LABORATORY OF ROBUST IDENTIFICATION AND CONTROL

Lecture notes

Carlo Migliaccio

AA 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:

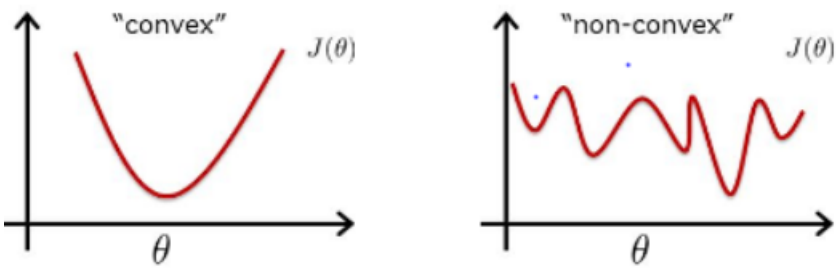
$$\hat{\theta} = \arg\min_{\theta} J(\theta) \tag{1.2}$$

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)**.

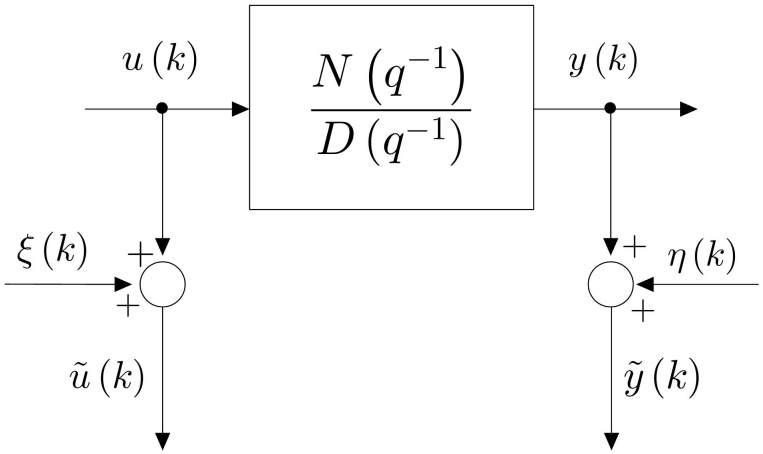


Figure 2.1: EIV SysId setting

- 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:

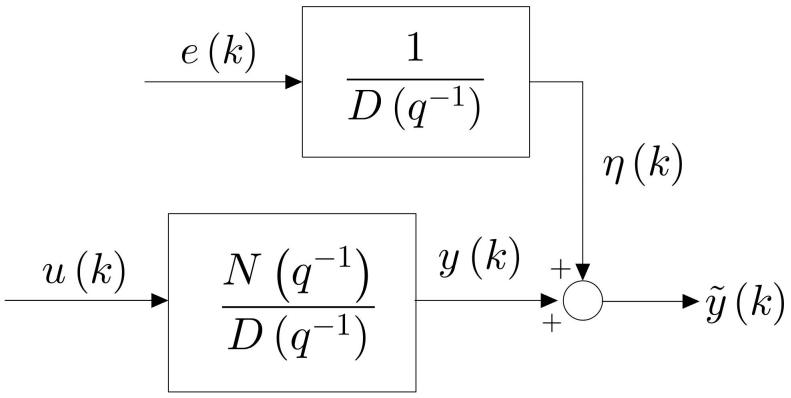


Figure 2.2: Equation Error (EE) noise structure

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 $\eta(k)$.

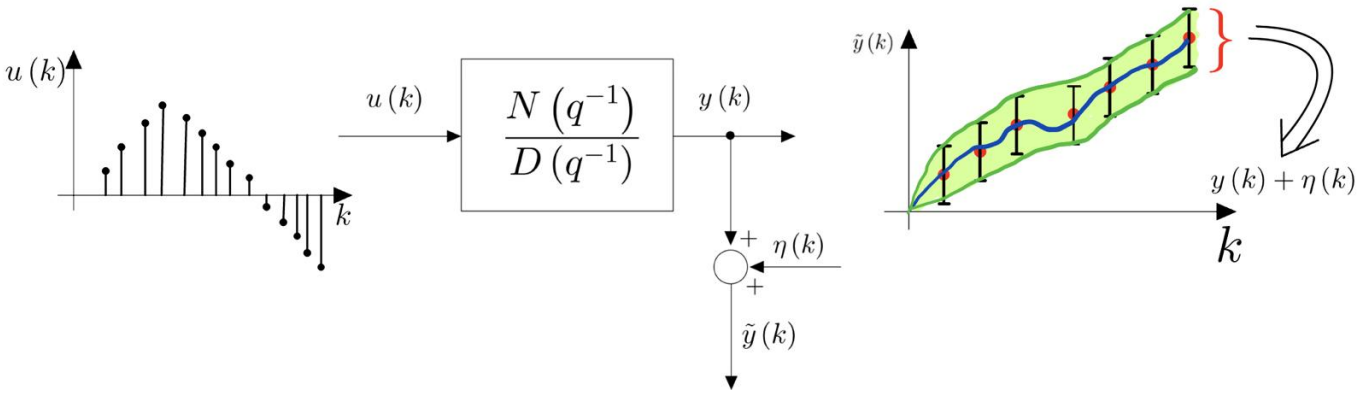


Figure 3.1: OE set-up experiment

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.

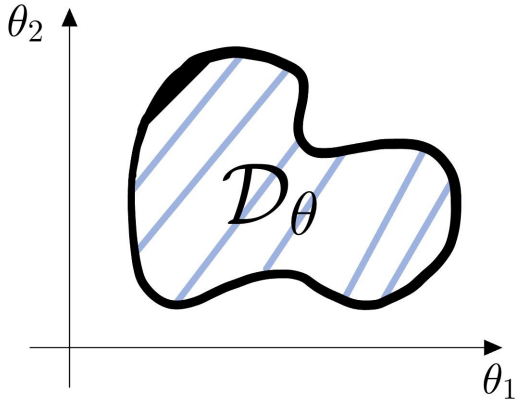
$$\begin{aligned}\mathcal{D}_\theta = &\{\theta \in \mathbb{R}^p : \ \tilde{y}(k) + \theta_1\tilde{y}(k-1) + \theta_2\tilde{y}(k-2) + \\ &- \theta_3\tilde{u}(k) - \theta_4\tilde{u}(k-1) - \theta_5\tilde{u}(k-2) = e(k), \quad k = 3, ..., H \\ &|e(k)| \leq \Delta_e, \quad k = 1, ..., H\} = \\ &\{\theta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

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.*

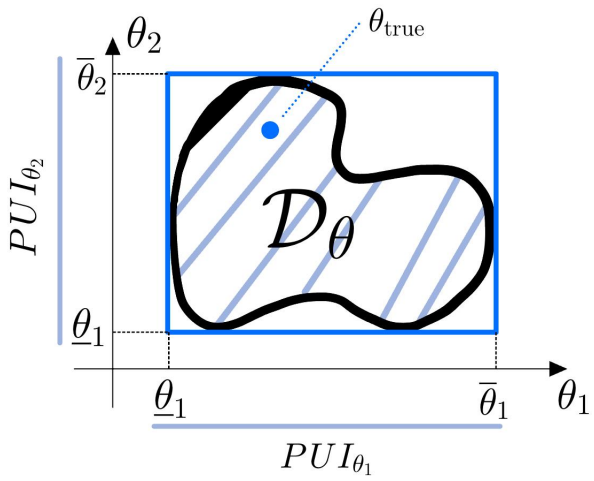
Question Q2 Assuming that the a-priori assumptions on the system and on the noise are correct, then θ_{true} is guaranteed to belong to $\mathcal{D}_\theta$ (this is very important for further theory development).

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

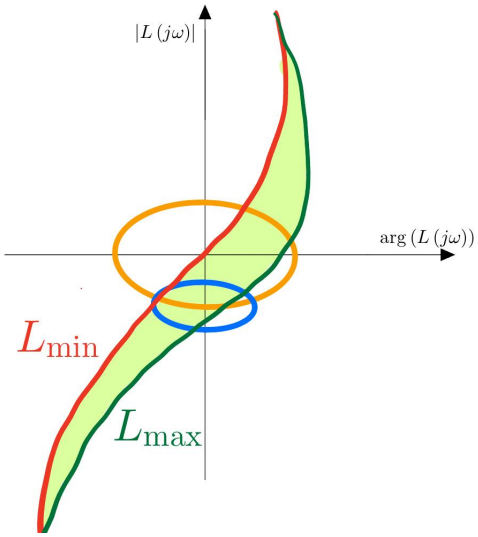


Figure 3.2: Nichols plot of the loop function

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).

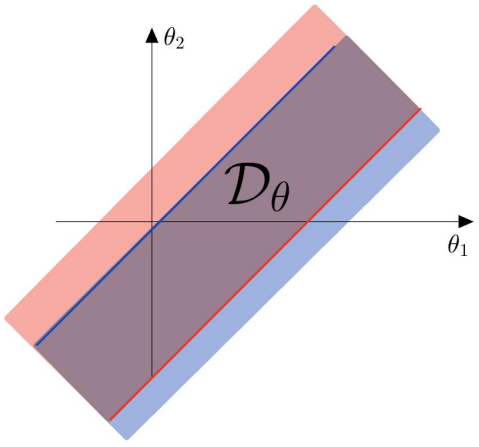


Figure 3.3: $\mathcal{D}_\theta$ resulting from a single equation

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 $k = 2$.

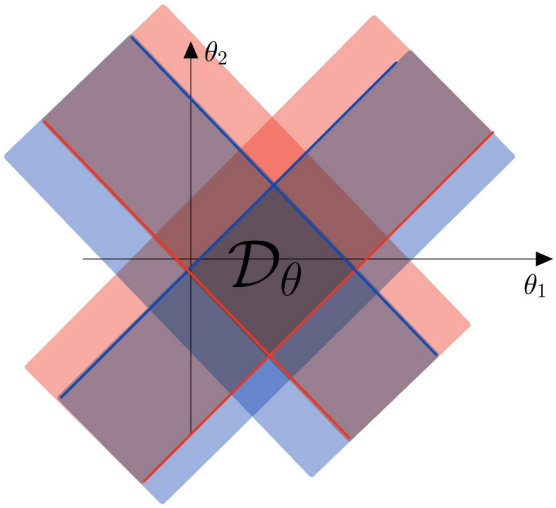


Figure 3.4: $\mathcal{D}_\theta$ resulting from a pair of equations

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.

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 $\mathbb{R}^3$.

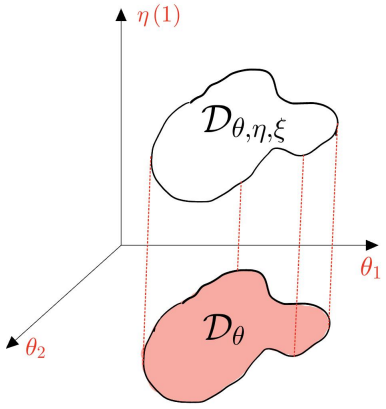


Figure 4.1: Example in $\mathbb{R}^3$ of the Extended feasible parameter set

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:

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

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.

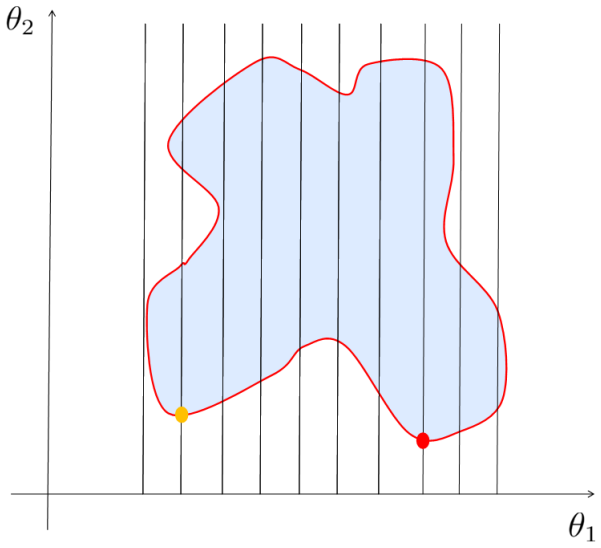


Figure 4.2: Example of min a linear objective (thin line) over a non-convex set (sky-blue)

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 $\mathcal{S}$.

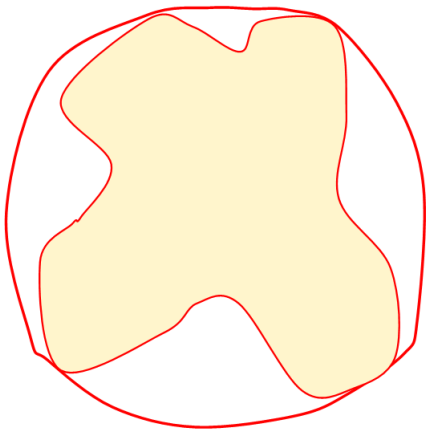
A very-nice property is that the **min/max** computed on the convex hull is equal to the one computed on the original set. In this way we gain the convexity of the problem.

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:

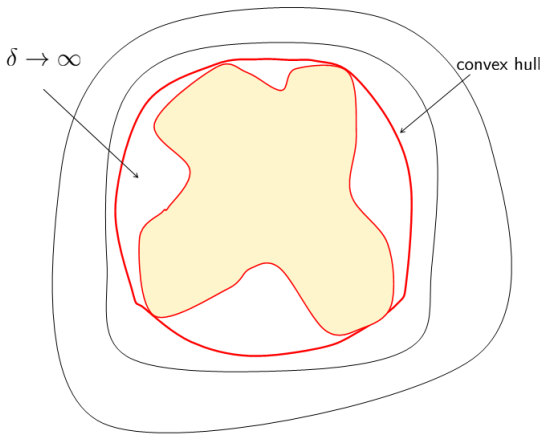
$$\min_{x \in C_{\mathcal{S}}} f(x) = \min_{x \in \mathcal{S}} f(x) \tag{4.8}$$

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:

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

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:

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

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.solver='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`

Part II

Direct Data-Driven Control

Part III

Appendix

Chapter 5

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);
```

5.1 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)
```

5.2 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)
```

5.3 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 6

Lab2: FPS and PUI with Equation-Error noise structure

In this problem we assume that the plant to be identified is exactly described as a discrete-time LTI model described by the following transfer function: