Control System Design
Information for the Laboratory Sessions

Ir. Xavier Jordens, Ir. Robin Wilmart & Pr. Dr. E. Garone
Control Engineering and System Analysis Department
Faculty of Applied Science
Université Libre de Bruxelles (U.L.B.)
Brussels, Belgium

Contacts:
Xavier.Jordens@ulb.be,
Robin.Wilmart@ulb.be,
Emanuele.Garone@ulb.be

November 3, 2021

Contents

Part I

Documentation

Chapter 1

Introduction

1.1 Objective

The goal of the laboratories is to design a **controller** which **meets a list of requirements** for a real plant. The students have to design it without knowing the right solution beforehand. They are given the responsibility to find the most appropriate solution to face the challenges and convince the jury about the **validity of the proposed solution**. The professor and the teaching assistant are available to guide the students by giving hints, discussing the proposed solutions, issues, unexpected behaviors, etc. Each team is assigned to one plant to control it. It must **at least reach the basic specifications** described in this document.

1.2 Planning

The students are suggested to keep in mind their objectives for the laboratory sessions during the lectures. Each team has to provide **before the first lab a first global planning attempt** for the **4 following lab sessions** and a **detailed planning for the first lab session**.

The goal of the project is to choose the most appropriate design solutions, do the parameters tuning, simulations and experimental tests to develop control schemes that are performant enough to meet the required specifications. It is required **for each lab session** to provide an **adapted global planning for the remaining lab sessions** and a **detailed planning for the current lab session**.

1.3 Laboratory and Extra Sessions

There are five laboratory sessions of 4 hours each in total. Some group may benefit up to 2 extra hours of lab access. The extra sessions are time slots of 1 hour dedicated to carry out experiments. They are allowed **only by appointment**. Registrations have to include an explanation of the aim of the tests and a **precise** description of the experiment(s) (control structure with the various gains, input signal(s), data collected, ...). Please note that a lack of time management will never be a valid explanation to justify extra laboratory sessions. Students are encouraged to make the most accurate schedule by dividing their work in several elementary tasks and identifying which one can be done in parallel to split the work equally between each group members.

1.4 Actions against COVID-19

In an effort to contain the spread of the corona virus, an hydro-alcoholic solution will be dispensed at the entrance of the laboratory. Students are invited to use it to clean their hands before entering into the lab. The use of a mask that covers the mouth and the nose is mandatory during the laboratory sessions. If they wish, the students can also clean the keyboard and the equipment before using it.

1.5 Tools

During the laboratory sessions, the students have access to one fixed computer equiped with a data acquisition system, Matlab and Simulink (with Data acquisition toolbox, control system toolbox and symbolic math toolbox). An example code is also given to the students to help them in their project.

1.6 Prerequisites and References

These laboratory sessions illustrate the lectures of Control System Design whose didactic material is available on http://www.gprix.it/teachingH407_2020.htm. The prerequisite for this course is **Basic Control Theory**. For one who should not be completely comfortable with this matter, it is worth to peruse the following book:

- **Feedback Control of Dynamic Systems -** Gene F. Franklin, J. David Powell, Abbas Emami-Naeini

1.7 Organization of the Evaluation

The last week of course is devoted to the oral presentation. Also, each team has to write a **technical report**. During the oral presentation, the students are invited to ask questions to the other teams.

During a whole laboratory session, students might be invited to stop their work in order to be individually assessed on basis of their knowledge about control theory by the completion of a small written or oral test. The oral assessment can also be done in continuity with the current laboratory session during interactions between the student(s) and the assistant(s). This scoring system will contribute to personalize the final score of each group individual.

1.7.1 Technical report

The technical report contains a description of the final results and a discussion of the design choices. It should be not more than 20 pages.

Assessment Criteria

The evaluation of the technical report is influenced by the following criteria:

- Performance and quality of the proposed solutions
- Possible comparison between alternative solutions
- Capability to motivate and defend your design choices (with theoretical explanation)
- Capability to face unexpected problems due to the real nature of the system and to formulate hypotheses able to explain the abnormal behaviors
- Quality of the written report

1.7.2 Weekly assessment

Be aware that your work will also be assessed each week by the assistant depending on :

- the way you behave during the laboratory sessions,
- the strategies you elaborate to control your plant,
- the individual tests about control theory,
- and the answers you formulate to the questions he may ask.

1.7.3 Technical presentation

A 10 minutes technical presentation has to be done using a projector. These presentations take place in front of all the students composing the laboratory sessions. The presentation is then followed by a question time.

1.7.4 Additional Assessment criteria

The final mark received for the project is influenced by the following criteria:

- Capability to motivate and defend the design choices
- Quality of the oral presentation
- Quality of the questions asked to the other groups
- Capability to answer the questions

The final project mark is a personal evaluation and may vary depending on the individual performance.

Chapter 2

Methodology

2.1 Preliminaries

Before thinking about the control law, it is **important** to **analyze the system** but also to **list the required specifications**. In general, the first step is to:

1. Define the available **sensors** and **actuators**;

2. Define the **specifications of the control**.

Classically, in control engineering, the main tasks are first to **identify a model of the system** and then to **control it**. The identification is motivated by the fact that it is important to have a mathematical model, possibly accurate enough with the knowledge of its limits, to be able to **analyze**, **validate** and **simulate**. The laboratory sessions are there to confirm the hypotheses and test the controller. It is also convenient to understand the given sample codes and intuitively understand the real plant. Therefore, **first of all**, it is advised to **at least** write down the **physical interconnected concepts** and, **if possible**, write down the **believed equations** that describe the system to get a first attempt of a mathematical model.

2.2 Identification

The identification of a system is a vast and complex subject and is mainly described in scientific literature. Three usual approaches exist to identify the system:

- The Black box modelling

- The Gray box modelling

- The White box modelling

Depending on your plant, you might have to use preferably one instead another one.

Generally, it is advised to write down an **attempt of a mathematical description** in order to **know the order of the system** if possible. In this case, the **gray box approach** can be used to determine the parameters. In the other case, a **black box approach** is prefered to more rapidly determine the transfer function **without knowing the system order**.

It is **very important** to keep in mind that, **in most of the cases**, the model is going to be **linearized around an operation point**. It is therefore judicious to appropriately **choose this point** in function of the given **static characteristics**. In reality, mainly due to friction, external perturbations and saturation, the system is in general **non-linear**. Thus, in most of the cases, a real plant **does not exhibit** the properties of **additivity and homogeneity**.

2.2.1 Black Box Approach

The black box method is usually used when we do not know the physical laws that govern the system dynamics. This model is represented in Figure 2.1. The classical approach is to:

- Create a step/impulse
- Observe the output
- Determine the order of the transfer function
- Compute the transfer function using Matlab

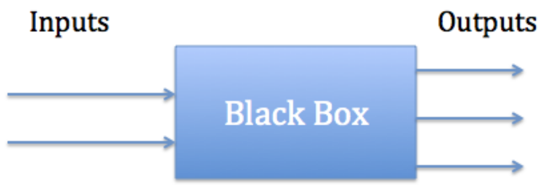


Figure 2.1: Black box representation

In **linear control theory**, it is proven that analyzing the **step/impulse response** is **enough** to **completely characterize the system**. To determine if one has to use a step or an impulse, one has to know the **stability of the system**. If the system already implements an integrator, it is more convenient to use an impulse instead of a step as an input (explain why).

The advantage of this approach is that it is **quick and reliable**. The disadvantage is that **the order of the system cannot be explained physically** and there are some **uncertainties in the model**.

After creating a step/impulse, one has to know the tools to determine the **order of the system** (i.e. the number of poles). This is done by observing the step/impulse response. Matlab already computes the approximate transfer function analyzing the acquired data from the analog computer[1]. The student just has to know the order of the system to enter it in Matlab.

In general, a real system is characterized by an **infinite number of poles** since a step excites all the frequencies of the system. But in order to facilitate the study, one can characterize the system with one or two dominant pole(s) and without any zero as a first approximation. In the following, to simplify the rationale, it is assumed that the system has **no pole at the origin**.

- **First Order Step Response**

  A typical step response of a first order system is represented in Figure 2.2.

  The derivative at the origin is different from zero and there is no overshoot nor oscillation.

- **Second Order Step Response**

  In this case, oscillation may appear and/or an overshoot. Typical step responses of a second order system is represented in Figure 2.3.

  It is seen that the red curve is quite similar to a first order system. However, the main difference is that the derivative at the origin is equal to zero.

[1] For more information on the algorithm used for the computation of the transfer function from the acquired data, please refer to the appendix.

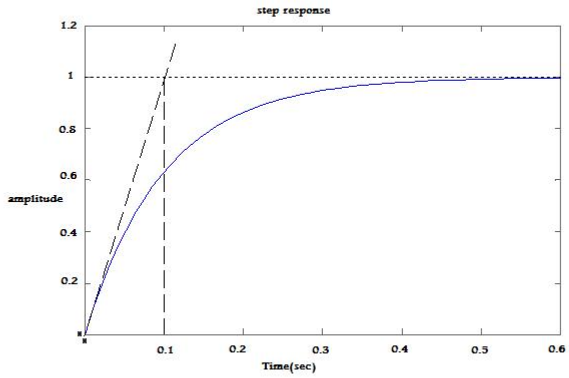


Figure 2.2: Typical step response of a first order system

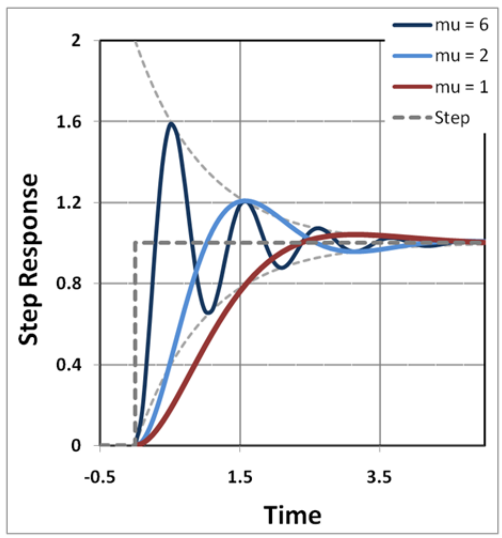


Figure 2.3: Typical steps response of a second order system

2.2.2 Gray Box Approach

With this approach, the physical understanding of the system can be used to derive the **dynamics of the system**. This approach is used when we are able to derive the system dynamics but when we do not know the system numerical parameters. Different tools can be used such as the Lagrangian function, Kirchkoff's circuit laws for electrical systems, etc. Usually, these approaches lead to **continuous equations** and, after linearization if needed, to a **transfer function**.

This approach has the advantage to understand physically the system and to understand the rationale that leads to the explanation of the number of poles and zeros. The disadvantage is that bad assumptions can lead to a wrong model and thus to a bad controller. This is why, the **validation part** is a crucial point to confirm the hypotheses made before.

2.2.3 White Box Approach

The White Box approach is similar to the Gray Box approach except that in this method, the numerical values of the system dynamics parameters are known and a transfer function (after linearization if needed) can thus be derived.

2.3 Control Design

In general, the continuous model is sufficient to design the controller. However, if one wants to increase the performance of the controller by working at low frequencies, one can use the discrete model and thus the Z-transform to design the controller. Indeed, due to the discrete nature of a computer, the input injected in the system is also discrete (see Figures 2.4 and 2.5). However, in the general case, the Laplace transform is sufficient to design the controller. For

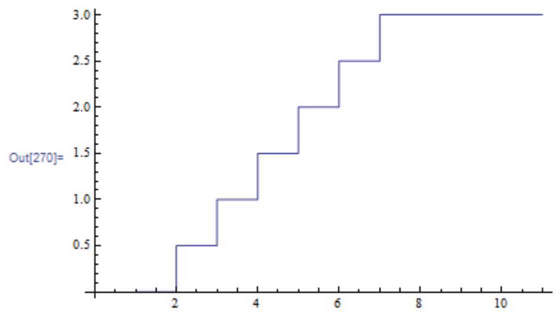


Figure 2.4: Typical discrete signal that the plant might receive.

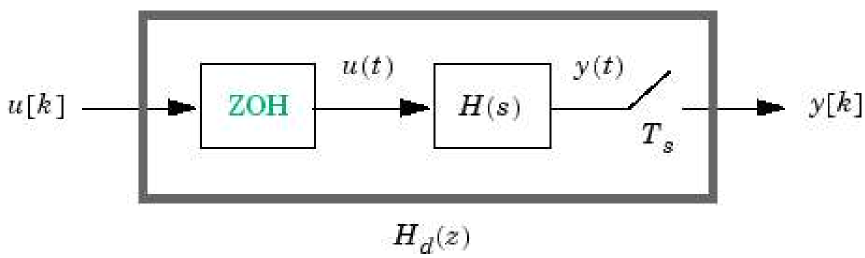


Figure 2.5: Zero order hold illustration for the model of a computer. $H_d(z)$ is the equivalent z-transform of the system.

numerical computer control design, the classical approach is to first design the controller in the continuous domain with well-known tools such as the harmonic analysis including the use of the Bode diagram or the pole placement theory with the root locus for instance, then discretize the system by holding the signal at the entrance of the system during a certain sampling time. This design method is called **design by discrete equivalent** or **emulation**.

Chapter 3

Documentation and Tools

3.1 Analog Computers

3.1.1 Introduction

For each plant, there is an interface available to the students to connect the measurements and the actuators to Matlab. This interface is called an analog computer (see Figure 3.1).

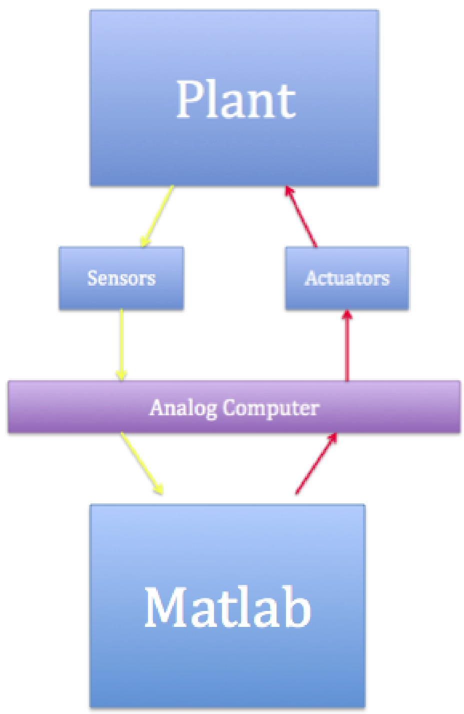


Figure 3.1: Interface between the plant and Matlab.

The values sent from the analog computer to Matlab are the outputs observed from the studied plant. In the other case, the values are the inputs which control the actuators of the plant.

In the most general case, the designed controller is a closed loop fed back by the outputs of the system. The controller is designed by manipulating the output data received from the analog computer using a Matlab script code. These manipulated data are then sent to the inputs of the analog computer, which in its turn sends them to the actuators of the plant to control it.

3.1.2 Color Standard Code

The color standard code for the plugs is:

- Yellow for the **inputs**
- Red for the **outputs**
- Green for the **interconnections**

There are 8 available inputs and 2 available outputs that are interfaced with Matlab using a Data Acquisition (DAQ) Board (see Figure 3.2). The tuners present on Figure 3.2 are potentiometers, generally used to tune the input voltage manually.

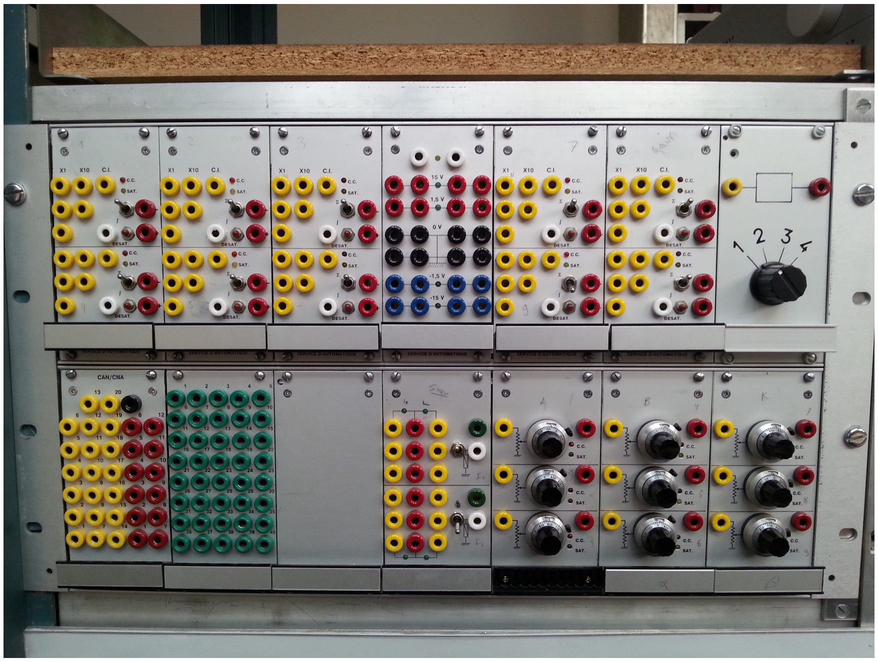


Figure 3.2: Picture of the interface panel of the analog computer

3.2 Matlab

3.2.1 Codes

To simplify the work of the students, a Matlab code is available to identify the system. A sample code for control is also provided. The students are asked to modify these codes regarding their strategy to control the plant.

Script Code

The files are available on the laboratory computer desktop in the folder **MATH-H-407**. This folder is protected against writing. Therefore, copy and paste the files in a folder of your choice before modifying them. Don't forget to save the files on a USB stick. Indeed we cannot guarantee that your files will remain untouched from one laboratory session to the other. Hereafter are two examples of Matlab codes : a sample code and a control code. It is recommended to read and understand them. Take them as an inspiration source for your project. Finally, to compile and launch your program, click on the **Run** button.

Acquisition code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Control System Design Lab: Sample Code
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Setup
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

openinout; %Open the ports of the analog computer.
Ts=;%Set the sampling time.
lengthExp=; %Set the length of the experiment (in seconds).
N0=lengthExp/Ts; %Compute the number of points to save the datas.
Data=zeros(N0,1); %Vector saving the datas. If there are several datas to save, change "1" to the ...
    number of outputs.
DataCommands=ones(N0,1); %Vector storing the input sent to the plant.
cond=1; %Set the condition variable to 1.
i=1; %Set the counter to 1.
tic %Begins the first strike of the clock.
time=0:Ts:(N0-1)*Ts; %Vector saving the time steps.

%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Loop
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

while cond==1
    input=; %Input of the system.
    anaout(input,0); %Command to send the input to the analog computer.
    [in1,in2,in3,in4,in5,in6,in7,in8]=anain; %Acquisition of the measurements.
    Data(i,1)=in1; %Save one of the measurements (in1).
    t=toc; %Second strike of the clock.
    if t>i*Ts
        disp('Sampling time too small');%Test if the sampling time is too small.
    else
        while toc<=i*Ts %Does nothing until the second strike of the clock reaches the sampling time set.
        end
    end
    if i==N0 %Stop condition.
        cond=0;
    end
    i=i+1;
end

closeinout %Close the ports.

%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Plots
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

figure %Open a new window for plot.
plot(time,Data(:,1),time,DataCommands(:)); %Plot the experiment (input and output).
```

Identification Code

```
u=DataCommands.';
y=Data(:,1).';
offset=0; %Operating point
SystemOrder=[0 1]; %Number of zeros and of poles (0 and 1), respectively.
sysIdent=IdentifySystem(u,y-offset,SystemOrder,Ts);
```

```
plot(time,y-offset,'.');
hold on;
lsim(sysIdent,u,time);
```

IdentifySystem() function

```
function sys = IdentifySystem(input,output,S_Order,Ts)

global u y SystemOrder k;

u = input;
y = output;
SystemOrder = S_Order;
k = 0:Ts:(length(u)-1)*Ts;

theta_0 = rand(1,SystemOrder(1) + SystemOrder(2) + 1);
% Nb parameters = Nb poles + Nb zeros + 1 (Static Gain)
theta = fminsearch('cost',theta_0);
Num = [];
Den = [];
for i = 1 : SystemOrder(1)+1
    Num = [Num theta(1,i)];
end
for j = i+1 : SystemOrder(2)+SystemOrder(1)+1
    Den = [Den theta(1,j)];
end
Den = [Den 1];
sys = tf(Num,Den);
%sys = c2d(tf(Num,Den),Ts,'matched');
```

cost() function

```
function J = cost(theta)

global u y SystemOrder k
Num = [];
Den = [];
for i = 1 : SystemOrder(1)+1
    Num = [Num theta(1,i)];
end
for j = i+1 : SystemOrder(2)+SystemOrder(1)+1
    Den = [Den theta(1,j)];
end
Den = [Den 1];

sys = tf(Num,Den);
hatY = lsim(sys,u,k)';

epsilon = y - hatY;

J = epsilon*epsilon';
```

Control Sample Code

```
%*********************************************************************
%                             Initialization
%*********************************************************************
openinout;
```

```
Tcycle=0.01;
lengthExp=20;
N0=lengthExp/Tcycle;
Data=zeros(N0,2);
DataCommands=zeros(N0,1);
cond=1;
i=1;
reference=6.5;
DataCommands(:)=reference;
%**********************************************************************
%                           Save in real time
%**********************************************************************
while cond==1
    tic
    [in1,in2,in3,in4,in5,in6,in7,in8]=anain;

    % PUT YOUR CONTROL LAW HERE

    out1=Kp*(reference-in1);
    % e.g. a proportional control law
    anaout(out1,0);
    %anaout is the function that applies the desired voltage out1.

    % STORE THE MEASUREMENTS, THE COMMAND OR ANY OTHER USEFUL SIGNALS

    Data(i,1)=in1;
    Data(i,2)=in2;
    DataCommands(i)=out1;


    i=i+1;
    t=toc;
    if t>Tcycle
        disp('Sampling time to small');
    else
        while toc<=Tcycle
        end
    end
    if i==N0+1
        cond=0;
    end
end
%**********************************************************************
%                               Plot
%**********************************************************************
closeinout;
i=i-1;
time=0:Tcycle:(i-1)*Tcycle;

figure
plot(time, Data(:,1), time, DataCommands(:),time,Data(:,2));
```

3.2.2 Simulink

To create a simulink model, go on the upper left of the window and click on **New** → **Simulink Model**. A new window will appear (see Figure 3.3). You can change the lapse of resolution time on the upper window (initially set at 10.0 s).

To create the different components of the control scheme, click on **View** → **Library Browser**. A new window will appear (see Figure 3.4).

In order to create a control scheme, drag the components needed from the library browser and constitute the desired scheme. You can also search for a component by typing in the toolbox present on the upper left of the window. The most usual tools are:

- Step

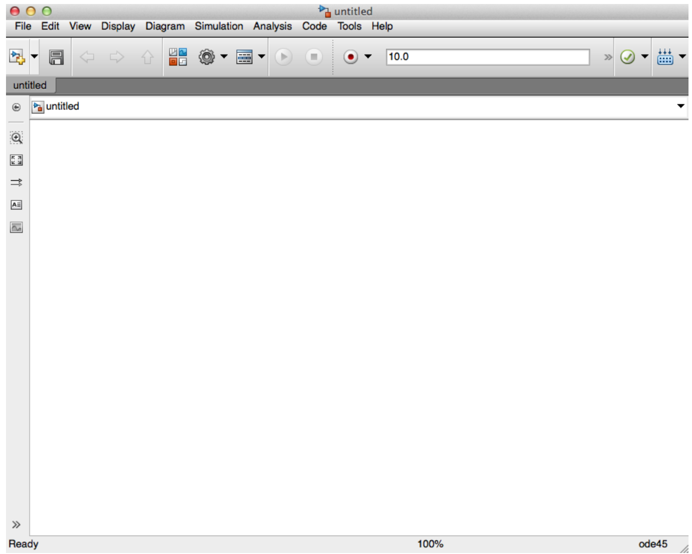


Figure 3.3: Window for simulink

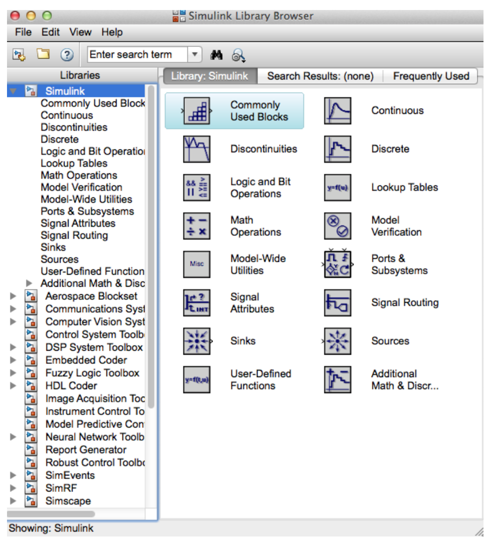


Figure 3.4: Library browser window

- Sum
- Transfer Fcn
- Integrator
- Derivative
- Gain
- Scope

Do not forget to put the scope in order to observe the signals. To connect the elements, just drag with the mouse from one component to the other. A random example is shown in Figure 3.5.

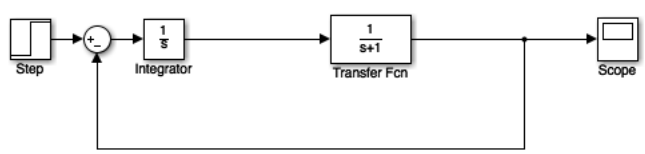


Figure 3.5: Example of a simulink model

Part II

Project Descriptions

Chapter 4

The Ball in the Tube

4.1 Plant description

A light ball is placed inside a tube. A fan generates an air flux which lifts the ball. The objective is to control the vertical position of the ball.

4.1.1 Instrumentation

- Sensors:
  - Tachometer (velocity sensor)
  - Infrared sensor
- Actuator: The fan is actuated by the current driven DC motor.

4.2 Specifications

4.2.1 Basic requirements

Stabilization of the ball at a fixed position without any steady state error. Beware of the distance sensor characteristic.

4.2.2 More advanced requirements

- Switch between two different points with a maximum of $1cm$ overshoot.
- Switch between two different points with a maximum of $1cm$ overshoot in maximum $2s$.

4.3 Documentation

4.3.1 Velocity sensor: Maxon motor DCT 22 tacho

- linearity between 500 and 5000 min^{-1}: $\pm 0.2\%$
- Output voltage per 1000 rpm: 2.5V

4.3.2 Infrared distance sensor (SHARP GP2YOD02YKOF)

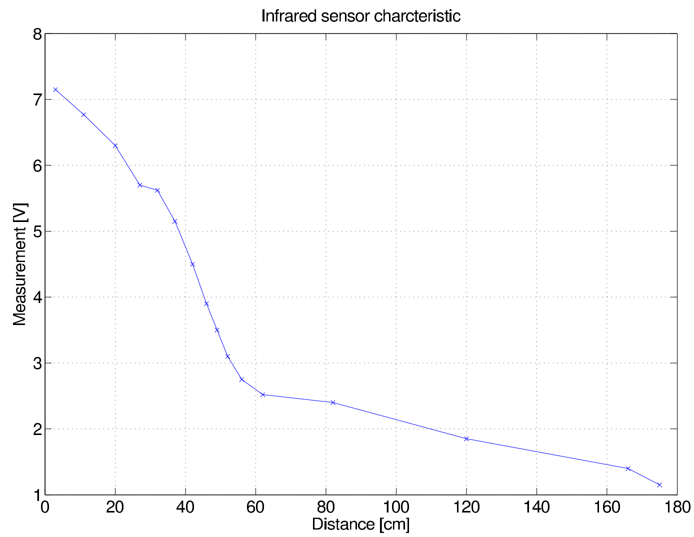


Figure 4.1: Ball in the tube: Infra-red distance sensor

4.3.3 Current - velocity static characteristic

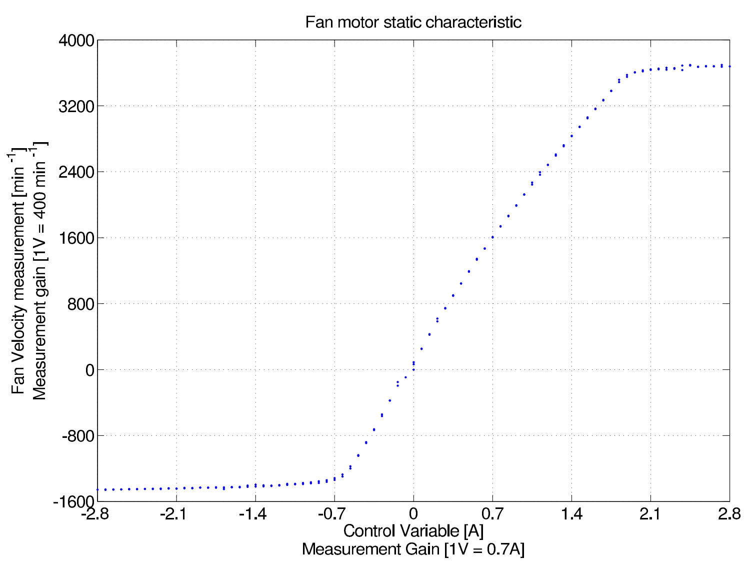


Figure 4.2: Ball in the tube plant: Current - velocity static characteristic

Chapter 5

The Heat Exchanger

5.1 Plant description

The heat exchanger is composed of two circuits. The first one is an air circuit with possibility to recycle the air. This circuit includes a heating section. The second circuit is a water circuit. Both circuits are interconnected by a heat exchanger between the air and the water. The objective is to have the water achieving a given temperature.

5.1.1 Instrumentation

- Sensors:
  - 4 water temperature sensors: 1 at the entry, 1 after 33% and 1 after 66% of the exchanger and 1 at the exit.
  - 4 air temperature sensors: 1 at the entry and 1 at the exit of the heating section an 1 at the entry and the exit of the exchanger
- Actuator: a warming resistor is installed on the air circuit.

One may manually allow partial or full recycling of the air and variation of the water flow.

5.2 Specifications

5.2.1 Basic requirements

Considering the water flow constant without any air recycling, warm the water flow at a given temperature with zero steady state error and no overshoot.

5.2.2 More advanced requirements

- Follow a water temperature trajectory (simulating a given temperature profile for a chemical reaction)

5.3 Documentation

5.3.1 Heating - air temperature static characteristic (without recycling)

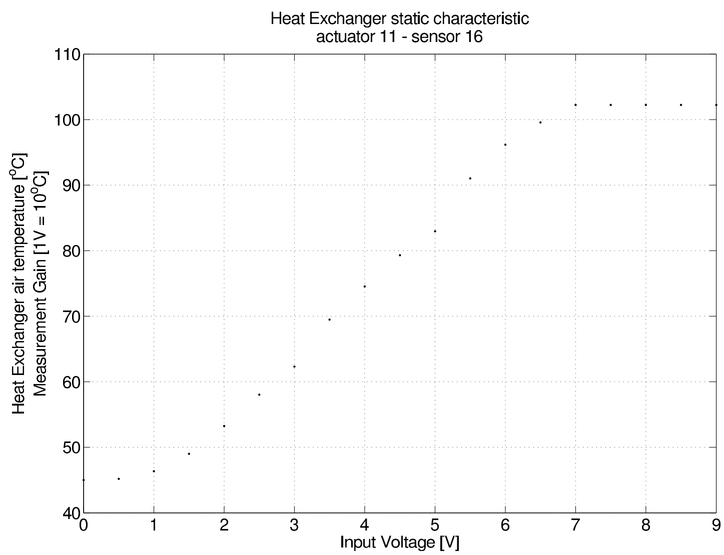


Figure 5.1: Heat exchanger: Voltage - Air temperature static characteristic

Chapter 6

The Ball and Beam

6.1 Plant description

A small ball is placed on a gutter that constrains the displacements of the ball along its direction. The purpose is to control the position of the ball along the beam by adjusting its angle. The beam is attached at one end to a roller support, and at the other end to a slider that can move vertically along a linear guide by means of a ball screw drive. The screw is actuated by a current-driven DC motor. The objective is to stabilize the ball at an arbitrary position of the beam.

6.1.1 Instrumentation

- Sensors:
  - The velocity of the DC motor is measured by a tachometer generator.
  - The horizontal angle of the beam is measured by a potentiometer.
  - The position of the ball along the beam is measured by an infrared distance sensor.
- Actuator: The slider is moved by a DC motor.

6.2 Specifications

6.2.1 Basic requirements

Stabilization of the ball at any available position.

6.2.2 More advanced requirements

- Switch between two different points with a maximum of $1cm$ overshoot.
- Switch between two different points with a maximum of $0.5cm$ overshoot in maximum $10s$.

6.3 Documentation

6.3.1 Infrared distance sensor

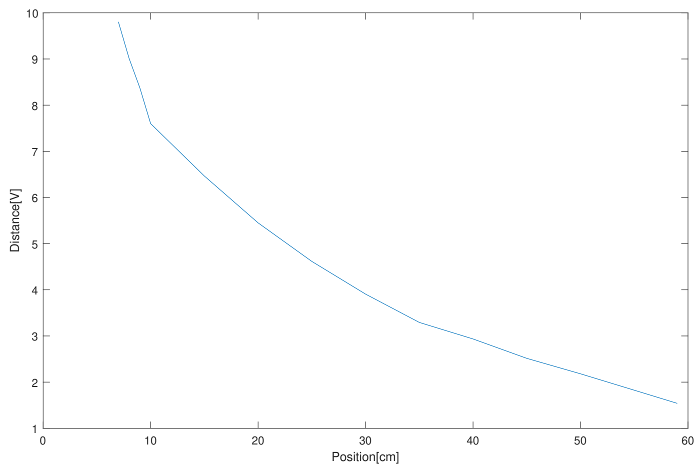


Figure 6.1: Ball and Beam: Infra-red distance sensor

6.3.2 Current - velocity static characteristic

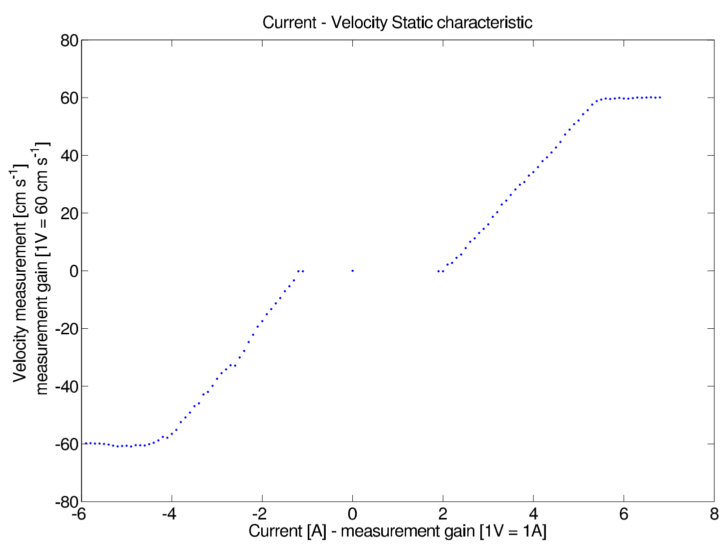


Figure 6.2: Ball and Beam: Current - velocity static characteristic

Chapter 7

The Temperature Control

7.1 Plant description

The temperature control plant can be compared to a house temperature simulator. It is composed of a copper tube representing the house wall connected to a heating ring replacing the house heating system. The objective is to stabilize the air temperature inside a cylinder to any admissible temperature and reject the disturbance.

7.1.1 Instrumentation

- Sensors:
  - 1 air temperature sensor
  - 1 temperature sensor of the wall
- Actuators: The system is actuated by a warming ring and a fan.

7.2 Specifications

7.2.1 Basic requirements

Reach any higher temperature than the present temperature without overshoot nor steady state error and as fast as possible.

7.2.2 More advanced requirements

Using the fan as a controllable input, reach any temperature (higher or lower) without overshoot nor steady state error and as fast as possible.

7.3 Documentation

7.3.1 Heating - wall temperature static characteristic

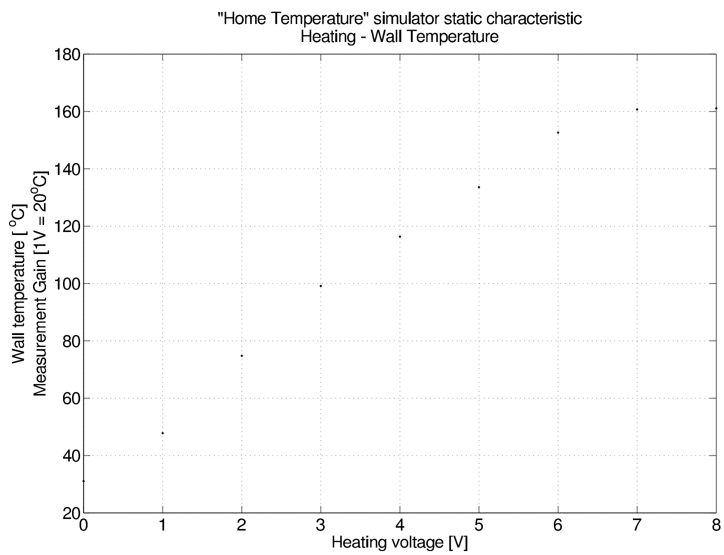


Figure 7.1: "Home Temperature" simulator: Heating - wall temperature static characteristic

Chapter 8

The Water Level Control

8.1 Plant description

This plant consists of two parts. The first one generates the water flow while the other one can collect the water in a tank or evacuate it. Both parts use a valve system. The objective is to achieve a certain water level in the tank by acting on the flow rate valve of the first part.

8.1.1 Instrumentation

- Sensors:
  - 1 flow rate sensor
  - 1 height sensor
- Actuators
  - valve of the first tank
  - valve of the second tank

8.2 Specifications

8.2.1 Basic requirements

Achieve a certain water level with a constant disturbance.

8.2.2 More advanced requirements

Achieve a certain water level with a variable disturbance.

Chapter 9

Two bars and one elastic

9.1 Plant description

This plant is composed of two bars mounted on two distinct DC motors and one elastic linking the two bars. The range of motion of the left arm (Motor 1) is 102° while it is 110° for the right arm (Motor2). Each arm is 150mm long. Two mechanical parts are also provided to block the arms in a definite position and calibrate the initial value of the encoders. The two motor shafts are distant of 277mm. The objective is to track any position trajectory.

9.1.1 Instrumentation

- Sensors: 2 encoders
- Actuator: The DC motors are current driven.

9.2 Specifications

9.2.1 Basic requirements

The objective is to reach any given constant position without overshoot nor steady state error. The response has to be as fast as possible.

9.2.2 More advanced requirements

- Set one motor to a constant position reference and track a sinusoidal reference with the other one.
- Synchronize the two motors to increase the bandwidth.

9.3 Documentation

9.3.1 Characteristics of the DC motors

Maxon RE motor with encoders: RE30 ; ϕ 30mm ; 24VDC ; 60W ; 369tr/min/V ; 25.9mNm/A

Chapter 10

The Centrifugal Ring Positioner

10.1 Plant description

A mobile hollow cylinder, the so-called ring, can slide on a tilted rod coupled to a motor. The objective is to play on the centrifugal force to control the ring position on the rod at an arbitrary setpoint.

10.1.1 Instrumentation

- Sensors:
  - Optical incremental encoder (velocity sensor)
  - Infrared sensor (position sensor)
- Actuator: The system rotates thanks to a current driven DC motor.

10.2 Specifications

10.2.1 Basic requirement

Stabilization of the ring at a fixed position on the rod with no (or the smallest possible) static error.

10.2.2 More advanced requirement

Track a sinusoidal ring position reference.

10.3 Documentation

10.3.1 Characteristics of the DC motor

Maxon RE25 gearmotor: ϕ 25mm, 24 VDC, 10W, 43.9mNm/A, reduction ratio (η_g) 35.

10.3.2 Plant parameters

- Tilt angle : 20°
- Ring mass : 0.0239 kg
- Ring friction coefficient: 0.3 kg/s
- Initial ring position : 8 cm
- Corresponding angular velocity : 6.89 rad/s

Chapter 11

The Rotary Inverted Pendulum

11.1 Plant description

The rotary inverted pendulum is composed of a rigid body driven by a motor around its z-axis and linked to a pendulum moving freely around its axis of rotation (the rotating arm). The main objective is to stabilize the pendulum at its unstable vertical position.

11.1.1 Instrumentation

- Sensors:
  - Incremental encoder (motor position sensor)
  - Incremental encoder (pendulum angle sensor)
- Actuator: The system is actuated by a current driven DC motor

11.2 Specifications

11.2.1 Basic requirements

Stabilization of the rotary inverted pendulum at its upward equilibrium point.

11.2.2 More advanced requirements

Stabilization of the rotary inverted pendulum at its upward equilibrium point at a specified motor angular position. Design a swing up strategy to lift the pendulum up.

11.3 Documentation

11.3.1 Mathematical model

From the Lagrange theory, we get:

$$\begin{cases} \ddot{\theta}\left(I_{a,z1} + I_{p,y2}\sin(\phi)^2 + m_p r^2\right) + \dot{\theta}\dot{\phi}\sin(2\phi)I_{p,y2} + \frac{1}{2}m_p lr\left(\ddot{\phi}\cos(\phi) - \dot{\phi}^2\sin(\phi)\right) + b_1\dot{\theta} = \tau_a \\ \ddot{\phi}I_{p,x2} + \frac{1}{2}m_p lr\ddot{\theta}\cos(\phi) - \frac{1}{2}\dot{\theta}^2 I_{p,y2}\sin(2\phi) - m_p g\frac{l}{2}\sin(\phi) + b_2\dot{\phi} = \tau_p \end{cases} \tag{11.1}$$

where I_α is the moment of inertia according to the axis α, r is the length of the rotating arm, l is the length of the pendulum, m_p is the mass of the pendulum, τ are the actuators (the notation $_a$ stands for the rotating arm and $_p$ for the pendulum)

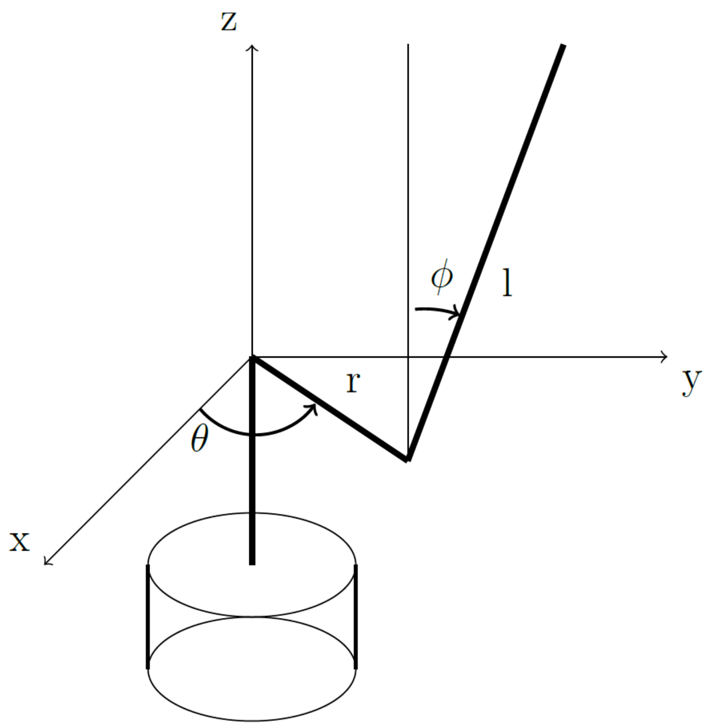
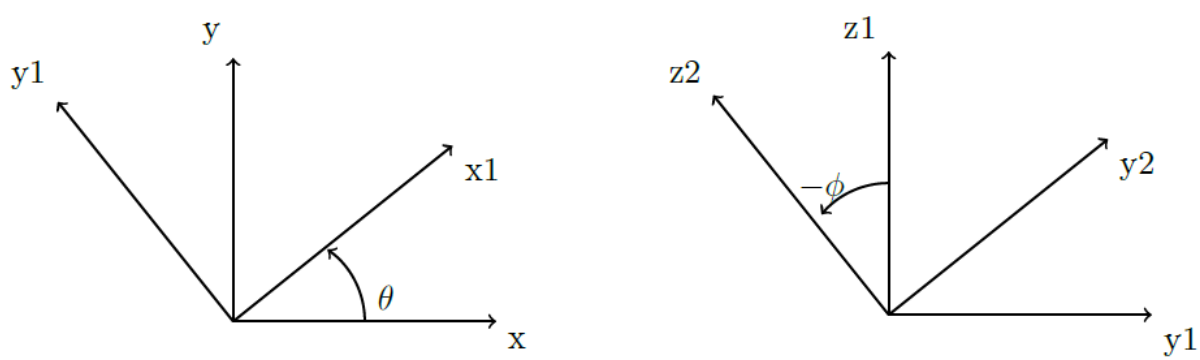


Figure 11.1: Rotary Inverted Pendulum



(a) Top view **(b)** Front view

Figure 11.2: Coordinate systems

Appendix A

System Modeling and Identification

Introduction

Given a certain *system*[1], modeling somehow attempt to characterize the link between the inputs and the outputs. The aim of modeling and identification is to find a link one may use in order to predict the output of a system by knowing only the inputs. This model can be used for simulation, model based controller design, ...

The present document addressed an introduction to modeling and identification. It will present only one structure of model namely the ARX model and only one method for the identification of the parameter of such a model.

A.1 Refresher

A.1.1 Response of linear time-invariant systems

Consider the system with a scalar input $u(t)$ and the scalar output $y(t)$. The system is linear time-invariant if:

linearity property if for a linear combination of inputs the output response is the same linear combination of the output responses of the individual inputs

time-invariant property if the response of a system to a certain signal do not depend on absolute time

Finally a a system is said to be causal if the output at a certain time only depends ont the inputs up to that time. Under all this conditions the response of the system can be described by:

$$y(t) = \int_0^\infty g(\tau)u(t-\tau)d\tau \tag{A.1}$$

where $g(\tau)$ is known as the impulse response. Consequently the impulse response of a linear time-invariant causal system completely defined its input/output response.

Taking the Laplace transforms $Y(s) = \mathcal{L}(y(t))$, $U(s) = \mathcal{L}(u(t))$ and $G(s) = \mathcal{L}(g(t))$, one obtains:

$$Y(s) = G(s)U(s) \tag{A.2}$$

$G(s)$ is also refered as the *transfer function* of the linear system (A.1).

[1]Not formally, a system is an object affected by external stimuli in which variables of different kinds interact and produce observable signals. The observable signals of interest are called *outputs*, while the stimuli are divided in two part: the *inputs* (possibly manipulated) and the *disturbances*

A.2 System modeling

A.2.1 Modeling of linear time-invariant systems

A common way of parameterizing G is to represent it as a rational function. A simple input-output relationship is obtained through an Ordinary Differential Equation (ODE):

$$a_0 y(t) + a_1 \frac{dy}{dt}(t) + a_2 \frac{d^2y}{dt^2}(t) + \cdots + a_{n_a} \frac{d^{n_a}y}{dt^{n_a}} = b_0 u(t) + b_1 \frac{du}{dt} + b_2 \frac{d^2u}{dt^2} + \cdots + b_{n_b} \frac{d^{n_b}u}{dt^{n_b}} \tag{A.3}$$

The parameters are in this case:

$$\theta = [a_0, a_1, \ldots, a_{n_a}, b_0, \ldots, b_{n_b}] \tag{A.4}$$

The corresponding transfert function is:

$$G_\theta = \frac{\sum_{i=0}^{n_b} b_i s^i}{\sum_{i=0}^{n_a} a_i s^i} \tag{A.5}$$

One way to predict the output $\hat{y}(t)$ is:

$$\hat{y}(t, \theta) = \mathcal{L}^{-1}(\hat{G}_\theta(s) U(s)) \tag{A.6}$$

A.3 Identification procedure

A.3.1 Parameter estimation: the least-squares method

Considering the sampled signals $u(k)$ (the input) and $y(k)$ (the output) (k = 1,2,. . . ,N, the sampling instant) and using the prediction (A.6), the *prediction error* is defined by:

$$\varepsilon(k, \theta) = y(k) - \hat{y}(k, \theta) \tag{A.7}$$

Using as cost function a quadratic function of the prediction error (A.8), the optimal parameters $\hat{\theta}^{LS}$ in the least square sense is defined by (A.9).

$$J^{LS}(\theta) = \sum_{k=1}^{N} \varepsilon^2(k, \theta) \tag{A.8}$$

$$\hat{\theta}^{LS} = argmin(J^{LS}(\theta)) \tag{A.9}$$

A.4 Example of identification using Matlab: program "getdataexample"

Lets consider the continuous-time system defined by its transfer function *sys*.

$$sys(s) = \frac{Y(s)}{U(s)} = \frac{1}{s+1} \tag{A.10}$$

The step response of such a system for $t \geq 0$ is (A.11) when the system is at rest for $t < 0$ ($y(0^-) = 0$)

$$y(t) = 1 - e^{-t} \tag{A.11}$$

If the signals are sampled at 2Hz here is the data:

$$\begin{aligned} y(k) &= [0, 0.3935, 0.6321, 0.7769, 0.8647, 0.9179, 0.9502, 0.9698, 0.9817, 0.9889, 0.9933, 0.9959, 0.9975] \\ u(k) &= [1, \ldots, 1] \end{aligned} \tag{A.12}$$

These datas are generated by the "lsim" matlab function for every example system "sys". The parameters of a selected transfert function structure are identified by the home made "IdentifySystem" function. This function is based on the procedure described before.

A.5 Deal with a real plant

Here is a common methodology for the identification of linear time-invariant model (LTI model):

1. Verify the properties of the plant.
2. Make one experiment to characterize the plant (at least)
3. Validate the result of the identification

A.5.1 Properties of the plant

The first step in order to identify a LTI model is to verify that the plant is indeed linear and time-invariant.

Linearity

It may be sometimes quite complicate to affirm that a plant is linear. The nonlinearities of a real plant can roughly be split in two categories: a static gain dependent of the input or a time response dependent of the input. To check the dependency of the static gain, one way is the *static characteristic* of the plant. For the dependency of the time response, one should check the dynamical response for different amplitude of input signals.

Time-invariance

The time-invariance is often assumed to be fulfilled but from time to time some experiment are required to assess the validity of this assumption.

A.5.2 Identification experiment

Books exist on the design of a experiment in order to perform an identification. The aim of this chapter isn't to present all the technics but only to informed about the basics properties of the experimental input.

First,one have to use a input signal who don't violate the LTI properties. Second, the input has to "fully" excite the plant. One way to achieve these properties is by using a step response between two values within the linear behavior of the plant.

A.5.3 Validation of the model

After running the identification procedure, one have to validate the results they obtain. The easiest way is by visually compare the response of the model of the plant with some real response (with the same inputs). One can also used different experiments for the identification and the validation, in this case you performed *cross validation*.