**MEEC - Mestrado em Engenharia Electrotécnica e de Computadores**

# Controlo Multivariável, Não-linear e Ótimo (CMN-O)

**2022/2023**

## Lab project

## *Linear State Feedback Control*

### *of an Inverted Pendulum*

**J. Miranda Lemos**     and     **José A. Gaspar**

**DEEC, Área Científica de Sistemas, Decisão e Controlo**

✿

The plant model, the design of the linearized controller and its implementation in SIMULINK has been done by **Samuel Balula**, in the framework of his M. Sc. dissertation, under the supervision of Prof. J. Miranda Lemos. Prof. **João Pedro Gomes** contributed with software for the interface between the optical encoder and the data acquisition boards.

Contact: João Miranda Lemos, jlml@inesc-id.pt

## Objectives

After completing this work, the student should be able to:

1. Design a state feedback controller for a linear system described by a state model, including a state observer.
2. Test the design through simulations performed in SIMULINK.
3. Test the design in the real system using SIMULINK and replacing the block that simulates the plant (inverted pendulum) by blocks connected to A/D and D/A converters that interconnect SIMULINK computations with the physical system in real time.

Therefore, this work illustrates a situation of **fast prototyping**, in which a simple **cyber-physical system** (a system with interacting physical and computational parts) is created.

In this example, the interconnection between the "cyber" and the physical" parts of the plant to achieve its objective (equilibrate the pendulum) is striking. Indeed, the pendulum would not stand up if the "cyber" part (the controller) is not working properly.

## Bibliography

- Course slides. *Available in Fénix, at the course web page.*
- João Miranda Lemos (2019). *Controlo no Espaço de Estados,* cap. 5.
- Franklin, Powell and Emami-Naeini. *Feedback Control of Dynamic Systems,* Pearson/Addison-Wesley (several editions). Chapter 7.
- Samulel Balula (2016). *Nonlinear control of an inverted pendulum.* M. Sc. Thesis, IST, Universidade de Lisboa. *Available in Fénix, at the course web page.*

## Work to perform

After completing the work, each group of students must deliver a report with the answer to the questions indicated below. This report must contain:

- The identification of the work
- The identification of the students (number and name)
- The answer to the questions indicated below, identified by their number. The answers must be concise.

By delivering the report the students confirm that it is original and corresponds to the work actually done by the students who sign it. Plagiarism will cause the grade to be zero, independently of the sanctions determined by the applicable law and IST/Univ. of Lisbon regulations.

## Lab session planning

- Previous to session 1 ("home" preparation):
  - Follow a tutorial to learn the basics of MATLAB and SIMULINK .
  - Write a MATLAB macro to design the controller (compute the controller and observer gains) and compute the step and frequency responses of the controlled plants (see this guide below).
  - Write a SIMULINK design to test the controlled system in simulation.
  - Perform simulations using different design parameters.
- **Session 1**: Finalize the work you have done at "home" for the design of the controller and try different designs in simulation. Document your results. Get insight on the impact of design options on performance.
- **Session 2**: using a given SIMULINK block diagram and a MATLAB macro file, replace the controller and observer gains with the ones you have previously computed and test the controller on the real system
- **Session 3**: Try with different designs. Document your work.

Each lab session lasts 3 hours.

## Description of the plant to control

The plant to control, shown in fig. 1, consists of an inverted pendulum, connected at the base to an horizontal bar that can be rotated in the horizontal plane by a servo motor with a vertical axis. This plant is manufactured by QUANSER.



Fig. 1 – The plant to control: QUANSER rotary inverted pendulum.

The plant has sensors that measure the angle of the motor shaft with respect to a reference direction (a potentiometer connected to the shaft), and the angle of the pendulum with respect to the vertical (an optical encoder).

The manipulated variable is the tension $u$ applied to the motor through a power amplifier, and the process output to control, $y$, is the rotating angle of the pendulum with respect to the vertical axis.

## The control objective

The control objective consists of moving the basis of the pendulum, using the motor, such that the upper tip of the pendulum is equilibrated in the upper-pointing position.

## Plant model

The plant is nonlinear. If you are curious, you may find a nonlinear model of it in chapter 2 of Samuel Balula's thesis (available in Fénix), although this model is not required for the present work.

An equilibrium point consists of the pendulum pointing upwards, with all the variables still. This equilibrium corresponds to all the variables being zero. If the variables denote increments with respect to this equilibrium, a linearized model that represents the plant with reasonably good accuracy is given by
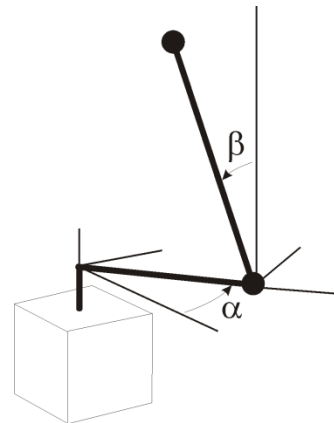
$$\dot{x} = Ax + Bu, \tag{1}$$

$$y(t) = Cx(t) + Du(t). \tag{2}$$

You can find matrices $A, B, C$ and $D$ in the MATLAB data file fpmod1.mat

The state is the vector of dimension 5 given by (see figure)

$$x = \begin{bmatrix} \alpha \\ \dot{\alpha} \\ \beta \\ \dot{\beta} \\ i \end{bmatrix}$$

in which $\alpha$ is the angle of the horizontal bar, $\beta$ is the angle of the pendulum with respect to the vertical (the variable that we want to control), and $i$ is the motor current.

## Work to perform ( 🏠 to do at home; 🛠 to do at the lab)

**1.** 🏠 Characterize the model. What are the eigenvalues of matrix $A$? What can you say about these eigenvalues? *Suggestion:* The matrices $A, B, C$ and $D$ can be obtained by loading file fp_lin_matrices_fit3.mat For that sake use the command ==load fp_lin_matrices_fit3.mat== . To know which variables are in the MATLAB workspace use the command ==who== . Use the MATLAB function ==eig== to compute the eigenvalues of matrix $A$.

**2.** 🏠Characterize the open loop system in terms of controllability. The relevance of this check stems from the fact that, if the system is controllable, then we can design a state feedback controller such that the closed-loop eigenvalues can be placed anywhere in the complex plane. The pair $(A, B)$ is controllable if the rank of the controllability matrix is equal to the dimension of the state. Thus, to check that the model is controllable, use MATLAB macro to compute the controllability matrix. *Suggestion*: Use the function *ctrb* to find the controllability matrix, and the function *rank* to to find the rank (number of linearly independent rows or columns) of a matrix.

**3.** 🏠 Characterize the open loop system in terms of observability. A state realization is observable if the observability mtrix has ran equal to the dimension of the state. Start by considering that the only output is the pendulum angle $(x_3)$. In this case, what is the matrix $C$? Repeat then the study for the case in which you measure $x_1$ and $x_3$. For that purpose write a MATLAB macro to compute the observability matrix and to find its rank. Use the function *obsv* to compute the observability matrix.

**4.** 🏠 Write a MATLAB macro to plot the Bode diagram of the open loop system in a range of frequencies that you find convenient. Comment the diagram shape considering the model. Suggestion: Use the MATLAB functions *ss2tf* to obtain the transfer function from the state model and *bode* to plot the Bode diagrams. Use the function *bode* to plot the frequency response.

**5.** 🏠 Write a MATLAB macro to find the vector of gains of the controller. In this phase of the design process assume that all the components of the state are available. The state feedback controller is defined by

$$u(t) = -Kx(t). \tag{3}$$

When coupled with the plant model (1), the control law (3) yields the closed-loop model

$$\dot{x} = (A - BK)x. \tag{4}$$

If the pair $(A, B)$ is controllable (a condition checked in point 2 above), then, it is always possible to find the vector of controller gains $K$ such that the closed dynamics $A - BK$ has its eigenvalues at the specified eigenvalues, wherever they might be. Therefore, to design the controller, one approach might be to specify the closed-loop eigenvalues and then compute $K$. However, in this work, we follow a different (although linked) road, in which the controller is optimized in a systematic way. Thus, we compute $K$ such as to minimize the quadratic cost

$$J = \int_0^{\infty} (x^T Q_r x + R_r u^2) dt. \tag{5}$$

The solution to the problem of minimizing $J$ defined by (5) amounts to finding the positive definite matrix $P$ that verifies the Algebraic Riccati equation (ARE)

$$A^T P + PA^T - PBR_r^{-1}B^T P + Q_r = 0, \tag{6}$$

and then computing $K$ from

$$K = R_r^{-1}BP. \tag{7}$$

These computations can be readily made in MATLAB using the function *lqr*. Use this function to compute the state feedback vector of gains $K$. Try different values for $Q$ (a positive semidefinite matrix) and $R$ (a positive scalar).

You may compute the closed loop poles that result from a given design by computing the eigenvalues of $A - BK$. Use the MATLAB function *eig* for that.

**6.** 🏠 Build a SIMULINK block diagram to simulate the controlled system.
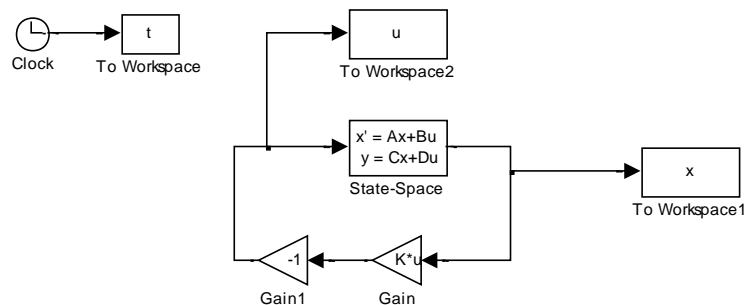


Figure 2 – SIMULINK block diagram to test the state feedback controller.

```
% Furuta pendulum - State feedback test
%_____

load('fp_lin_matrices_fit3.mat'); %%Load Matrices A, B, C, D

Qr = diag([10,0,1,0,0]); %Weight Matrix for x
Rr = 1; %Weight for the input variable

K = lqr(A, B, Qr, Rr); %Calculate feedback gain

%----------------------------------------------------------------------
% Simulate controller

x0=[0.1 0 0 0 0]';
D=[0 0 0 0 0]';
T=2; % Time duration of the simulation

sim('statefdbk',T);

gg=plot(t,x);
set(gg,'LineWidth',1.5)
gg=xlabel('Time (s9');
set(gg,'Fontsize',14);
gg=ylabel('\beta (rad)');
set(gg,'Fontsize',14);

%----------------------------------------------------------------------
% End of file
```

Figure 3 – A MATLAB macro to perform simulations with the SIMULINK block diagram of figure 2 to test the controller gains

Figure 2 shows a block diagram used to test the controller gains. This block diagram is called by the macro shown in figure 3, that also contains the call to function lqr, used to compute $K$. It is remarked that, since the reference is zero, the excitation must come from the initial conditions that must be non-zero.


**7.** 🏠 Write a MATLAB macro to find the vector of gains of the observer (sate estimator). The observer is a piece of software that builds an estimate $\hat{x}$ of the state $x$. It consists of a replica of the plant model excited by the same input $u$, plus the difference between what one expects the plant output y to be (given by $C\hat{x}$) and the actually observed output, $y$. In mathematical terms, the observer is defined by the differential equation

$$\dot{\hat{x}} = A\hat{x} + Bu + L(y - C\hat{x}) \tag{8}$$

The observer adds poles to the closed-loop. These poles are the eigenvalues of $A - LC$.

```
G = eye(size(A)); %Gain of the process noise
Qe = eye(size(A))*10; %Variance of process errors
Re = eye(2); %Variance of measurement errors

L = lqe(A, G, C, Qe, Re); %Calculate estimator gains
```

Figure 4 – Software to design the vector of observer gains

The vector of observer gains is designed such that the estimation error converges to zero. A way to do this is to employ the Kalman filter, for which the gain is computed using the MATLAB function *lqe*. Figure 4 shows the code to use this function.

**8.** Build a SIMULINK block diagram to simulate the controlled system, using the controller and observer that you have designed. Show the simulation results that you have obtained.
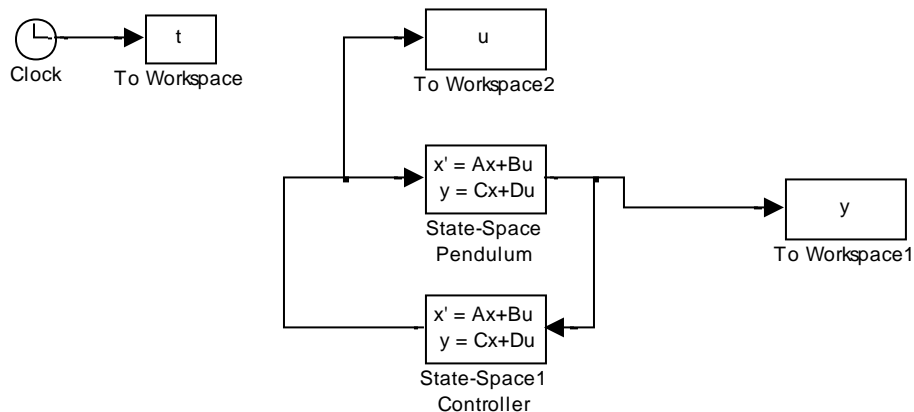


Figure 5 – Interconnection of plant and controller, represented by state models.

When using the observer, the feedback is not from the state (assumed to be unavailable for measure) but from its estimate, according to

$$u(t) = -K\hat{x}(t). \tag{9}$$

Combining the observer equation (8), and the control equation (9), it is possible to write the following state model for the controller

$$\dot{\hat{x}} = \hat{x}(A - BK - LC) + Ly, \tag{10}$$

$$u(t) = -K\hat{x}(t). \tag{11}$$

According to (10,11), the controller is represented by a state model with input $y$ (the plant output), and output $u$ (the plant input). Therefore, the new dynamics matrix is $A - BK - LC$ (the controller "$A$" matrix), the input matrix is $L$ (the controller "$B$" matrix), and the output equation is $-K$ (the "$C$" matrix of the controller). We may therefore use the state-space block of SIMULINK to define the controller in a compact way, as in figure 5.

**9. ✂ Lab session 1**. Simulate the controller and simulation set-up that you have developed at "home" (points 1 to 8). Try different weights for the cost functions and compare the results obtained. Explain how you evaluate the performance of your controller in a systematic (numeric) way. If you have time, you may start with the experiments on the real pendulum (see point 10 below).
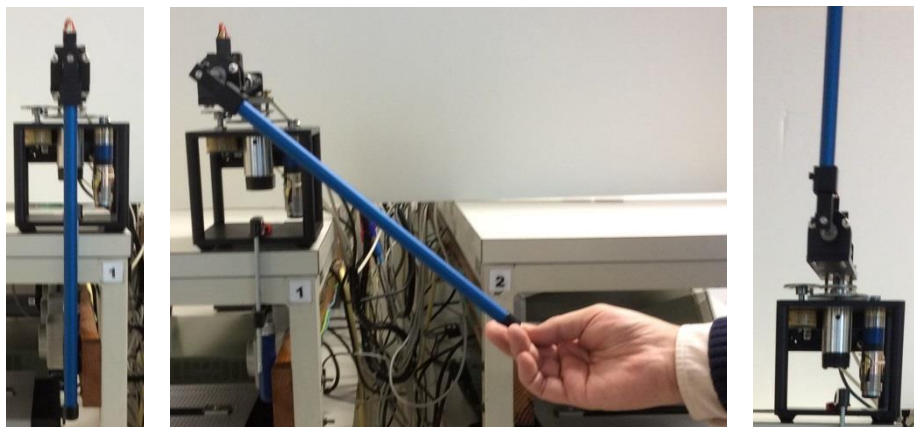


Figure 6 – Starting the experiments.

**10.** ⚒ **Lab session 2**. Using the gains that you have computed above, conduct now a test on the real system. For that sake, perform the following steps:

- Enter MATLAB (look for the icon of MATLAB 2015a in the desktop), and in the MATLAB command line write the command *login_cee*. This command will place MATLAB in a directory of your own, where you can work safely, without destroying any other important software.

- Check what is the data acquisition board your computer is equipped with. For that sake, look on the back side of the computer on your bench, and locate the data acquisition board.
  - If the connection is metallic, with a silver color, then the board is the NI-PCI6221
  - If the connection is plastic, with blue color, then the board is the NI-MIO16E4

- Look at the software you have available in your directory. You have two m-files (that contain the main and 2 slx-files. You should use the ones that correspond to the type of data acquisition board that you have in your work bench.

- Using the MATLAB editor, edit the m-file that corresponds to your board. Observe that it has an instruction to load the file that contains the plant model, then a few lines that configure parameters used by the data acquisition board, and finally a block of lines to design the controller and the observer. You may change the cost weights (for the controller, and for the observer) in order to change the design. Run this macro file.

- Activate now the SIMULINK block diagram. To conduct an experiment proceed as follows:
  - Place your pendulum in the vertical position, and such that it is perfectly still, as in the left picture of figure 6.
  - Click the button ▸ to start the experiment. Wait for SIMULINK to compile the block. When the time starts counting, wait 1 second.

The sensors are calibrated and you now have 6 seconds to gently raise the pendulum to the vertical upwards position (as in figure 6 center) and keep it there. Then the controller will start working and you may remove your hand while the pendulum is equilibrated automatically (figure 6 right). Use the button ■ to stop the program or (preferably) wait for the time-out. See figure 7 to locate the buttons. Remark that there are start/stop buittons for MATLAB and for SIMULINK. Don't confuse them.

Observe and register the results. Compare the time responses obtained using the real system with the ones obtained in simulation. Comment the differences.
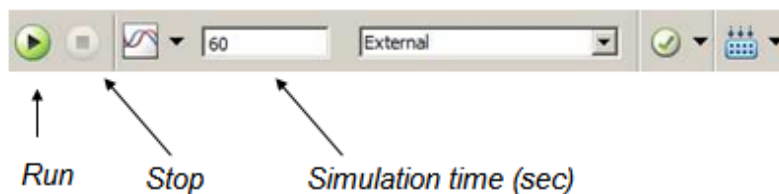


Fig. 7 – Buttons in the SIMULINK project.

**11. �֎ Lab session 3**. Try other weights for the observer and/or controller. Register and comment your results. Try to obtain the best design (how do you define it?). Compare the different options you have considered using an objective figure of merit, such as the tracking error mean square value or other. How can you to improve the results by changing the controller in order to make the angle $\alpha$ stand still?