1)}{(z+1)} + \alpha_0} \\ & \tilde{y}(k) = y^d(k) + \eta'(k), \quad |\eta'(k)| \leq \Delta_\eta + \Delta_\delta \\ & \tilde{u}(k) = u(k) + \epsilon(k) \quad y^d(k) = H(z)u(k) \quad k = 1, ..., N \big\}\end{aligned} \tag{7.29}$$

We know that such a set $\mathcal{D}_\theta$ must be extended, but it is the case to rewrite $H(z)$ in a more compact form, in particular if we do all the algebraic steps are needed at the end we obtain:

$$H(z) = \frac{z^2 \overbrace{[2T_s\beta_1 + T_s\beta_0]}^{b_2(\beta)} + z\overbrace{[2T_s\beta_0]}^{b_1(\beta)} + \overbrace{T_s^2\beta_0 - 2T_s\beta_1}^{b_0(\beta)}}{z^2 \underbrace{[4 + 2\alpha_1 T_s + T_s^2\alpha_0]}_{a_2(\alpha)} + z\underbrace{[2T_s^2\alpha_0 - 8]}_{a_1(\alpha)} + \underbrace{4 - 2\alpha_1 T_s + T_s^2\alpha_0}_{a_0(\alpha)}} \tag{7.30}$$

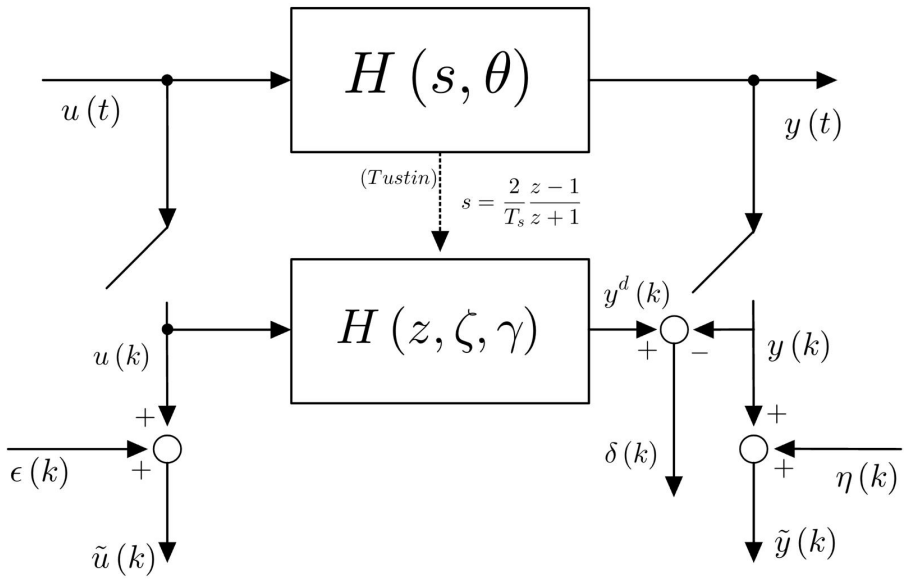


Figure 7.4: Tustin disretization experimental settings

We want that the denominator being a *monic* polynomial (the leading coefficient equal to 1)[2], so that at the end of the day we can write:

$$H(z,\zeta,\gamma)=\frac{\zeta_2 z^2+\zeta_1 z+\zeta_0}{z^2+\gamma_1 z+\gamma_0} \tag{7.31}$$

The new parameters ζ,γ are such that

$$\zeta_2=\frac{b_2(\beta)}{a_2(\alpha)} \quad \zeta_1=\frac{b_1(\beta)}{a_2(\alpha)} \quad \zeta_0=\frac{b_0(\beta)}{a_2(\alpha)} \qquad \gamma_1=\frac{a_1(\alpha)}{a_2(\alpha)} \quad \gamma_0=\frac{a_0(\alpha)}{a_2(\alpha)} \tag{7.32}$$

These new additive (slack) optimization variable are dependent on the parameters $\theta=[\alpha,\beta]^T$ and results in bilinear constraints on the optimization variables, they must be embedded in the description of the *extended feasible parameter set*:

$$\begin{aligned}\mathcal{D}_{\theta,\zeta,\gamma,\epsilon,\eta'} =&\big\{(\theta,\zeta,\gamma,\epsilon,\eta')\in\mathbb{R}^{n_\theta+5+2N}:\\ &|\eta'(k)|\le\Delta_\eta+\Delta_\delta \quad |\epsilon(k)|\le\Delta_\epsilon \; k=1,...,N\\ &\tilde{u}(k)=u(k)+\epsilon(k) \quad k=1,...,N\\ &\tilde{y}(k)-\eta'(k)+\gamma_1\tilde{y}(k-1)-\gamma_1\eta'(k-1)+\gamma_2\tilde{y}(k-2)+\gamma_2\eta'(k-2)=\\ &=\zeta_2\tilde{u}(k)-\zeta_2\epsilon(k)+\zeta_1\tilde{u}(k-1)-\zeta_1\epsilon(k-1)+\zeta_0\tilde{u}(k-2)-\zeta_0\epsilon(k-2) \quad k=n+1,...,N\\ &\underbrace{\zeta_i a_2(\alpha)=b_i(\beta) \quad i=0,1,2 \quad \gamma_i a_2(\alpha)=a_i(\alpha)}_{\text{bilinear in } \theta,\zeta,\gamma \quad i=0,1}\big\}\end{aligned} \tag{7.33}$$

The problem of computing the PUI on α,β is a POP of degree 2 (can be solved by SparsePOP). The optimization problems to be solved are

$$\underline{\theta}_i=\min_{(\theta,\zeta,\gamma,\epsilon,\eta')\in\mathcal{D}_{\theta,\zeta,\gamma,\epsilon,\eta'}}\theta_i, \quad \overline{\theta_i}=\max_{(\theta,\zeta,\gamma,\epsilon,\eta')\in\mathcal{D}_{\theta,\zeta,\gamma,\epsilon,\eta'}}\theta_i, \qquad PUI_i=[\underline{\theta}_i\;\overline{\theta}_i] \tag{7.34}$$

[2]Allowing a "no-one" leading coefficient we are allowing arbitrary rescaling of all the coefficient, achieving bad-posedness of the problem.

Remark. If we assume that $\Delta_\delta = 0$ and we use **noise free** data, then the problem of finding the PUIs is **infeasible**, since even if we are using data not being corrupted by noise we are assuming that there is bounded noise on both input and output. Moreover we are saying that we are in the ideal condition in which between one point and the other there is a **straigth line**.

Remark. Notice that the bound on the discretizationn error, must be provided as an *a-priori assumption*, however it depends on T_s and $u(t)$.

From these two remarks we can say that the information Δ_δ cannot be neglected otherwise we have infeasibility. So a common way to go beyond such a problem is to find **the minimal value for Δ_δ such that the problem has at least a solution**. We have to solve an optimization problem over the same feasible parameter set, with the only difference that the bound on the discretization error is an optimization variable.

7.3.3 Estimating Δ_δ from data

There is the possibility to estimate the bound Δ_δ on the discretization error, however there is no guarantee that we are not adding conservativeness. Anyway the way of proceeding is the following:

$$\begin{aligned}
\Delta_\delta^* = & \min_{(\Delta_\delta,\theta,\zeta,\gamma,\epsilon,\eta')\in\mathcal{D}_{\Delta_\delta,\theta,\zeta,\gamma,\epsilon,\eta'}} \Delta_\delta \\
& \text{s.t.} \\
& \tilde{y}(k) = y^d(k) + \eta'(k) \quad |\eta'(k)| \leq \Delta_\eta + \Delta_\delta \quad |\epsilon(k)| \leq \Delta_\epsilon \; k = 1, ..., N \\
& \tilde{u}(k) = u(k) + \epsilon(k) \quad k = 1, ..., N \\
& y^d(k) = H(q^{-1}, \zeta, \gamma)u(k) \quad k = n+1, ..., N \\
& \zeta_i a_2(\alpha) = b_i(\beta) \quad i = 0, 1, 2 \quad \gamma_i a_2(\alpha) = a_i(\alpha) \quad i = 0, 1
\end{aligned} \tag{7.35}$$

References

- Johansson, *"Identification of continuous-time models"*, 1994
- Cerone et al., *"Set-membership identification of continuous-time systems through model transformation"*, 2022

Chapter 8

Enforcing stability for the identified model

Everytime we perform an open-loop experiment we need BIBO stability otherwise the output of the system to be identified may diverge, so that we are not able to measure it. This is a **strong information** on the parameters of the system itself! Indeed, adding such an information in the problem allow us to get more *accurate estimate*, that is **smaller parameter uncertainty intervals** in out Set-Membership framework. The main reference for this part is Cerone, Piga, and Regruto *"Enforcing stability constraints in set-membership identification of linear dynamic systems", 2011 [7]*

8.1 Review on stability of Discrete Time LTI systems

We know that a generic discrete time LTI model can be described by means of a *transfer function* in the z-domain:

$$G_p(z) = \frac{N(z)}{D(z)} \tag{8.1}$$

Definition 8.1.1 (**BIBO Stability**)**.** $G_p(s)$ is said to be BIBO stable if and only if the response of the system is bounded for all the possible bounded input signals.

Referred to such a definition there is an important result that states:

> $G_p(z)$ is BIBO stable if and only if the poles of $G_p(s)$ (the roots of $D(z) = 0$) have all magnitude less than 1.

It is remarkable that both $N(z)$ and $D(z)$ depends on some parameter θ to be estimated. In fact:

$$G_p(z, \theta) = \frac{N(z, \theta)}{D(z, \theta)} \tag{8.2}$$

Thus, we can define a set on the parameter space which is the set of all the parameter values such that the model $G_p(z, \theta)$ is BIBO stable. Such a set can be expressed as

$$\mathcal{D}_{\text{stab}} = \{\theta \in \mathbb{R}^{n_\theta} : \ D(z, \theta) \neq 0, \forall z \in \mathbb{C}, \ |z| \geq 1\} \tag{8.3}$$

Henceforth, we can understand that **stability depends only on the parameters appearing at the denominator** which in turn are the *poles of the system*. Given the denominator of a transfer function $G_p(z)$ there a theorem (Jury Theorem) by which we can retrieve **conditions** on the parameters from which it depends, so that the resulting system (for us the system which has been identified) is BIBO stable.

8.2 Jury Theorem

The **Jury Theorem** is based on some conditions which rely on a table that is called the **Jury array**. Before exposing the theorem we give the construction of such an array.

8.2.1 Construction of the Jury array

The denominator of our transfer function is given by:

$$D(z,\theta) = 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_n z^{-n} \tag{8.4}$$

The following table is the so called **Jury array**:

$$\begin{array}{lcccccc}
\underline{\text{row}} & \underline{z^n} & \underline{z^{n-1}} & \underline{z^{n-2}} & \cdots & \underline{z^1} & \underline{z^0} \\
1 & a_n & a_{n-1} & a_{n-2} & \cdots & a_1 & 1 \\
2 & 1 & a_1 & a_2 & \cdots & a_{n-1} & a_n \\
3 & c_{n-1} & c_{n-2} & \cdots & c_1 & c_0 & \\
4 & c_0 & c_1 & \cdots & c_{n-2} & c_{n-1} & \\
5 & d_{n-2} & d_{n-3} & \cdots & d_0 & & \\
6 & d_0 & d_1 & \cdots & d_{n-2} & & \\
\vdots & \vdots & \vdots & \vdots & \vdots & & \\
2n-5 & p_3 & p_2 & p_1 & p_0 & & \\
2n-4 & p_0 & p_1 & p_2 & p_3 & & \\
2n-3 & q_2 & q_1 & q_0 & & &
\end{array} \tag{8.5}$$

We assume that the polynomial at denominator is *monic* so the leading coefficient a_0 is always equal to one. Moreover the terms c and d are defined as:

$$\begin{aligned}
c_{n-1} &= \begin{vmatrix} a_n & a_{n-1} \\ 1 & a_1 \end{vmatrix}, \quad c_{n-1} = \begin{vmatrix} a_n & a_{n-2} \\ 1 & a_2 \end{vmatrix}, \quad c_{n-j} = \begin{vmatrix} a_n & a_{n-j} \\ 1 & a_j \end{vmatrix}, \quad , \ldots , \quad c_0 = \begin{vmatrix} a_n & a_1 \\ 1 & a_n \end{vmatrix} \\
d_{n-2} &= \begin{vmatrix} c_{n-1} & c_{n-2} \\ c_0 & c_2 \end{vmatrix}, \quad d_{n-j} = \begin{vmatrix} c_{n-1} & c_{n-j} \\ c_0 & c_j \end{vmatrix}, \quad , \ldots , \quad d_0 = \begin{vmatrix} c_{n-1} & c_0 \\ c_0 & c_{n-1} \end{vmatrix}
\end{aligned} \tag{8.6}$$

Theorem 2 (Jury Theorem)**.** *The roots of the polynomial $D(z)$ belong to the* ***open unit circle*** *if and only if the following equations hold:*

1. $D(z=1) = 1 + a_1 + a_2 + \cdots + a_n > 0$
2. $(-1)^n D(z=1) = (-1)^n (1 - a_1 + a_2 - a_3 + \cdots \pm a_n) > 0$
3. $|a_n| \leq 1$
4. $|c_{n-1}| < |c_0|, \;\; |d_{n-2}| < |d_0|, \ldots, |q_2| < |q_0|$

It is more convenient that the last constraints are recasted in $c_{n-1}^2 < c_0^2, \ldots$, otherwise it is not so easy to handle such constraints. They keep their non-convex shape, but at least they are polynomial. Finally, it is remarkable that all of the elements of the Jury array are **polynomial functions of** θ.

8.3 SM-ID and Jury theorem derived constraints

Once we have obtained a strong result on the stability of discrete time systems which are dependent on a certain number of parameters, we can **enforce stability** of the identified model by:

1. Adding optimization variables representing the entries of the Jury array $c_{n-1}, ..., d_{n-2}, ..., d_0$;
2. Modify the extended feasible parameter set in order to include the relation between θ and the entries of the Jury array $(a_1, ..., a_n \rightarrow c_{n-1}, ..., c_0)$;
3. Adding the relations between different entries of the Jury Array $(c_{n-1}, ..., c_0 \rightarrow d_{n-2}, ..., d_0...)$
4. Adding the equations (1)-(4) coming from the *Jury Theorem*.

Remark. Even if we include these constraints there is no guarantee that the identified model is stable because:

1. We relax our POPs to convex SDP, and so since we are computing an outer approximation of the feasible parameter set, we may fall outside the true FPS, and finding a point (in the parameter space) not satisfying all of the constraints.
2. Even if we could describe the exact FPS when taking the central estimate, we may fall out of the set.

Then, **Why enforcing stability constraints?** They shrink the FPS and ultimately improve the accuracy of the model because the obtained PUI will be tighter.

8.4 An applied example

Given the discrete time LTI system described by

$$H(z) = \frac{\beta_1 z^{-1} + \beta_0}{1 + a_1 z^{-1} + a_2 z^{-2} + a_3 z^{-3}} \tag{8.7}$$

using the Jury theorem give necessary and sufficient conditions for which the given system is BIBO stable. The Jury array for the denominator of the given transfer function is:

$$\begin{array}{cccc} a_3 & a_2 & a_1 & 1 \\ 1 & a_1 & a_2 & a_3 \\ c_2 & c_1 & c_0 & \end{array} \tag{8.8}$$

The necessary and sufficient conditions dictated by the theorem 2 are:

$$\begin{cases} 1 + a_1 + a_2 + a_3 > 0 \\ -(1 - a_1 + a_2 - a_3) > 0 \\ |a_3| < 1 \iff -1 < a_3 < 1 \\ |c_2| < |c_0| \iff c_2^2 < c_0^2 \end{cases} \tag{8.9}$$

These equations, together with the other describing how the several additive variables and parameters of the system are related, must be embedded into the extended feasible parameter set. Each one of this additive constraints will have its own field of the structure `ineqPolySys` of `SparsePOP`.

Part II

Direct Data-Driven Control

Chapter 9

Direct Data-Driven Control design: Set-Membership approach

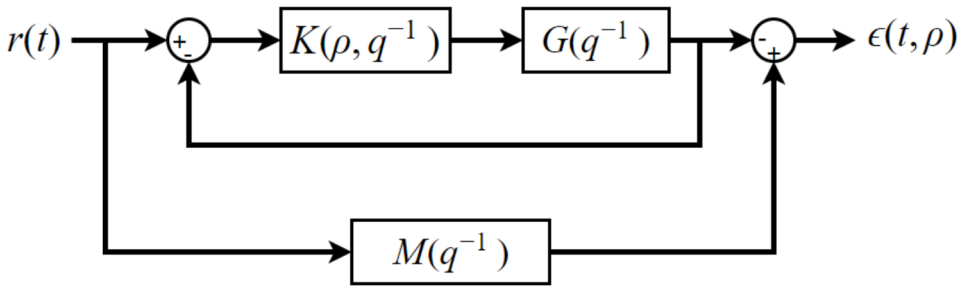


Figure 9.1: Direct Data-Driven Control (DDDC) setting

In the previous chapter we have seen how can we build a model for a dynamical system, linear/nonlinear, SISO/MIMO. For sure, you know that one of the objective of having a mathematical model/description for a given plant is to build a certain *controller* in order to modify the behaviour of the plant itself. In this chapter, going through the description of the *Direct Data-Driven control problem* (DDDC), we will see how to control a certain plant without identify it, but passing directly to the controller design using data.

9.1 Introduction

In the field of Control Theory there are mainly three approaches to design a controller for a given plant:

1. *Model-Based technique* in which we have a model for the plant to control, and the problem is solved using some control technique according to the specific problem we have to solve (requirements based, optimal control...);
2. *Undirect Data-Driven* here since we do not have a model for the plant, at first we retrieve a model for the plant to be controlled (using some SysId technique), then using the obtained model, some control technique is applied;
3. *Direct Data-Driven* Here we have that the controller is designed skipping the plant identification stage, so data are used to build a model directly for the controller.

In particular here the objective, with reference to the Figure 9.1 is to design the controller K such that the **controlled system** could match as well as possible the behaviour of the **reference model** M. In particular given $K(\rho, q^{-1})$, ρ is the *parameter vector* defined as

$$\rho = [\rho_1 \quad \rho_2 \quad \dots \quad \rho_{n_\rho}] \tag{9.1}$$

and we want that

$$T_{ry}(q^{-1}) = \frac{G(q^{-1}K(q^{-1}))}{1 + G(q^{-1}K(q^{-1}))} \approx M(q^{-1}) \tag{9.2}$$

Both K and G are assumed to be LTI dynamical systems and the description of G is not available, that is there is no model for the plant to be controlled. From now on we will assume $M = M(q^{-1})$, $G = G(q^{-1})$ and $K = K(q^{-1})$

9.2 Ideal controller K^*

Ideally we want that the Equation (9.2) could hold with the equality so that

$$M = \frac{KG}{1 + KG} \tag{9.3}$$

which is the so called **model matching problem**. We have said that G is not known, but assume for a moment that we are able to know exactly G. In this case the model matching problem has a *trivial solution* which corresponds to obtaining an **ideal controller** K^*. By doing simple algebraic manipulation we obtain

$$M = \frac{GK}{1 + GK} \iff M + MGK = GK \iff KG(1 - M) = M \iff K^* = \frac{M}{G(1 - M)} \tag{9.4}$$

Such a K^* has interesting theoretical property useful in order to go on the description of the DDDC problem, however it can be easily shown that:

1. The resulting K^* from algebraic computations can be *not physically realizable*, using such a trivial formula you are not sure that

$$\deg(D_{K^*}) \geq \deg(N_{K^*})$$

2. K^* is not guaranteed to provide internal stability of the feedback control system[1], since there can be *unstable zero-pole cancellations*.

It can be easily prooved that if we include all of the unstable zeros of $G(q^{-1})$ in the reference model $M(q^{-1})$ then K^* computed is ensuring internal stability of the FCS. But this way you have to pay something: if M (how we will see) is obtained by using some performance requirements, including such new zeros in it will result in a non accurate description of the desired I/O behaviour! We have to derive the controller using a **different approach**.

[1]Remember that an DT LTI Feedback control system is stable if:

- All of the roots of $1 + L(z)$ belongs to the open unit circle
- No unstable zero-pole cancellations occurs when $L()z$ is formed.

9.3 Formulation of the DDD control problem

We wonder here how to solve the model-matching problem when: i) $G(q^{-1})$ is not known, (ii) we can collect I/O data by performing an *open-loop* experiment on the plant to be controlled. Then we want to solve the problem (9.3), where K is **to design**, G is unknown, finally M is given and retrieved taking into account some performance requirements. Going on, you can see that from Equation (9.3) you can find a **model matching error transfer function**

$$E(q^{-1}) = M - \frac{KG}{1+KG} \tag{9.5}$$

The ideal task is to **find** K such that $E(q^{-1}) = 0$, we can define the **output matching error** by multiplying both sides of Equation (9.3) by the reference $r(k)$

$$\epsilon(\rho, q^{-1}) = \left(M - \frac{KG}{1+KG}\right) r(k) \tag{9.6}$$

From this you can see that if $T \approx M$ then $\epsilon(\rho, q^{-1}) = 0$, and

$$Mr = \frac{KG}{1+KG} r \quad \forall r(k) \tag{9.7}$$

Here the problem is that we do not have the model $G(q^{-1})$ for the plant and so we cannot find properly the controller $K(\rho, q^{-1})$! How can we go ahead? If we better analyze the Equation (9.7) we can exploit a useful insight, in fact

$$Mr - MKGr = KGr \iff (1-M)KGr = Mr \iff KGr(k) = \frac{M}{1-M} r(k) \tag{9.8}$$

Here we have that tber term $Gr(k)$, is by definition equal to the (noise-free) output $y(k)$ of the plant G when you apply the reference signal $r(k)$. The left hand side term is a signal obtained (by simulation) obtained by filtering $r(k)$ by using the transfer function $M/(1-M)$, we can call

$$s(k) = \frac{M}{1-M} r(k) \tag{9.9}$$

Then the Equation (9.8) can be written as:

$$K(\rho, q^{-1}) y(k) = s(k) \tag{9.10}$$

We can notice that Equation (9.10) is formulating a System Identification problem. We want to find K, and so its parameters ρ given the input $y(k)$ and the output $s(k)$, where:

- $y(k)$ are the noise free samples of the output of the plant G when the reference signal $r(k)$ is applied as input. Note that since we perform an experiment, we collect noisy data, then you must use
$$y(k) = \tilde{y}(k) - \eta(k) \tag{9.11}$$
- $s(k)$ are the samples obtained by stimulating (in simulation) the model $M/(1-M)$ with the reference signal $r(k)$.

Finally, we have

$$K(\rho, q^{-1})[\tilde{y}(k) - \eta(k)] = s(k) \tag{9.12}$$

that is an **Input-Error** Set-Membership identification problem where the noise samples are assumed to be UBB (Unknown but bounded), so that

$$|\eta(k)| \leq \Delta_\eta \tag{9.13}$$

Remark. Note that from the development of the theory the following three conditions are equivalent:

- $E(q^{-1}) = 0 \quad \forall r(t)$
- $\epsilon(\rho, q^{-1}) = 0 \quad \forall r(t)$
- $K(\rho, q^{-1})r(k) = M(1-M)^{-1}r(k)$

Remark. The first two conditions must be satisfied **for any possible signal** $r(t)$. To this aim we have to select a signal $r(t)$ to excite the plant that is as close as possible to a *White Gaussian Noise (WGN)*, indeed such a signal applied to the plant G is able to excite the system dynamics as well as any other signal!

9.4 Set-Membership approach to DDDC (SM-DDDC)

Now, we have started from the model matching problem, by using some simple algebraic manipulations we have obtained the *output-matching error*, finally we have formulated the problem of designing a controller in the form given by Equation (9.12). Here the objective is to explore how we can formulate a feasible set for the solutions of the problem, and how can be formulated the uncertainty intervals for the parameters describing the controller. It is noticeable that the Input-Error SM-ID problem is a particular case of the Error-In-Variables one! Nothing new, except for the focus we have in this chapter.

In order to solve the SM-ID problem we need to select the controller class $\mathcal{C}$. For example, this is the same to decide:

1. $\mathcal{C}$=\{class of PID controllers\}
2. $\mathcal{C}$=\{class of LTI controllers of fixed and given order n\}

How we are going to see in a minute, this framework is providing us a sistematic way to check that the choosen class $\mathcal{C}$ is suitable or not. But first, we introduce also here the equivalent of the feasible parameter set on which is based the SM approach.

9.4.1 Feasible Feasible Controller Parameter Set (FCPS)

Here the parameters that the SM procedure outputs are referred to the controller $K(\rho, q^{-1})$, so we are seeking for a **Feasible Controller Parameter Set (FCPS)**. In general this can be written as:

$$\begin{aligned}\mathcal{D}_\rho = &\{\rho \in \mathbb{R}^{n_\rho} \ : s(k) = K(\rho, q^{-1})[\tilde{y}(k) - \eta(k)] \\ &|\eta(k)| \leq \Delta_\eta \quad k = 1, ..., N\}\end{aligned}$$

In order to give an example we can assume that $\mathcal{C} = \{$class of first order LTI controllers$\}$, in this way we have a closed form for $K(\rho, q^{-1})$:

$$K(\rho, q^{-1}) = \frac{\rho_2 + \rho_3 q^{-1}}{1 + \rho_1 q^{-1}} \tag{9.14}$$

In this way the FCPS becomes ($n_\rho = 3$):

$$\begin{aligned}\mathcal{D}_\rho = &\{\rho \in \mathbb{R}^3 : s(k) + \rho_1 s(k-1) - \rho_2 \tilde{y}(k) \\ &- \rho_3 \tilde{y}(k-1) + \rho_2 \eta(k) + \rho_3 \eta(k-1) = 0 \quad k = 2, ..., N \\ &|\eta(k)| \leq \Delta_\eta \quad k = 1, ..., N\}\end{aligned} \tag{9.15}$$

We know that there is no way to eliminate the dependence on the noise samples $\eta(k)$, fir this reason we have to extend the FCPS into an *Extended Feasible Controller Parameter Set* (EFCPS) $\mathcal{D}_{\rho,\eta}$, defined by:

$$\begin{aligned}\mathcal{D}_{\rho,\eta} =&\{\rho \in \mathbb{R}^3,\ \eta \in \mathbb{R}^N : s(k) + \rho_1 s(k-1) - \rho_2 \tilde{y}(k) \\ &- \rho_3 \tilde{y}(k-1) + \rho_2 \eta(k) + \rho_3 \eta(k-1) = 0 \quad k = 2, ..., N \\ &|\eta(k)| \leq \Delta_\eta \quad k = 1, ..., N\}\end{aligned} \tag{9.16}$$

If at this stage $\mathcal{D}_{\rho,\eta}$ is **empty** than, there is no controller in the considered class $\mathcal{C}$ which solves the control problem. This is an evidence that the EFCPS is giving us a tool by which we can check if we have correctly chosen the controller class.
Moreover, it holds that $\mathcal{D}_{\rho,\theta} = \varnothing$ if and only if at least one of the POP to be solved for computing the **Controller Parameter Uncertainty Intervals (CPUIs)** has `exitflag`<0.

9.4.2 Summarizing the $K(\rho, q^{-1})$ design procedure

In order to **design a Direct Data-Driven controller (DDDC)** $K(\rho, q^{-1})$:

1. Perform an open-loop experiment by applying $r(k)$ to the plant to be controlled and collect the output $\tilde{y(k)} = y(k) + \eta(k)$, with $\eta(k)$ bounded;
2. Build the Feasible Controller Parameter set and Extended Feasible Controller Parameter Set ($\mathcal{D}_\rho$ and $\mathcal{D}_{\rho,\eta}$).
3. Compute the *Controller Parameter Uncertainty Intervals*:

$$CPUI_{\rho_i} = [\underline{\rho}_i, \overline{\rho}_i] \quad \underline{\rho}_i = \min_{\rho,\eta \in \mathcal{D}_{\rho,\eta}} \rho_i \quad \overline{\rho}_i = \max_{\rho,\eta \in \mathcal{D}_{\rho,\eta}} \rho_i \tag{9.17}$$

4. Two sitautions can occur at this point:
   a If one of the problem is infeasible than update $\mathcal{C}$ (see[2]);
   b If all the POPs are feasible (`exitflag`>0) we obtain the CPUIs for all rho_i
5. Build the controller transfer function as:

$$K_c(\rho^c, q^{-1}) = \frac{\rho^c_{n+1} + \rho^c_{n+2} q^{-1} + ... + \rho^c_{2n+1} q^{-n}}{1 + \rho^c_1 q^{-1} + \rho^c_2 q^{-2} + ... + \rho^c_n q^{-n}} \tag{9.18}$$

   where

$$\rho^c = [\rho^c_1, ..., \rho^c_{2n+1}] \iff \rho^c_i = \frac{\underline{\rho}_i + \overline{\rho}_i}{2} \tag{9.19}$$

   If K^c belongs to $\mathcal{C}$, then the ideal controller belongs to the Feasible Controller Set

Remark. It can be prooved that the central estimate ρ^c is the solution to the following optimization problem

$$\rho^c = \arg \min_{\rho \in \mathbb{R}^p} \max_{\rho' \in \mathcal{D}_\rho} \|\rho - \rho'\|_\infty \tag{9.20}$$

this is called the **Chebychev center in the ℓ_∞-norm** and it is the point of minimum distance from the farthest point in the FCPS. Then this is the point guaranteeing the *minimum uncertainty* which is possible.

[2] A possible way to proceed is the following: starting from $n = 1$, FCPS is empty? YES $\rightarrow$ increase n; NO *to* Ok, we are done and we can compute CPUIs!

9.5 Data-driven stability certification

Stability is a property of paramount importance for all the feedback control systems (FCS). By definition a FCS is stable if *all signals* remanin bounded when bounded inputs are applied, in other terms this implies that all of the transfer functions from any input to any output of the FCS is BIBO stable. Approaching the problem of DDDC we have seen that the problem can be recasted as an input error SM-ID problem.
After having chosen the dynamical order for K, the problem is solved and a **central controller** $K_c(\rho^c, q^{-1})$ is found. Now, an important question to be answered is the following:

How to certify that K^c stabilizes the unknown plant?

In the DDDC framework, stability must be certified according to the following conditions:

- The only available information of the unknown plant is the dynamical order that could be (in a conservative way) less than or equal to a given non negative integer n_P;
- A central controller K_c has been obtained at this stage;
- A set of input-output data collected on the plant is available and finally the output measurements are affected by Unknown-but-bounded (UBB) noise.

Uncertainty on data makes the problem of certifying stability harder, since the stability must be provided for **all allowed realizations** of the noise within the specified bound. Due to the presence of such uncertainty, some key ideas from *robust control framework* are used, with the only difference that our focus here is not on an uncertain description of the plant, but on an uncertain description of the controller K that has been obtained by using I/O noisy data.

9.5.1 Background concepts

$\mathcal{H}_\infty$-norm

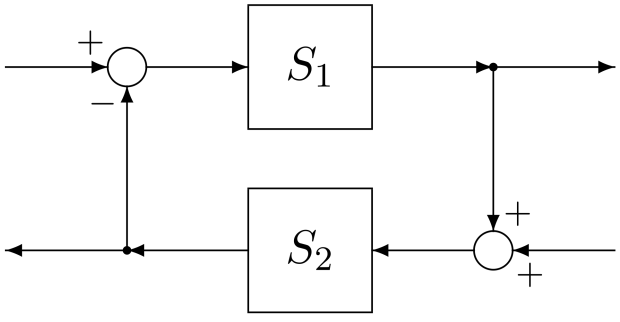
In the framework of $\mathcal{H}_\infty$ robust control, a controller G_c which fulfills robust stability and nominal performance requirements is designed. The optimization objective are given in term of $\mathcal{H}_\infty$-norm whose definition is given in the following:

Definition 9.5.1 ($\mathcal{H}_\infty$-norm)**.** Consider a transfer function $G(z)$, the $\mathcal{H}_\infty$-norm of $G(z)$ is defined as

$$\|G(z)\|_\infty \doteq \sup_{\omega\in[0,2\pi]} |G(e^{i\omega})| \tag{9.21}$$

Roughly speaking this is the maximum amplification of the G input's energy.

Small-gain theorem



Theorem 3 (Small-Gain Theorem)**.** *Suppose S_1 and S_2 are stable. Then, the interconnected system shown in the figure is well-posed and internally stable if and only if*

$$\|S_1 S_2\|_\infty < 1 \tag{9.22}$$

See Part III, Chapter 13 for an example.

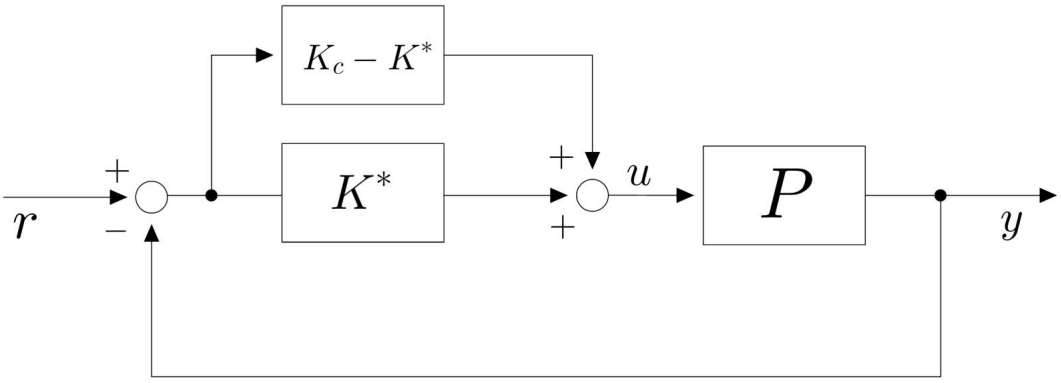


Figure 9.2: Uncertain description of the controller

9.5.2 Small-gain theorem in DDDC

Since the plant is unknown, any attempt to describe the uncertainty affecting the plant leads to an **indirect data-driven control** problem. As we gave anticipated we describe the uncertainty as if it was on the controller and in particular: the *ideal controller* $K*$ plays the role of the nominal plant, while the uncertainty is given by the *distance between the actual and ideal controller*.
In this context the *Small-gain theorem* can be applied where the subsystems S_1 and S_2 are defined as:

$$S_1 = K_c - K^*, \quad S_2 = \frac{-P}{1 + K^* P} \tag{9.23}$$

The following important theorem is given (whose proof can be retrieved from [22]):

Theorem 4. *[Van Heusden, Karimi, and Bonvin, 2011)] Let*

$$\Delta(z) = M(z) - K_c(z)P(z)(1 - M(z)) \tag{9.24}$$

the controller K_c stabilizes the plant P if

1. *The ideal controller $K^* = \frac{M}{P(1-M)}$ stabilizes the plant;*
2. *$\Delta(z)$ is stable;*
3. *$\|\Delta(z)\|_\infty < 1$*

At this point, we can say that: (i) Condition 1 is satisfied if M is stable and no unstable cancellations occur; (ii) Condition 2 is satisfied if both P and M are stable. But, **How to compute $\|\Delta(z)\|_\infty$ without using the plant and given noisy data?** The following section is aimed to answer to this question.

9.5.3 Set-Membership estimation of $\|\Delta(z)\|_\infty$

A three-steps procedure can be used, in order to obtain frequency by frequency a worst-case estimate on $|\Delta(e^{i\omega})|$.

First step: a-priori information on Δ

The system $\Delta(z)$ can be generally defined as

$$\Delta(q^{-1}) = \frac{\gamma_0 + \gamma_1 q^{-1} + \cdots + \gamma_{n_\Delta} q^{-n_\Delta}}{1 + \delta_1 q^{-1} + \cdots + \delta_{n_\Delta} q^{-n_\Delta}} \tag{9.25}$$

from the definition of Δ we can say that $n_\Delta \leq n_K + n_P + n_M$, where K, P, M are respectively the central controller, the plant and the reference model. Moreover using Equation (9.24), and multiplying both sides by $r(k)$ you can get rid of the unknown plant

$$\Delta(q^{-1})r(k) = M(q^{-1})r(k) - K_c(q^{-1})(1 - M(q^{-1}))\overbrace{P(q^{-1})r(k)}^{y(k)=\tilde{y}(k)-\eta(k)} \tag{9.26}$$

$$\Delta(q^{-1})r(k) = M(q^{-1})r(k) - K_c(q^{-1})(1 - M(q^{-1}))[\tilde{y}(k) - \eta(k)] = \tag{9.27}$$

$$= Mr(k) - K_c(1 - M)\tilde{y(k)} + K_c(1 - M)\eta(k) = \tag{9.28}$$

$$= z(k) + F(q^{-1})\eta(k) \tag{9.29}$$

Here the samples of $z(k)$ can be obtained by simulation, while $F(q^{-1})$ is completely known.

Second step: worst-case norm estimation problem

For a fixed frequency ω a worst-case norm estimation can be obtained as:

$$\begin{aligned} &\max_{\Delta,\eta} |\Delta(e^{i\omega})| \\ &\text{s.t.} \\ &\Delta(q^{-1})r(k) = z(k) + F(q^{-1})\eta(k) \\ &|\eta(k)| \leq \Delta_\eta \end{aligned} \tag{9.30}$$

Third step: obtaining a POP

With a non-negligiblle effort the problem in eq. (9.30) can be recasted into a POP. At first we can note that $\Delta(e^{i\omega})$ is nothing but a complex number that can be written as

$$\Delta(e^{i\omega}) = a + ib \tag{9.31}$$

we are interested in the magnitude of such a number, that is:

$$|\Delta(e^{i\omega})| = \sqrt{a^2 + b^2} \iff \exists t : t = |\Delta(e^{i\omega})|,\ t^2 = a^2 + b^2 \tag{9.32}$$

The powers of the numbers $e^{ik\omega}$, $k = 0, ..., n_\Delta$ can be written in cartesian form as:

$$\begin{bmatrix} e^{in_\Delta\omega} \\ \vdots \\ e^{i2\omega} \\ e^{i\omega} \\ e^{0} \end{bmatrix} = \begin{bmatrix} \rho_0 \\ \rho_1 \\ \rho_2 \\ \vdots \\ \rho_{n_\Delta} \end{bmatrix} + i \begin{bmatrix} \sigma_0 \\ \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_{n_\Delta} \end{bmatrix} \tag{9.33}$$

As a **function of the parameters** γ, δ we can evaluate $\Delta(e^{i\omega})$ and estabilish a relation between a, b, γ, δ as:

$$\Delta(z = e^{i\omega}) = \frac{\gamma_0 e^{in_\Delta\omega} + \gamma_1 e^{i(n_\Delta-1)\omega} + \cdots + \gamma_{n_\Delta-1} e^{i\omega} + \gamma_{n_\Delta}}{\delta_0 e^{in_\Delta\omega} + \delta_1 e^{i(n_\Delta-1)\omega} + \cdots + \delta_{n_\Delta-1} e^{i\omega} + \delta_{n_\Delta}} = a + ib \tag{9.34}$$

$$\sum_{j=0}^{n_\Delta} \gamma_j e^{i(n_\Delta-j)\omega} = (a + ib) \sum_{j=0}^{n_\Delta} \delta_j e^{i(n_\Delta-j)\omega} \tag{9.35}$$

We have assumed that $\delta_0 = 1$ in order to simplify the notation. Replacing the cartesian form of the complex numbers $e^{i(n_\Delta - j)\omega}$ we obtain:

$$\sum_{j=0}^{n_\Delta} \gamma_j(\rho_j + \sigma_j) = (a + ib) \sum_{j=0}^{n_\Delta} \delta_j(\rho_j + \sigma_j) \tag{9.36}$$

By separating real and imaginary parts we obtain:

$$\sum_{j=0}^{n_\Delta} \gamma_j\rho_j + i\gamma_j\sigma_j = \sum_{j=0}^{n_\Delta} a\delta_j\rho_j + ia\delta_j\sigma_j + ib\delta_j\rho_j - b\delta_j\sigma_j \tag{9.37}$$

The two sides are equal if and only if the real parts are equal each other, the same for the imaginary parts, then two equations are obtained:

$$\sum_{j=0}^{n_\Delta} \gamma_j\rho_j = \sum_{j=0}^{n_\Delta} a\delta_j\rho_j - b\delta_j\sigma_j \quad \text{Real parts} \tag{9.38}$$

$$\sum_{j=0}^{n_\Delta} \gamma_j\sigma_j = \sum_{j=0}^{n_\Delta} a\delta_j\sigma_j + b\delta_j\rho_j \quad \text{Imaginary parts} \tag{9.39}$$

In red the optimization variables, σ_j and ρ_j are known numbers. Next, γ, δ are related to the data and noise samples according to:

$$\Delta(q^{-1})r(k) = z(k) + F(q^{-1})\eta(k)$$

In the following we are using r_k instead of $r(k)$ to ease the notation. Then, by using the definition of Δ and being n_F and α_j^F, β_j^F, respectively the number of parameters and the parameters of $F(q^{-1})$, we can obtain:

$$\frac{\sum_{j=0}^{n_\Delta} \gamma_j q^{-j}}{\sum_{j=0}^{n_\Delta} \delta_j q^{-j}} r_k = z_k + \frac{\sum_{j=0}^{n_F} \beta_j^F q^{-j}}{\sum_{j=0}^{n_F} \alpha_j^F q^{-j}} \eta_k \qquad \text{Removing the denominators...} \tag{9.40}$$

$$\left(\sum_{j=0}^{n_F} \alpha_j^F q^{-j}\right)\left(\sum_{j=0}^{n_\Delta} \gamma_j q^{-j}\right) r_k = \left(\sum_{j=0}^{n_F} \alpha_j^F q^{-j}\right)\left(\sum_{j=0}^{n_\Delta} \delta_j q^{-j}\right) r_k + \left(\sum_{j=0}^{n_F} \beta_j^F q^{-j}\right)\left(\sum_{j=0}^{n_\Delta} \delta_j q^{-j}\right)\eta_k \tag{9.41}$$

$$\left(\sum_{j=0}^{n_F}\sum_{h=0}^{n_\Delta} \alpha_j^F \gamma_h q^{-j-h}\right) r_k = \left(\sum_{j=0}^{n_F}\sum_{h=0}^{n_\Delta} \alpha_j^F \delta_h q^{-j-h}\right) r_k + \sum_{j=0}^{n_F}\sum_{h=0}^{n_\Delta} \beta_j^F \delta_h q^{-j-h}\eta_k \tag{9.42}$$

$$\sum_{j=0}^{n_F}\sum_{h=0}^{n_\Delta} \alpha_j^F \gamma_h \; r(k-j-h) = \sum_{j=0}^{n_F}\sum_{h=0}^{n_\Delta} \alpha_j^F \delta_h \; r(k-j-h) + \sum_{j=0}^{n_F}\sum_{h=0}^{n_\Delta} \beta_j^F \delta_h \; \eta(k-j-h) \tag{9.43}$$

Summing up... (*Don't loose ourself!*)

A lot of boring computations have been done, in order to fix the ideas and clarify the steps, let us summarizing what we have done keeping in mind that our main focus was to obtain a POP starting from the problem in Equation (9.30):

1. A generic complex number can be described by Equation (9.31), and its norm is given by Equation (9.32), this leads to the definition of the slack variables a, b, t for real and imaginary part and for the magnitude of $\Delta(e^{i\omega})$;

2. The description of Δ has been provided in Equation (9.25) in function of its parameters (γ, δ), exploiting the relationship $q \to z$, and evaluating it in $e^{i\omega}$ we have obtained the Equation (9.35) which relates a, b, γ, δ. In order to complete this first block of transformation, equations eq. (9.38) and eq. (9.39) have been obtained;

3. Exploiting the expression of $\Delta(q^{-1})$ in Equation (9.25) and using the final step eq. (9.29) we can obtain a relation between γ, δ and the collected data $\{r_k, \tilde{y}_k\}$;

4. This leads to the equation Equation (9.40) in which we have put everything together and in order to unify the notation we have expressed also $F(q^{-1})$ in function of its parameters (α^F, β^F);

5. Multiplying both sides for the denominators of a term and the other the expression in eq. (9.41) has been obtained. To collect all the terms involving the backward-shift operator nested summations are used in Equation (9.42) step;

6. The backward-shift operator property has been applied in order to obtain the Equation (9.43), which relates the parameters of the $\Delta(q^{-1})$ with the experimentally collected data. To conclude note that all of the parameters of $\mathrm{F}(q^{-1})$ can be easily obtained since both K_c and M are also known at this stage.

We are ready! Overall, the optimization problem is recast to:

$$
\begin{aligned}
&\max_{t,a,b\in\mathbb{R},\gamma\in\mathbb{R}^{n_\Delta+1},\delta\in\mathbb{R}^{n_\Delta},\eta\in\mathbb{R}^{N}} t \\
&\text{s.t.} \\
&t^2 = a^2 + b^2 \\
&\sum_{j=0}^{n_\Delta} \gamma_j \rho_j = \sum_{j=0}^{n_\Delta} a\delta_j \rho_j - b\delta_j \sigma_j \\
&\sum_{j=0}^{n_\Delta} \gamma_j \sigma_j = \sum_{j=0}^{n_\Delta} a\delta_j \sigma_j + b\delta_j \rho_j \\
&\sum_{j=0}^{n_F} \sum_{h=0}^{n_\Delta} \alpha_j^F \gamma_h \; r_{k-j-h} = \\
&= \sum_{j=0}^{n_F} \sum_{h=0}^{n_\Delta} \alpha_j^F \delta_h \; r_{k-j-h} + \sum_{j=0}^{n_F} \sum_{h=0}^{n_\Delta} \beta_j^F \delta_h \; \eta_{k-j-h} \\
&|\eta(k)| \leq \Delta_\eta
\end{aligned}
\tag{9.44}
$$

By solving this problem by using convex SDP relaxation a bound on the true $|\Delta(e^{i\omega})|$ can be found. Theoretically since at the end we have to find an $\mathcal{H}_\infty$-norm, the problem (9.44) must be solved for all $\omega \in [0, 2\pi]$. Alternatively, to simplify an already complicated problem, we can introduce a Lipshitz continuity assumption on $|\Delta(e^{i\omega})|$, assuming that it does not increase or decay faster than a certain $L_b = 100\mathrm{dB/dec}$ or $L_b = 200\mathrm{dB/dec}$. Finally, after having computed the bound on a certain number of frequencies, you take the max, such a value gives you an estimate of the searched quantity $\|\Delta(z)\|_\infty$. For more details you can refer to the paper [1], in which you can find these results and others in the MIMO case.

To conclude this discussion, *What if the Condition 3 of the Theorem 4 is not satisfied?* A possible way to solve this problem is to collect a larger amount of experimental data so that

the size of the FCPS is reduced. In this way the distance between the central estimate and the ideal controller becomes smaller, and in turn, $\|\Delta\|_\infty$ decreases.

References

- ✍ Cerone, Regruto, and Abuabiah, *"Direct data-driven control design through set-membership errors-in-variables identification techniques"*, 2017
- ✍ Abuabiah et al., *"A non-iterative approach to direct data-driven control design of MIMO LTI systems"*, 2023
- ✍ Van Heusden, Karimi, and Bonvin, *"Data-driven model reference control with asymptotically guaranteed stability"*, 2011

Chapter 10

Model selection for Direct data-driven control through $\mathcal{H}_\infty$ techniques

Here the objective is to describe a systematic approach by which a *Reference Model* $M(q^{-1})$, that can be used in the framework of DDDC, can be obtained given suitable *quantitative performance requirements*. In the literature (see Campestrini et al. [2] and Silva et al. [21]) several valid guidelines are provided about this topic, however they do not explicitly take into account for performance requirements.

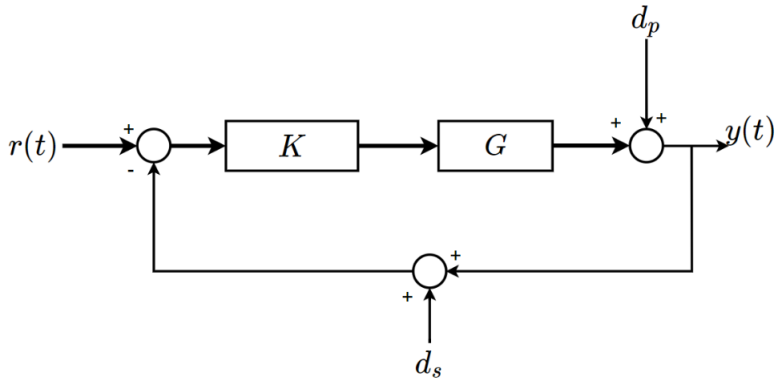


Figure 10.1: General FCS with disturbance and reference

In general, we want to find a reference model M that is able to take into account both tracking and disturbance/noise rejection.
Standard approaches which address the problem of designing a reference model, are aimed to impose a certain behaviour to the complementary sensitivity function $T(q^{-1})$ of the system in Figure 10.1. For example:

- Prototype I Order System the step response can be obtain, in the frequency domain, as

$$M(s) = \frac{1}{1 + \frac{s}{p}} \tag{10.1}$$

  the time constant $1/p$ is the only degree of freedom which is used in order to obtain faster/slower rise time, in this case no oscillations and overshoot occur.

- Prototype II Order System We aim at building a reference model M as follows:

$$M(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{10.2}$$

here according to the parameter ζ different damping properties can be obtained (oscillations in the step response), the time constant $1/\zeta\omega_n$ is tuned in order to obtain certain convergence properties.

It seems that all the needed performances can be achieved by properly tuning the parameters of such prototype models. However, it is remarkable that the input/output behaviour of the feedback control system in Figure 10.1 is given by:

$$y = Tr + Sd_p + Td_s \tag{10.3}$$

Even if the function $S(q^{-1}) = 1 - T(q^{-1})$ directly depends on the complementary sensitivity function whose shape can be decided using Equation (10.1) and Equation (10.2), it is not said at all that it enjoys good attenuation properties with respect to the disturbance $d_p(t)$ acting on the plant!
The **proposed approach** is to design $M(q^{-1})$ by solving a fictitious control design problem accounting for all the quantitative performance requirements involved in the considered control problem.

10.1 Model reference design: the $\mathcal{H}_\infty$ approach

Here we consider the problem of designing $M(q^{-1})$ by solving a fictitious control problem in the framework of the $\mathcal{H}_\infty$ optimization (see Cerone et al. [3]). Common classes of **quantitative performance requirements** are:

- steady-state response to polynomial reference inputs
- steady-state response to polynomial disturbance d_p
- steady-state response to measurement disturbance d_s
- transient step response requirements in term of: rise time t_r, settling time $t_{s,\alpha\%}$ and overshoot $\hat{s}$.

It is well known that in the context of $\mathcal{H}_\infty$ control, performance requirements can be translated into frequency domain constraints on a **weighted** $\mathcal{H}_\infty$-norm on the sensitivity function (S) and complementary sensitivity function (T):

$$\|W_T(j\omega)T(j\omega)\|_\infty \leq 1 \tag{10.4}$$
$$\|W_S(j\omega)S(j\omega)\|_\infty \leq 1 \tag{10.5}$$

10.1.1 Case 1: stable and minimum phase plant

When the unknown plant is stable and minimum phase the problem of finding a reference model $M(s)$ is given by:

$$M(s) = \tilde{T}(s) = \frac{\tilde{K}(s)\tilde{G}(s)}{1 + \tilde{K}(s)\tilde{G}(s)} \tag{10.6}$$

where $\tilde{K}(s)$ is the controller obtained by solving the optimization problem:

$$\tilde{K}(s) = \arg \min_{\tilde{K} \in \tilde{K}(s)^{stab}} \|T_{wz}(s)\|_\infty \tag{10.7}$$

where T_{wz} is the closed loop transfer function between the input w and the output z, and W_S and W_T are rational transfer function designed taking into account frequency domain constraints imposed by performance requirements, that is

$$T_{wz}(s) = \begin{bmatrix} W_T\tilde{T} \\ W_S\tilde{S} \end{bmatrix} \tag{10.8}$$

the block-diagram of the general control configuration we are referring to is the one depicted in Figure 10.2 which is the generalized plant for *nominal performance requirements*.

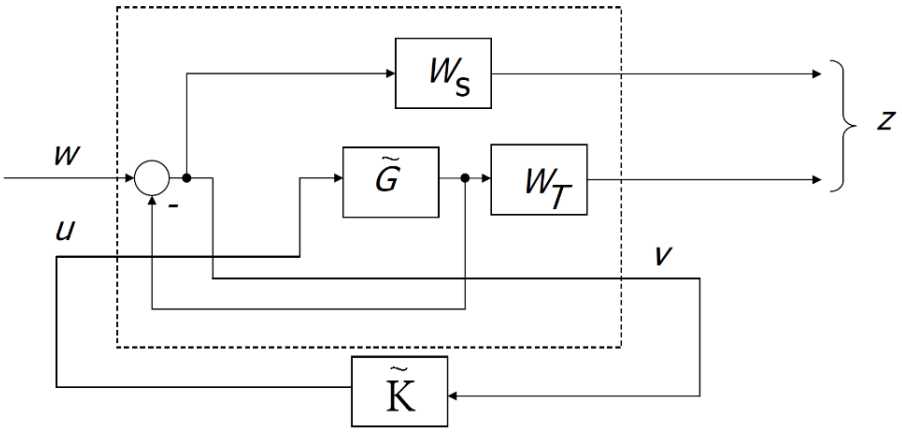


Figure 10.2: Generalized plant for nominal performances

Remark. The controller $\tilde{K}$ is instrumental to the computation of $\tilde{T}$ and then the reference model M. In other words the controller $\tilde{K}$ does not solve the problem for the actual plant which is unknown.

10.1.2 Case 2: stablle non-minimum phase systems

Definition 10.1.1 (**Non-minimum phase system**)**.** A non-minimum phase system is a dynamic system whose transfer function has one or more zeros in the right-half of the s-plane (for continuous-time systems) or outside the unit circle (for discrete-time systems). These zeros lead to certain undesirable characteristics, particularly in transient response and stability.

Step-response of non-minimum phase exhibit *undershoot*, in particular it is a behaviour of the transient response of a system where the output initially moves in the direction opposite to the desired final value before eventually settling at the target.

In this case, the reference model can be designed using the $\mathcal{H}_\infty$ approach, but instead of selecting $\tilde{G} = 1$ as fictitious plant, we include all the NMP zeros of the plant a number of poles equals or greater than the NMP ones. The additional poles are chosen in arbitrary way[1] under the condition that they are stable, moreover if the system type is greater or equal than one a suitable number of poles at the origin can be included in $\tilde{G}$.

[1]It is well-known that stable poles do not induce any performance limitation in the feedback control system.

10.1.3 Last step: Discretization of $M(s)$

Since in the DDDC based on the SM approach, we use a discrete-time description, the reference model is supposed to be discretized using a suitable sampling time T_s, obtaining then the reference model $M(q^{-1})$

10.2 Simulation example and discussion

In this example, we use the proposed approach to tune a controller for a SISO NMP plant. A numerical comparison with the standard approach is presented. Here we assume that the plant is a NMP system described by the following transfer function

$$G(s) = \frac{(s + 9.925)(s - 1.818)}{(s + 12.04)(s + 2.231)} \tag{10.9}$$

the disturbances are

$$\begin{aligned} d_p(t) &= a_p \sin(\omega_p t),\ |a_p| \leq 2 \cdot 10^{-2},\ \omega_p \leq 0.02\text{rad s}^{-1} \\ d_s(t) &= a_s \sin(\omega_s t),\ |a_s| \leq 10^{-1},\ \omega_s \leq 40 \text{ rad s}^{-1} \end{aligned} \tag{10.10}$$

The Table 10.1 reports the time domain specifications and the performances achieved with the contrrolled systems obtained with the two approaches (SMRC) and ($\mathcal{H}_\infty$RMC).

Performance Specifications	**Upper Bound Value**	**SRMC**	H_∞**RMC**
Steady-state output error when the reference is a ramp	0.5	**0.7**	0.495
Steady-state output error in the presence of d_p	$6 \cdot 10^{-4}$	$2.84 \cdot 10^{-4}$	$2.22 \cdot 10^{-4}$
Steady-state output error in the presence of d_s	$8 \cdot 10^{-3}$	**0.0196**	$6.83 \cdot 10^{-3}$
Step response overshoot	11%	**11.5%**	9.77%
Rise time	2 (s)	0.939 (s)	0.988 (s)
Settling time	8 (s)	6.36 (s)	7.33 (s)

Table 10.1: Performance Comparison

It is remarkable that even if the two step-responses are quite similar, the controlled system designed by using the standard approach does not fulfill all the requirements, on the contrary the proposed approach fulfills all of the performance requirements.

References

- Silva, Bazanella, and Campestrini, *"On the choice of an appropriate reference model for control of multivariable plants"*, 2018
- Campestrini et al., *"Unbiased MIMO VRFT with application to process control"*, 2016
- Cerone, Abuabiah, and Regruto, *"An H_∞ method to design the reference model in direct data-driven control design"*, 2020

Chapter 11

Direct-Data Driven Control in presence of Non-Minimum Phase plant dynamics[1]

Designing a controller for a **non-minimum phase system** is challenging since, no matter what is the used approach for designing it, there will be very commonly unstable zero-pole cancellations. Moreover, in the framework of Set-Membership DDDC, the plant is unknown and no estimate of the NMP zeros is done. The approach proposed in Cerone, Regruto, and Abuabiah [13] deals with a **two-stage systematic procedure** in order to detect and manage the presence of non-minimum phase dynamics in the plant to be controlled.

11.1 Two stage procedure for DDDC of NMP plant

Before starting, it is crucial to give the definition of **prospective plant** G^*:

Definition 11.1.1 (**Prospective plant**)**.** Given a reference model with transfer function M and a designed controller with transfer function K, the *prospective plant* G^* is defined as

$$G^* = \frac{M}{K(1-M)} \tag{11.1}$$

moreover given the designed controller K, the prospective plant G^* is the one giving a closed-loop transfer function that perfectly match the reference model in M.

Using the notion of *prospective plant* G^* we have just defined the following procedure allows the user to detect if the designed controller is leading to unstable zero-pole cancellations.

11.1.1 Stage #1: Controller design and detection of unstable zero-pole cancellations

Given some collected data $\{r, \tilde{y}\}_k$ and a reference model $M(q^{-1})$ a controller $K(q^{-1}, \rho)$ is designed following the procedure described in Chapter 9. At this point, there are two cases:

1. Stable $K(\rho)$ The controller can be realized, since no unstable cancellations can occur! (The controller is stable).
2. Unstable $K(\rho)$ You have to reject the obtained $K(\rho)$ and adopt one of the following two approaches.

[1] This topic is not included in the course topics. It is an additional chapter, to extend the subject of Direct Data-Driven control.

1st approach: Enforcing stability for $K(\rho)$

Using the notions presented in Chapter 8, by adding to the FCPS a proper number of constraints. Such an approach is a conservative one.

2nd approach: go to Stage #2

11.1.2 Second stage: M modification and re-design of K

All the unstable poles of the designed controller $K(\rho)$ close to the NMP zeros must be added in a suitable way to the reference model M, if the reference model is designed according to the $\mathcal{H}_\infty$ fictitious control problem, you have to put such poles as NMP zeros of the *fictitious plant* $\tilde{G}$, together with a number of poles equals or greater than the unstable ones.

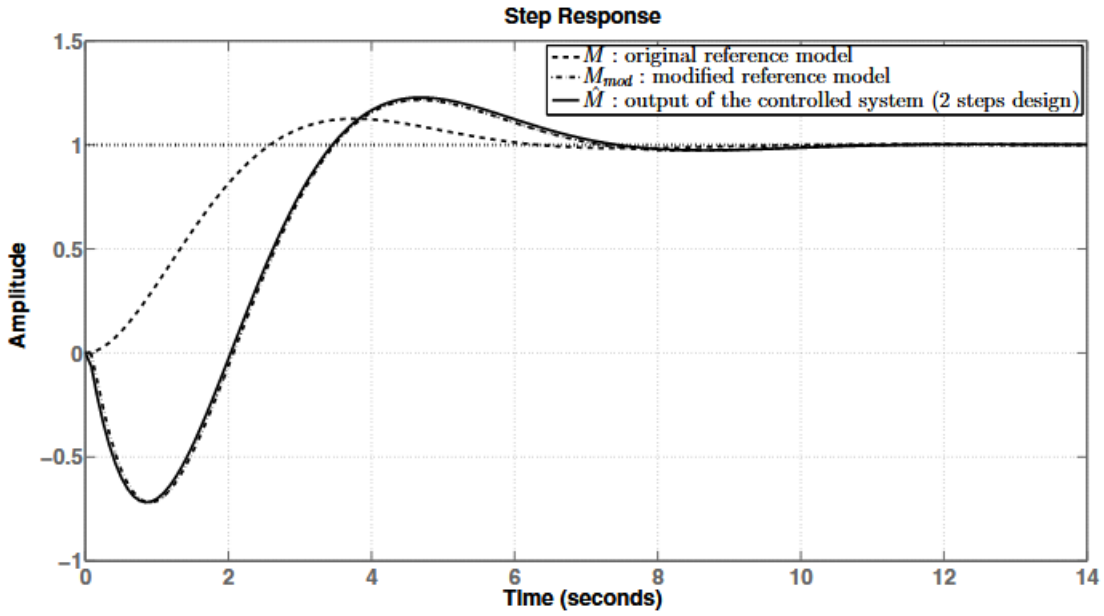


Figure 11.1: Comparison of step responses: designed feedback control system (2 step-procedure) (solid), the original reference model(dashed), modified reference model (dashed-dotted)

References

✍ Cerone, Regruto, and Abuabiah, *"A set-membership approach to direct data-driven control design for SISO non-minimum phase plants"*, 2017

Part III

Appendix

Chapter 12

Lab 1: Least-Squares parameter estimation (Solution)

In this problem we assume that the plant is a continuous time LTI dynamical system assumed to be exactly described by the following transfer function:

$$G_p(s) = \frac{100}{s^2 + 1.2s + 1}$$

```
G=100/(s^2+1.2*s+1);
```

Assuming that the input-output data have been collected with a sample time of Ts = 1s, compute a discrete-time model for the plant as follows (see help of the Matlab command `c2d`): $G_d(z) =$ `c2d`$(G_p, 1,$ 'zoh'):

```
Gz=c2d(G,1,'zoh');
```

Use the obtained discrete-time model to generate input-output data by applying to the system a randomsequence of N samples and amplitude 1 as input (see commands rand and command `lsim`). Collect the input sequence in the array u and the output sequence in the array w.

```
n=2;
k=n+1;                  %Point where I have to start
H=1e5+n;                %I need at least (3n+1) samples

u=rand(H, 1);
%noisy-data (Equation Error)
w_nf=lsim(Gz,u);
```

12.1 Noise-Free experiment

Using the array u and w, build matrix A and array b required for the computation of the least squares estimate of the discrete-time model parameters, according to the theory and examples about least square estimation presented in the classroom.

```
[numGz, denGz]=tfdata(Gz,'v');
th_true = [denGz(2:end) numGz]'

A = [-w_nf(2:H-1) -w_nf(1:(H-2)) u(3:H) u(2:(H-1)) u(1:(H-2))
   ];
```

```
th_est_noise_free = A\w_nf(k:H)
```

12.2 Equation-Error setting

Repeat the exercise (for different values of N) by adding a random *equation error* e while simulating the data.

```
D=tf([0 0 1],denGz,1);
e = 5*randn(H,1);
w_nEE = lsim(Gz,u)+lsim(D,e);
A = [-w_nEE(2:H-1) -w_nEE(1:(H-2)) u(3:H) u(2:(H-1)) u(1:(H
   -2))];
th_est_noiseEE = A\w_nEE(k:H)
```

12.3 Output-Error setting

Repeat the exercise (for different values of N) by adding a random *output measurement error* η.

```
%simulation in case of Output-Error(OE) setting
std=5;
eta=std*randn(H,1);
w_nOE=lsim(Gz,u)+eta;

A = [-w_nOE(2:H-1) -w_nOE(1:(H-2)) u(3:H) u(2:(H-1)) u(1:(H
   -2))];
th_est_noiseOE = A\w_nOE(k:H)
```

Chapter 13

(Example) Small-Gain theorem

Consider an unknown plant P. Let P_n be a nominal description of the plant and W_u a description of its uncertainty. Assume for example that a *multiplicative uncertainty model set* is used, so that P belongs to the following set:

$$M_m = \{P = P_n(1 + W_u(z)\Delta(z)),\ \|\Delta\|_\infty < 1\} \tag{13.1}$$

When you put together the controller K and the plant P with its (unstructured) uncertain description, you obtain a block diagram like the following:

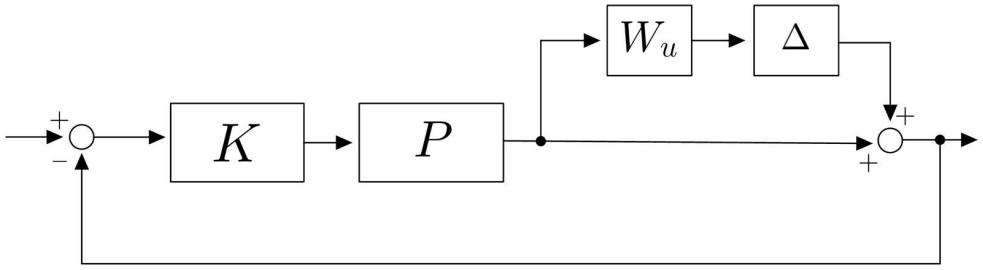


Figure 13.1: Uncertain FCS structure

If we use the *small-gain theorem*, we can assume that

$$S_1 = \Delta \tag{13.2}$$

$$S_2 = \frac{-KP_nW_u}{1 + KP_n} = W_uT_n \tag{13.3}$$

where Δ is stable by assumption, W_u is also stable by assumption while T_n is stable if K stabilizes P_n (nominal stability fulfilled). Then, S_1 and S_2 are stable by assumption. A condition on robust stability can be found, if the small gain theorem is applied:

$$\|S_1S_2\|_\infty = \left\|\Delta\frac{-KP_nW_u}{1 + KP_n}\right\|_\infty \leq \quad \text{(sub-multiplicativity)} \tag{13.4}$$

$$\leq \|\Delta\|_\infty \cdot \left\|\frac{-KP_nW_u}{1 + KP_n}\right\|_\infty \leq \left\|\frac{-KP_nW_u}{1 + KP_n}\right\|_\infty < 1 \iff \tag{13.5}$$

$$\iff \|W_uT_n\|_\infty < 1 \tag{13.6}$$

Then robust stability (stability for all the plants in M_m) is guaranteed if and only if

$$\|W_uT_n\|_\infty < 1 \tag{13.7}$$

Chapter 14

Neural Networks for Nonlinear System Identification

In **Part I** we have seen the Set-Membership approach for black-box identification of a dynamical system (linear, nonlinear, SISO or MIMO). In the literature there are other approaches which uses machine learning models such as *Fully connected neural networks* and *Recurrent Neural Networks*. In this chapter we will briefly introduce the topic of Neural Networks, and we will give some ideas on what are the models that can be used in the context of **nonlinear system identification**.

We have seen the SM-ID of nonlinear systems in very particular cases when the nonlinearities are block-structured in the overall system. This is a strong information on the system itself which, in turn, provides useful insights to face the problem in the framework of Set-Membership identification. However, there are many fields where such assumptions cannot be done since we are dealing with complex MIMO nonlinear systems! **Neural Networks** (NN) in this case are useful, since they have some nice properties (see *Universal approximation theorem*).
Here our aim is to identify a *MIMO Discrete Time dynamical system* of the kind:

$$y_t = \mathcal{S}(y_{t-1}, ..., y_{t-n}, u_t, ..., u_{t-n}) \tag{14.1}$$

where $\mathcal{S} : \underbrace{\mathbb{R}^p \times \cdots \times \mathbb{R}^p}_{n \text{ times}} \times \underbrace{\mathbb{R}^q \times \cdots \times \mathbb{R}^q}_{n \text{ times}} \to \mathbb{R}^p$ is a nonlinear function. In Equation (14.1) the function $\mathcal{S}$ is in the so-called *regressor form* or *input-output form*. For the identification, a set of experimentally collected input-output data are given:

$$\{u_t, \tilde{y}_t\}_{t=1}^N$$

NNs are useful especially when the structure of $\mathcal{S}$ is assumed to be completely unknown, leading to a black-box model. In the following we are giving a description of neural-networks from a functional analysis point of view.

14.1 Brief introduction to Neural Networks

A **Neural network** is the composition of functions called **layers**. A layer ℓ_i is, in particular a function $\ell_i : \mathbb{R}^{\nu_i - 1} \to \mathbb{R}^{\nu_i}$ defined by:

$$\alpha = \ell_i(x) \doteq \phi(W_i x + \beta_i) \tag{14.2}$$

where ν_i is the output dimension of the i-th layer, also referred as the **number of neurons** of the i-th layer; α is the output of the layer; x is the *input of the layer*, while W_i and β_i are respectively the **weight matrix** and the **bias vector**, finally the ϕ is the **activation function** (a nonlinear function in general[1]).
The Equation (14.2) is the formulation of a single layer of the so-called *fully-connected neural network*. A single *hidden* layer is a function $\mathcal{N} : \mathbb{R}^{n_i} \to \mathbb{R}^{n_o}$. For example, for the second hidden layer the output (before passing through the activation function) is given by:

$$z = \mathcal{N}(x) = \ell_2 \circ \ell_1(x) = W_2 \text{ReLU}(W_1 x + \beta_1) + \beta_2 \tag{14.3}$$

In general a NN composed of L layers is the function

$$\begin{aligned} z &= \mathcal{N}(x) = \ell_L \circ \ell_{L-1} \circ \cdots \circ \ell_2 \circ \ell_1(x) = \\ &= W_L \phi(W_{L-1}\phi(...\phi(W_2\phi(W_1 x + \beta_1) + \beta_2) + ...) + \beta_{L-1}) + \beta_L \end{aligned}$$

14.1.1 Why Neural Networks?

There is a nice result on NN: the L_2-norm of the distance between the real f and $\mathcal{N}$ can be made arbitrarily small, in this way the function $\mathcal{N}$ is a "good" approximator for f. This is what is stated in the so-called **Universal Approximation Theorem (Barron, 1993)**.

Theorem 5 (*Universal Approximation Theorem*)**.** *Consider a single-layer neural network* $\mathcal{N} : \mathbb{R}^{n_i} \to \mathbb{R}^{n_o}$ *with* n *neurons and* tanh *or* σ *activation function. Let* $\Omega \subset \mathbb{R}^{n_i}$ *be a compact set. The* ***approximation error*** *of any function* $f : \Omega \to \mathbb{R}^{n_o}$ *can be bounded as:*

$$\|f - \mathcal{N}\|_{L_2} \doteq \int_\Omega \left(|f(x) - \mathcal{N}(x)| \, dx \right)^{1/2} \leq \frac{C}{\sqrt{n}} \tag{14.4}$$

where C *is a suitable constant dependent on the Fourier transform of* f*.*

This vanilla-version of the Universal approximation theorem can be extended for both *multi-layer networks* in presence of *different activation function*. How you can see the higher the number of neurons the more accurate the approximation that $\mathcal{N}$ gives for f. The number of neurons n is an *hyperparameter* to tune, in the sense that trial and error procedure is needed in order to find it.
This was just to introduce in a "simple" way the topic of NN, but how they can be used for identifying a dynamical system? This is the topic of the next section.

14.2 The NNARX model

This was the first use of neural networks for system identification. The main idea is to *replace* $\mathcal{S}$ with a neural network $\mathcal{N}$. If the network is large enough then $\mathcal{N} \approx \mathcal{S}$. For the **true system**, where the data are not affected by noise, we have that:

$$\begin{aligned} \tilde{y}_{n+1} &= \mathcal{S}(\tilde{y}_n, ..., \tilde{y}_1, u_{n+1}, ..., u_1) \\ \tilde{y}_{n+2} &= \mathcal{S}(\tilde{y}_n + 1, ..., \tilde{y}_2, u_{n+2}, ..., u_2) \\ &\vdots \\ \tilde{y}_N &= \mathcal{S}(\tilde{y}_N - 1, ..., \tilde{y}_N - n, u_N, ..., u_N - n) \end{aligned}$$

[1]Such a ϕ can be: a sigmoid, an hyperbolic tangent, a ReLU, or the identity (in the last layer).

that is a set of nonlinear equations where $\tilde{y}_t = y_t$. Replacing $\mathcal{S}$ with $\mathcal{N}_\theta$ we get a system of equations in the parameters

$$\theta \doteq [\text{vec}(W_1)^T, \beta_1^T, ..., \text{vec}(W_L), \beta_L^T] \tag{14.5}$$

We are obtaining:

$$\begin{aligned}
\tilde{y}_{n+1} &= \mathcal{N}_\theta(\tilde{y}_n, ..., \tilde{y}_1, u_{n+1}, ..., u_1) + e_{n+1} \\
\tilde{y}_{n+2} &= \mathcal{N}_\theta(\tilde{y}_n + 1, ..., \tilde{y}_2, u_{n+2}, ..., u_2) + e_{n+2} \\
&\vdots \\
\tilde{y}_N &= \mathcal{N}_\theta(\tilde{y}_N - 1, ..., \tilde{y}_N - n, u_N, ..., u_N - n) + e_N
\end{aligned}$$

where the noise and the mismatch between $\mathcal{S}$ and $\mathcal{N}$ can be modeled as the **equation-error** e_t, also called **one-step ahead prediction error**. The parameters can be found by solving the following unconstrained *nonlinear least-squares problem*:

$$\arg\min_\theta \sum_{t=n+1}^{N} \|e_t\|_2^2 = \arg\min_\theta \sum_{t=n+1}^{N} \|\tilde{y}_t - \mathcal{N}_\theta(\phi_t)\|_2^2 \tag{14.6}$$

where ϕ_t is the regressor. The problem is nonlinear, non-convex and high dimensional, that in general can be solved (to a local minimum) by using the gradient descent algorithm. Note that in Equation (14.6) we are minimizing the one-step ahead prediction error, such a network is giving us *very low performance* in **simulation**[2] since the dynamical part of the system is not explicitly taken into account.

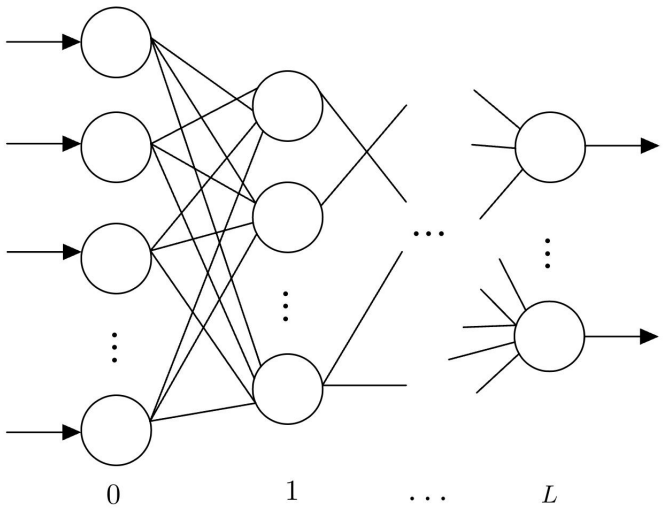


Figure 14.1: Neural network used in the NNARX model

14.3 Recurrent Neural Networks (RNN)

NNARX models use *static NN* taking as input the whole regressor, bad performances are obtained in simulation, since there is not an explicit embedding of the dynamical part of the system in the neural model. A NN which contains the dynamics in its definition is called a **Recurrent Neural Network (RNN)**.

[2]This is the result of the model when the prediction of the output $\hat{y}$ are used in the regressor ϕ_t.

14.3.1 Elman RNN

This RNN model was introduced in the 90s. Here each layer i of the RNN is a nonlinear state-space model with state $h^{(i)}$ of the kind

$$h_t^{(i)} = \phi(W_i x_t^{(i)} + V_i h_{t-1}^{(i)} + \beta_i)$$

where the term $V_i h_{t-1}^{(i)}$ makes a linear combination of the previous states. In the case the output and the state are the same. The first layer takes as input x the input of the system u while the hidden state h is randomly initialized. The last layer that defines the output use the identity activation function.

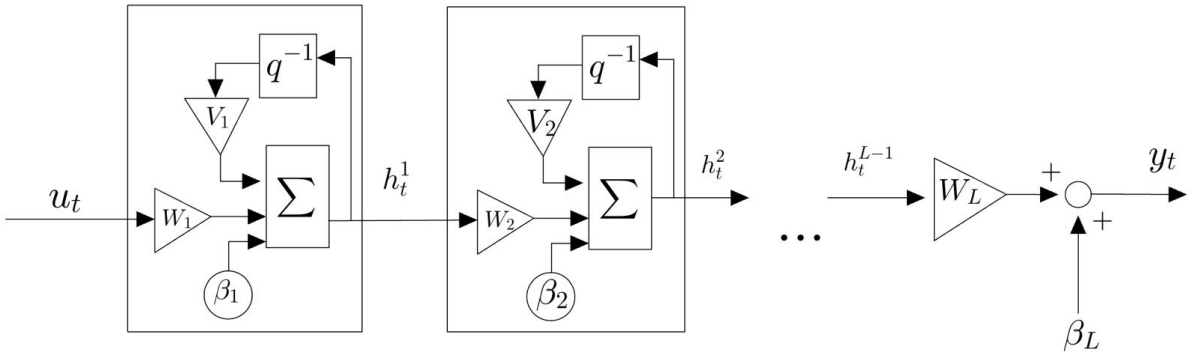


Figure 14.2: Elman RNN structure

14.3.2 Neural State-space models

We know that, in discrete time, in general a nonlinear dynamical system can be represented in the **state-space form** as:

$$\begin{aligned} x_{t+1} &= f(x_t, u_t) \\ y_t &= h(x_t, u_t) \end{aligned}$$

a **neural state-space model** replace the nonlinear functions f and h with static NN $\mathcal{N}_f$ and $\mathcal{N}_h$, then:

$$\begin{aligned} x_{t+1} &= \mathcal{N}_f([x_t^T, u_t^T]^T) \\ y_t &= \mathcal{N}_h([x_t^T, u_t^T]^T) \end{aligned}$$

the static NNs can have an arbitrary number of layers and neurons-per-layer. The difference here (that makes the overall architecture a recurrent one) is that the there is a feedback: the x_t used are the one produced by $\mathcal{N}_f$. This is quite there is no x in the collected data.

14.3.3 Nonlinear Output Error models (NNOE)

Is the dynamic counterpart of the NNARX model, with the only difference that here we use the *predicted output* as input of the network instead of the collected data. It is defined by the recursive equation:

$$y_t = \mathcal{N}([y_{t-1}^T, ..., y_{t-n}^T, u_t^T, ..., u_{t-n}^T]^T)$$

14.3.4 Training RNN and related issues

Differently than the NNARX case, here the **simulation error** is minimized, that is the difference between the measured output and the one produced by the system. We have to solve the following optimization problem in order to retrieve the parameters θ:

$$\arg\min_{\theta} \underbrace{\sum_{t=1}^{N} \|\tilde{y}_t - \hat{y}(u_t, x_0)\|_2^2}_{\mathcal{L}(\theta)} \tag{14.7}$$

where $\hat{y}(u_t, x_0)$ is the output of the RNN given the measured input u_t and initial conditions x_0 usually assumed to be randomly selected or zero. The functional $\mathcal{L}(\theta)$ is a strongly nonlinear function of the parameters.
Also in this case the gradient descent algorithm is used in order to solve the problem. Due to the dynamic nature of the problem the gradients $\nabla_\theta(\hat{y}_t)$ are related each other, this implicitly defines a dynamical system which is unstable as $t \to \infty$ and the gradients go to infinity, or it is asymptotically stable and the gradients are zero with the increasing time. These two conditions are called, respectively, **exploding** and **vanishing gradient issue**.
In order to tackle the issues related to the gradients computation, different models introducing a *direct propagation path* for the gradients were introduced. We are talking about GRU (Gated Recurrent Units) and LSTM (Long-Short term memory) whose details are not discussed here. Even if, there are such architectures the problems are attenuated but not fully solved, moreover a LSTM or a GRU requires **much more parameters** than standard RNN. To conclude this discussion, it is remarkable that the **Transformer** archicture are the best alternative to RNNs, but they are not suitable for system identification since the *attention mechanisms* totally avoid the presence of an explicit feedback in the model.

14.4 Modern trends

What is the best way to avoid the issues related to the gradients? Avoiding to compute them! This is possible if we formulate the identification problem as a **constrained optimization problem** such as:

$$\begin{aligned} &\arg\min_{\theta, y} \sum_{t=1}^{N} \|\tilde{y}_t - y_t\|_2^2 \\ &\text{s.t. } \tilde{y}_{n+1} = \mathcal{N}_\theta(\tilde{y}_n, ..., \tilde{y}_1, u_{n+1}, ..., u_1),\ t = n+1, ..., N \end{aligned} \tag{14.8}$$

The resulting problem is more challenging from the computational point of view since it is constrained and has more variables. These results are presenteed in Cerone et al., [15] but are, of course, outside the scopes of this chapter.

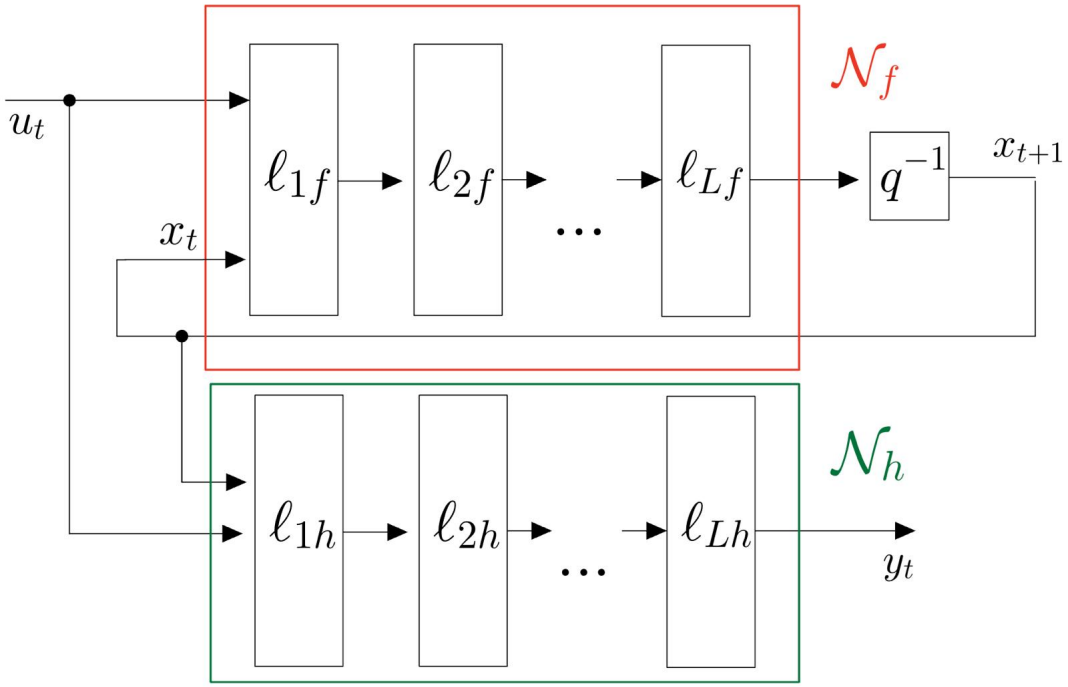


Figure 14.3: Neural state-space model

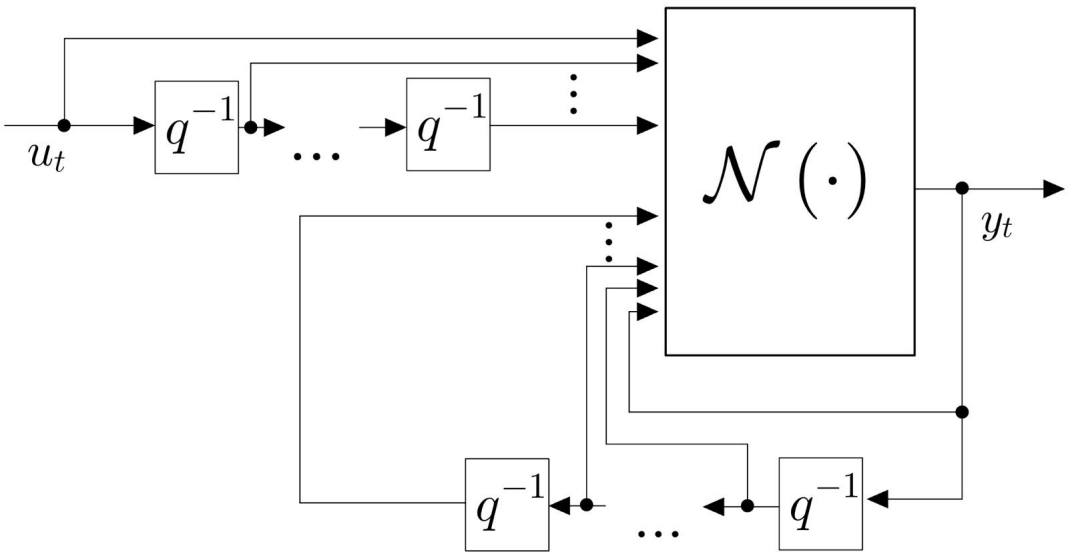


Figure 14.4: NNOE structure

Bibliography

[1] Mohammad Abuabiah et al. "A non-iterative approach to direct data-driven control design of MIMO LTI systems". In: *IEEE Access* 11 (2023), pp. 121671–121687.

[2] Lucíola Campestrini et al. "Unbiased MIMO VRFT with application to process control". In: *Journal of Process Control* 39 (2016), pp. 35–49.

[3] Vito Cerone, Mohammad Abuabiah, and Diego Regruto. "An H_∞ method to design the reference model in direct data-driven control design". In: *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE. 2020, pp. 6004–6009.

[4] Vito Cerone, Dario Piga, and Diego Regruto. "Bounded error identification of Hammerstein systems through sparse polynomial optimization". In: *Automatica* 48.10 (2012), pp. 2693–2698.

[5] Vito Cerone, Dario Piga, and Diego Regruto. "Bounding the parameters of block-structured nonlinear feedback systems". In: *International Journal of Robust and Nonlinear Control* 23.1 (2013), pp. 33–47.

[6] Vito Cerone, Dario Piga, and Diego Regruto. "Computational load reduction in bounded error identification of Hammerstein systems". In: *IEEE Transactions on Automatic Control* 58.5 (2012), pp. 1317–1322.

[7] Vito Cerone, Dario Piga, and Diego Regruto. "Enforcing stability constraints in set-membership identification of linear dynamic systems". In: *Automatica* 47.11 (2011), pp. 2488–2494.

[8] Vito Cerone, Dario Piga, and Diego Regruto. "Improved parameter bounds for set-membership EIV problems". In: *International Journal of Adaptive Control and Signal Processing* 25.3 (2011), pp. 208–227.

[9] Vito Cerone, Dario Piga, and Diego Regruto. "Set-membership error-in-variables identification through convex relaxation techniques". In: *IEEE Transactions on Automatic Control* 57.2 (2011), pp. 517–522.

[10] Vito Cerone, Valentino Razza, and Diego Regruto. "A unified framework for the identification of a general class of multivariable nonlinear block-structured systems". In: *International Journal of Robust and Nonlinear Control* 31.15 (2021), pp. 7344–7360.

[11] Vito Cerone, Valentino Razza, and Diego Regruto. "Set-membership errors-in-variables identification of MIMO linear systems". In: *Automatica* 90 (2018), pp. 25–37.

[12] Vito Cerone and Diego Regruto. "Parameter bounds evaluation of Wiener models with noninvertible polynomial nonlinearities". In: *Automatica* 42.10 (2006), pp. 1775–1781.

[13] Vito Cerone, Diego Regruto, and M Abuabiah. "A set-membership approach to direct data-driven control design for SISO non-minimum phase plants". In: *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE. 2017, pp. 1284–1290.

[14] Vito Cerone, Diego Regruto, and M Abuabiah. "Direct data-driven control design through set-membership errors-in-variables identification techniques". In: *2017 American Control Conference (ACC)*. IEEE. 2017, pp. 388–393.

[15] Vito Cerone et al. "A new framework for constrained optimization via feedback control of Lagrange multipliers". In: *arXiv preprint arXiv:2403.12738* (2024).

[16] Vito Cerone et al. "Set-membership identification of continuous-time systems through model transformation". In: *2022 IEEE 61st Conference on Decision and Control (CDC)*. IEEE. 2022, pp. 868–873.

[17] Rolf Johansson. "Identification of continuous-time models". In: *IEEE Transactions on Signal Processing* 42.4 (1994), pp. 887–897.

[18] Mario Milanese and Gustavo Belforte. "Estimation theory and uncertainty intervals evaluation in presence of unknown but bounded errors: Linear families of models and estimators". In: *IEEE Transactions on automatic control* 27.2 (1982), pp. 408–414.

[19] Mario Milanese et al. *Bounding approaches to system identification*. Springer Science & Business Media, 2013.

[20] Rimantas Pupeikis, Dalius Navakauskas, and Lennart Ljung. *Identification of Wiener systems with hard and discontinuous nonlinearities*. Linköping University Electronic Press, 2003.

[21] Gustavo R Gonçalves da Silva, Alexandre Sanfelici Bazanella, and Lucíola Campestrini. "On the choice of an appropriate reference model for control of multivariable plants". In: *IEEE Transactions on Control Systems Technology* 27.5 (2018), pp. 1937–1949.

[22] Klaske Van Heusden, Alireza Karimi, and Dominique Bonvin. "Data-driven model reference control with asymptotically guaranteed stability". In: *International Journal of Adaptive Control and Signal Processing* 25.4 (2011), pp. 331–351.