TUHH
Technische Universität Hamburg

ICS
Institute of Control Systems

Control Lab
CSTD2

Identification and Control of a Magnetic Levitation Plant

13th November, 2020

Room: Online | N-1.077 Winter Semester 2020/21

Contents

About this Document

This document is to help you with the design task *CSTD2—Identification and Control of a Magnetic Levitation System* of the *Control Lab* practical course. It is written mainly in the style of a tutorial and should provide you with all the necessary tools and Matlab commands to solve the task.

This document is accompanied by Matlab files that you need to modify and execute in order to develop your own design.

`cstd2_ident.m` This file deals with the identification of the system.

! You need to complete the code on your own and submit a published version

of this file (Use the `publish` command) no later than one week before the scheduled date for the experiment.

`d_step.mat` , `d_chirp_1.mat` , `d_chirp_2.mat`, `m_prbs.m` , `m_noise.m` These files contain measurements with different types of excitation signals.

`cstd2_design.m` This file deals with the controller design and the analysis of the controller performance.

! You need to complete the code on your own and submit a published version of this file (Use the `publish` command) no later than one week before the scheduled date for the experiment.

`cstd2_sim_ff.slx` , `cstd2_sim_lqg.slx` , `cstd2_sim_lqg_int.slx` These models should be used to simulate the different controller structures.

! You need to complete some of these models to successfully run the `cstd2_design.m` file.

There are also several other auxiliary files provided. Just make sure that you add the `simulink` folder and all of its subfolders to your MATLAB path. The code is guaranteed to work with Matlab 2016b 64bit, other versions might not be supported. You can get the latest Matlab version from `https://www.tuhh.de/rzt/usc/matlab/index.html` or use the pool computers.

In this document, you will encounter blocks that indicate Matlab code:

```
[MATLAB COMMANDS]=USEFUL(TOOLS)
```

These are meant to get you started. You can (and should) use the `help` command within Matlab to find out more about a particular command.

Another thing that you will encounter are preparations and tasks:

Preparation: Review the concept of state feedback in the *Control Systems Theory and Design* lecture notes.

! These are meant to prepare you for the question session that will take place prior to conducting the experiment.

Task: Read this document carefully.

! These are meant to guide you through the tasks which needed to be completed in order to be uploaded.

Task

Identify a state space model and design a controller for the magnetic levitation plant and evaluate it in simulation. Attach all necessary MATLAB and SIMULINK files that were provided to you no later than one week before the experiment via email to the responsible Tutor and Supervisor. You will get an email when your preparation is not sufficient to pass the Lab and will get time to revise your design.

Checklist

☐ I read this whole document carefully.

☐ I did all preparation tasks and can explain them.

☐ I completed the file `cstd2_ident.m`.

☐ I completed the file `cstd2_design.m`.

☐ I submitted all MATLAB files.

1 MAGNETIC LEVITATION PLANT

The plant, shown in Fig. 1, consists of upper and lower drive coils that produce a magnetic field in response to a DC current. One or two magnetic disks travel along a precision ground glass guide rod. By energising the lower coil, a single magnetic disc is levitated through a repulsive magnetic force. As the current in the coil increases, the field strength increases and the levitated magnet's height is increased. For the upper coil, the levitating force is attractive. Two magnetic discs can be controlled simultaneously by stacking them on the glass rod. The magnets are of an ultra-high field strength rare earth type and are designed to provide large levitated displacements.

Two laser-based sensors measure the magnet's positions. The lower sensor is used to measure a magnet's position in proximity (8 cm range) to the lower coil, and the upper one for proximity to the upper coil.

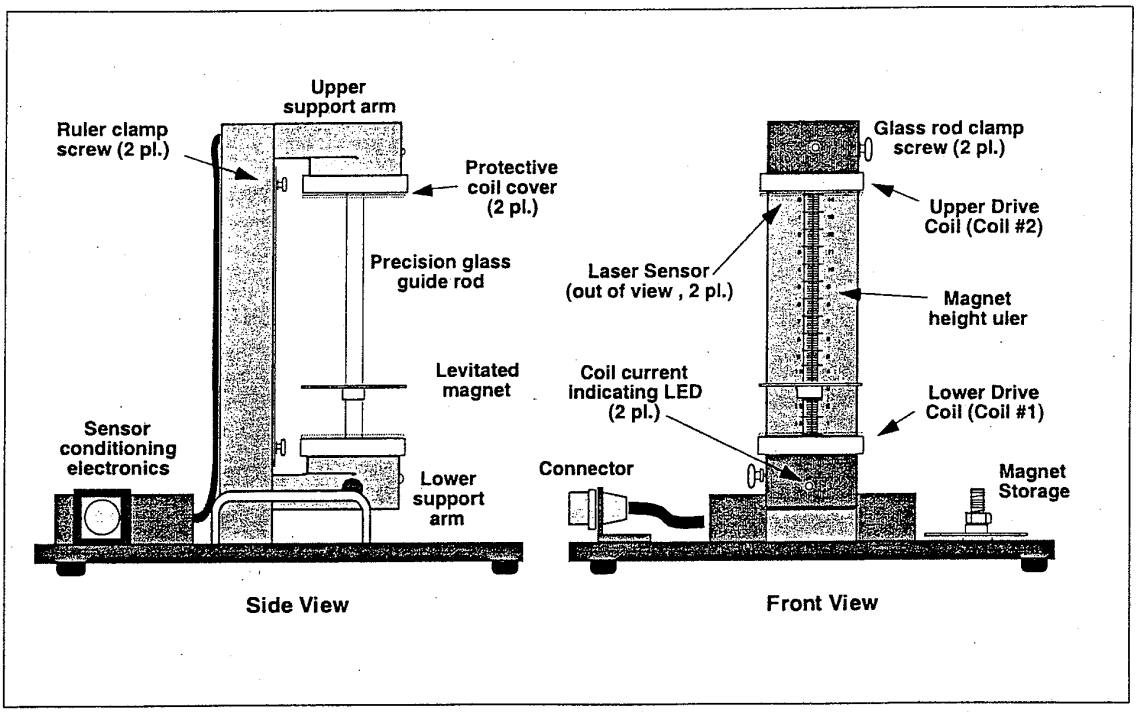

Figure 1: Magnetic Levitation Apparatus

The objective of the lab project is to control the position of the two magnetic disks independently. Thus the measured disk positions are the controlled outputs, and the currents in the two coils are the control inputs. Since the field interaction between the two magnets causes strong cross coupling, controlling the disk positions independently requires the design of a MIMO controller. In this project an LQG controller will be designed for this purpose.

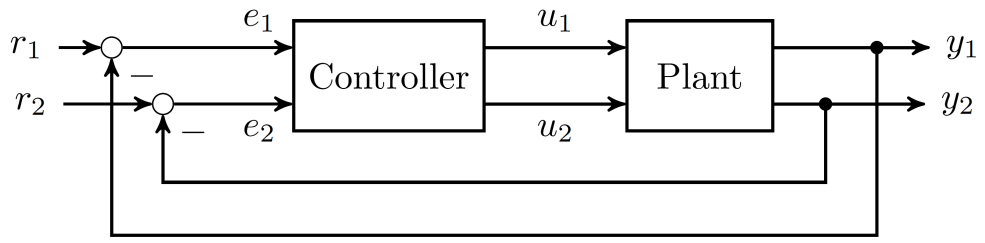

Figure 2: Block diagram of the closed-loop system

Referring to Fig. 2, the plant inputs u_1 and u_2 are the currents in the upper and lower coil, respectively, while y_1 and y_2 denote the positions oft he upper and lower coil.

Rather than deriving a dynamic model of this plant from first principles, a linear black box model will be identified using subspace identification tools. The magnetic field nonlinearities are compensated using nonlinear static inversion, such that the compensated system can be approximated by a linear state space model with sufficient accuracy. The controller design will be carried out in continuous-time domain, but the controller will be implemented digitally.

1.1 Control Objectives

The requirements for the controller are as follows:

1. Disk positions should follow the trajectories shown in Fig. 3 without a steady state error.
2. No overshoot in the responses.
3. Rise time of 0.3 s.
4. Cross coupling requirements:
   - at a step change on $y_2(t)$, $|e_1(t)| \leq 0.15\,\text{cm}$
   - at a step change on $y_1(t)$, $|e_2(t)| \leq 0.2\,\text{cm}$

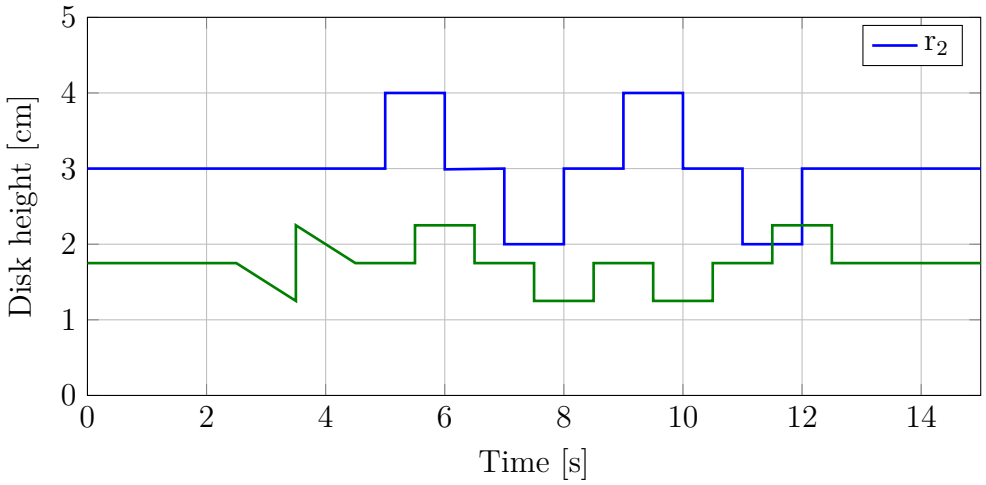

Figure 3: Reference trajectories, also given in `cstd2_sim_model.slx`

2 SYSTEM IDENTIFICATION

A linear multivariable state space model of the plant with two inputs and two outputs is to be identified experimentally. Refer to the lecture notes *Control Systems Theory and Design* Section 7.3 for the identification of state space models using subspace techniques. A requirement for system identification in open loop is that the plant to be identified is open-loop stable.

Prep. 2.1: From the physical plant description and the definition of input and output signals, would you expect the linearized plant model to be stable or unstable? Why?

2.1 Choice of the Identification Signal

Fig. 4 shows the response to step inputs on u_1 and u_2.

Prep. 2.2: What closed-loop bandwidth would we – roughly – expect to achieve, and in what frequency range should the input signal used in the identification experiment excite the plant?

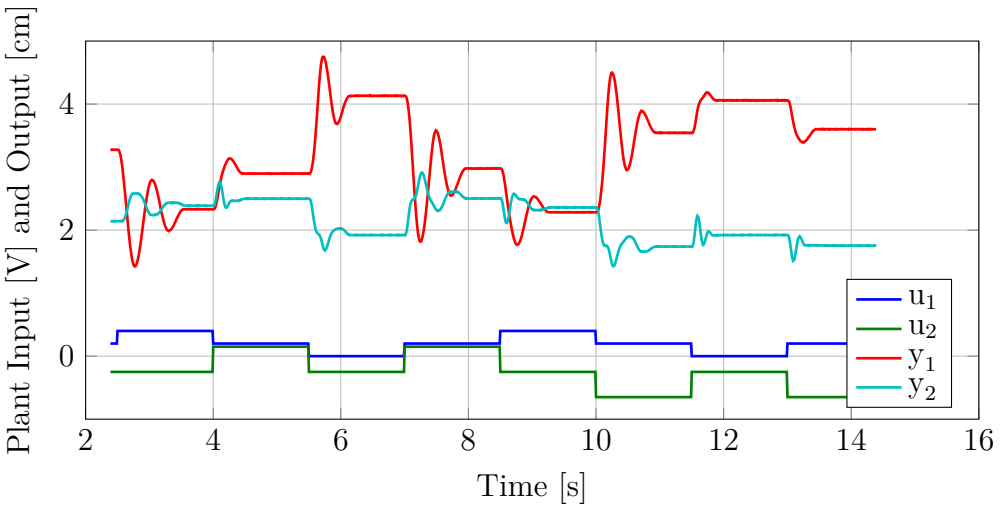

Figure 4: Step inputs on u_1 and u_2 and the open-loop response (`d_steps.mat`)

Prep. 2.3: What type of input signal do you think is suitable to be used in the identification experiment?

Prep. 2.4: Review how input signals for identification experiments are generated using the `idinput` function (Use the `help` command if required).

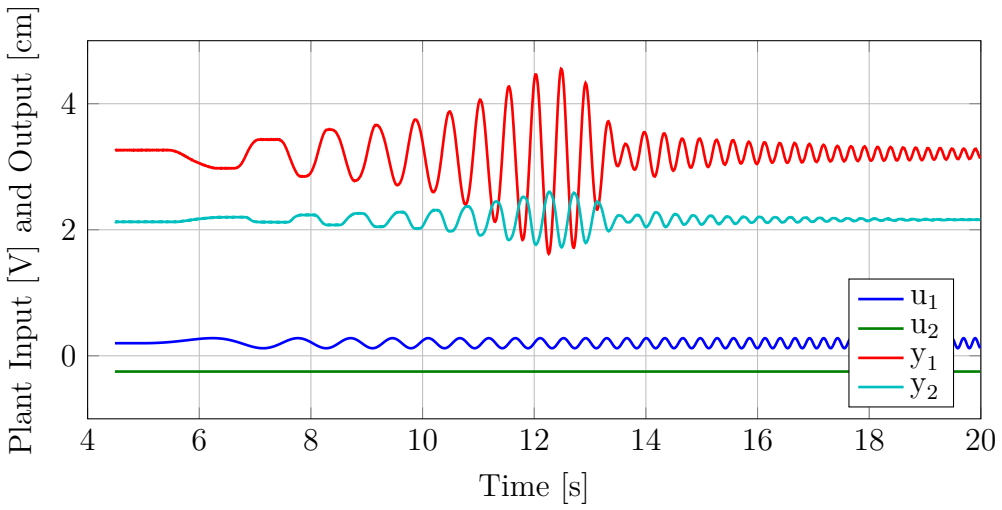

Figure 5: Chirp signal on u_1 and the open-loop response (`d_chirp.mat`)

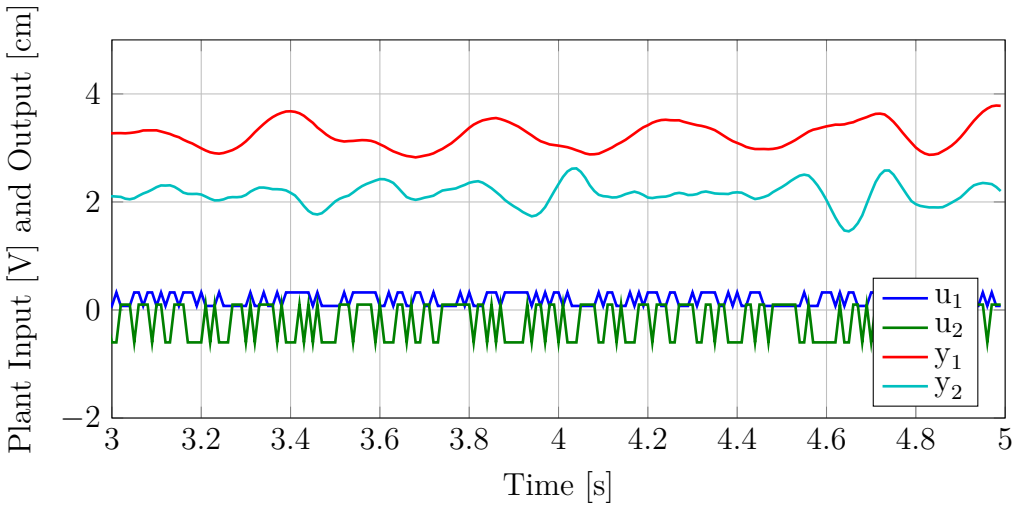

Figure 6: Pseudorandom binary sequence on u_1 and u_2 and the open-loop response (`d_prbs.mat`)

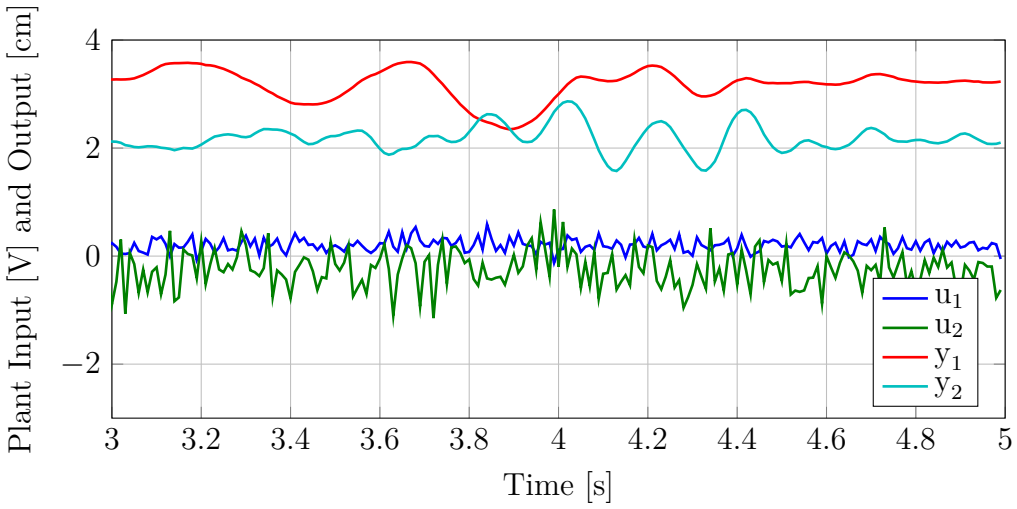

Figure 7: Band limited white noise on u_1 and u_2 and the open-loop response (`m_noise.mat`)

2.2 Subspace Identification of the State Space Model

In this section, the method for identifying state space models is presented shortly. To introduce the idea, we begin with a SISO state space model:

$$\begin{aligned} x(k+1) &= \Phi x(k) + \Gamma u(k) \qquad & x(0) = 0 \\ y(k) &= cx(k) + du(k) \end{aligned}$$

Now assume that $x(0) = 0$, and consider the impulse response of the above model, i.e. the response to the input $u(k) = \delta(k)$.
This yields $x(k) = \Phi^{(k-1)}\Gamma$ and the impulse response $g(k)$ is given by:

$$g(k) = \begin{cases} 0, & k < 0 \\ d, & k = 0 \\ c\Phi^{k-1}\Gamma, & k > 0 \end{cases} \tag{1}$$

For a MIMO model, the matrices Φ, Γ, c and d increase in size, and we can apply a unit impulse to one input channel at a time and observe the resulting response at each output

channel:

$$u_{\delta i}(k) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ . \\ . \\ \delta(k) \\ 0 \\ 0 \\ . \\ . \end{pmatrix} \qquad y_{\delta i}(k) = \begin{pmatrix} g_{1i}(k) \\ . \\ . \\ . \\ . \\ . \\ g_{li}(k) \end{pmatrix}$$

An entry $g_{ji}(k)$ in the output vector represents the response at output channel j to a unit impulse applied at input channel i.
Now, assume that measured impulse response data of a system are available and have been arranged in the form of a matrix:

$$H = \begin{pmatrix} g(1) & g(2)... & g(k) \\ g(2) & g(3)... & g(k+1) \\ g(3) & .... & ... \\ . & . & . \\ g(k) & g(k+1) & ...g(2k-1) \end{pmatrix} \tag{2}$$

The H matrix is known as the "Hankel Matrix". Assume the number of samples is sufficiently large so that k > n, where n is the order of the state space model. One can observe that:

$$H_k = \begin{pmatrix} C \\ C\Phi \\ . \\ . \\ . \\ C\Phi^{k-1} \end{pmatrix} \begin{pmatrix} \Gamma & \Phi\Gamma & ... & \Phi^{k-1}\Gamma \end{pmatrix} = O_k C_k \tag{3}$$

Where O_k and C_k are the extended observability and controllability matrices respectively, and $\mathbf{rank}O_k = \mathbf{rank}C_k = \mathbf{rank}H_k = n$. **Define** $\bar{O_k} = O_k\Phi$, this implies that one can compute:

$$\Phi = (O_k^T O_k)^{-1} O_k^T \bar{O}_k \tag{4}$$

2.3 Direct Identification

Consider the system given by:

$$Y_k = O_\alpha x(k) + \Psi_\alpha U_k \tag{5}$$

where: $O_\alpha = \begin{pmatrix} C \\ C\Phi \\ C\Phi \\ . \\ . \\ . \\ C\Phi^{\alpha-1} \end{pmatrix}$ and $\Psi_\alpha = \begin{pmatrix} D & 0 & 0 & ..... & 0 \\ C\Gamma & D & 0 & .... & 0 \\ C\Phi\Gamma & C\Gamma & D & 0 & ..0 \\ .. & .. & .. & .. & .. \\ C\Phi^{\alpha-2}\Gamma & C\Phi^{\alpha-1}\Gamma & ... & ... & D \end{pmatrix}$

Assume that a sufficient number of measurements has been collected so that we can form the input and output data matrices:
$\underline{\mathrm{Y}} = \begin{pmatrix} Y_1 & Y_2 & Y_3 & ... & Y_N \end{pmatrix}$ and $\underline{\mathrm{U}} = \begin{pmatrix} U_1 & U_2 & U_3 & ... & U_N \end{pmatrix}$
The data satisfies:

$$\underline{Y} = O_\alpha X + \Psi_\alpha \underline{U} \tag{6}$$

Now define the matrix

$$\Pi = I - \underline{U^T}(\underline{UU^T})^{-1}\underline{U} \tag{7}$$

This yields:

$$\underline{Y}\Pi = O_\alpha X \Pi \tag{8}$$

and hence the right hand side term can be computed. Using the "Singular Value Decomposition" method, the matrix O_α is obtained. Substituting back into the model,

the matrix Ψ_α can be determined from:

$$\Phi = (O_k^T O_k)^{-1} O_k^T \bar{O}_k \tag{9}$$

Moreover, the elements of these two matrices, which are the matrices D, Γ and C can be found by solving a system of linear equations, relating the known inputs to the outputs collected.

2.4 Application

The estimation of the levitation plant model is done with subspace identification of the state space model. MATLAB provides an identification toolbox that is used. To allow you to prepare yourself for identifying a model experimentally in the lab, here we provide four different sets of data, obtained in identification experiments with different types of input signals (step, chirp, prbs, bandlimited white noise, shown in Fig. 5-7).

2.4.1 Step 1: Input Data

Step 1.1: Open the identification toolbox.(Fig.8)

To estimate the model, a sufficient number of inputs and outputs should be gathered. The number of samples must be sufficiently large so that k > n, where n is the order of the state space model. For this experiment, the data is provided.

Step 1.2: Import the data using "Import data" then press "data object" and write any of the generated data objects from the m-file.

It is also important to preprocess the data in order to obtain good estimations. The above procedure for identifying a state space model relies on the assumption that the measured data were indeed generated by a linear system and are not corrupted by measurement noise. In practice, neither assumption is true. One consequence of this is that no matter how large k is chosen, the matrix H_k (Hankel Matrix) usually has full rank. To extract information about the model order in spite of data being corrupted, one can use the technique of singular value decomposition (discussed in chapter 7 of the lecture notes).

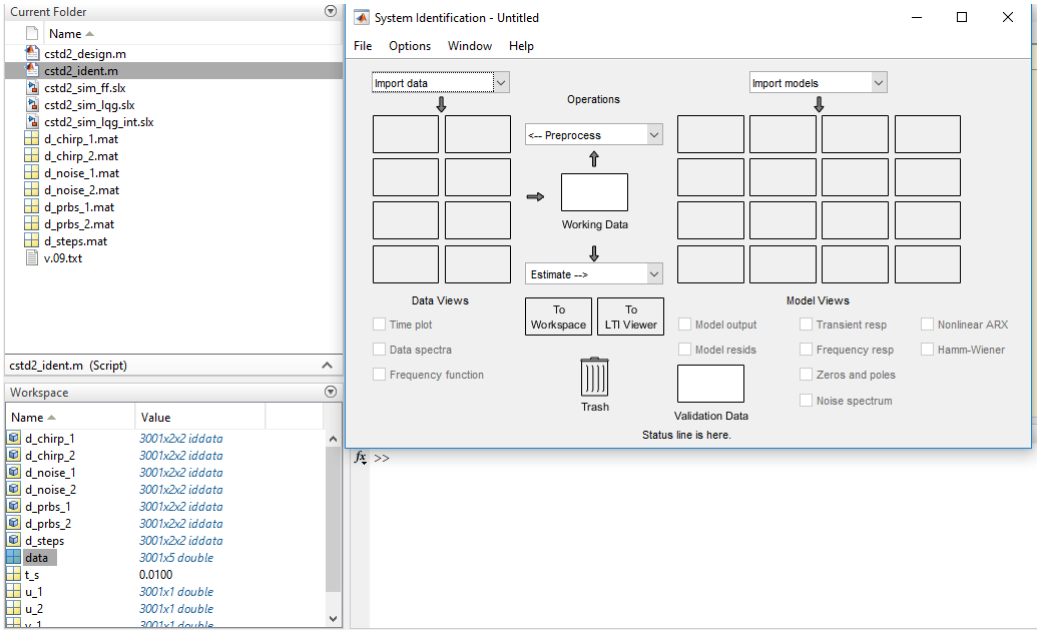

Figure 8: Identification toolbox in MATLAB

Therefore, the data must be filtered of noise.

Step 1.3: Use the preprocess method in the identification toolbox (Fig.9), then select ”remove means” and ”remove trends”. Removing means removes any offset of the measured values, and removing trends estimates any linear trend and removes it. The reason the mean values from each signal are subtracted is because, typically, linear models are built so that they describe the responses for deviations from a physical equilibrium. With steady-state data, it is reasonable to assume that the mean levels of the signals correspond to such an equilibrium. Thus, seeking models around zero without modeling the absolute equilibrium levels in physical units is possible. Note that in between each step the new generated data has to be dragged and dropped to the “working data” section.

Step 1.4: Partition the data in two ranges: 0-15 secs and 15-30 secs.The first range is used for identification and the second range for validation. Other preprocessing schemes such as filtering may be tried as well.

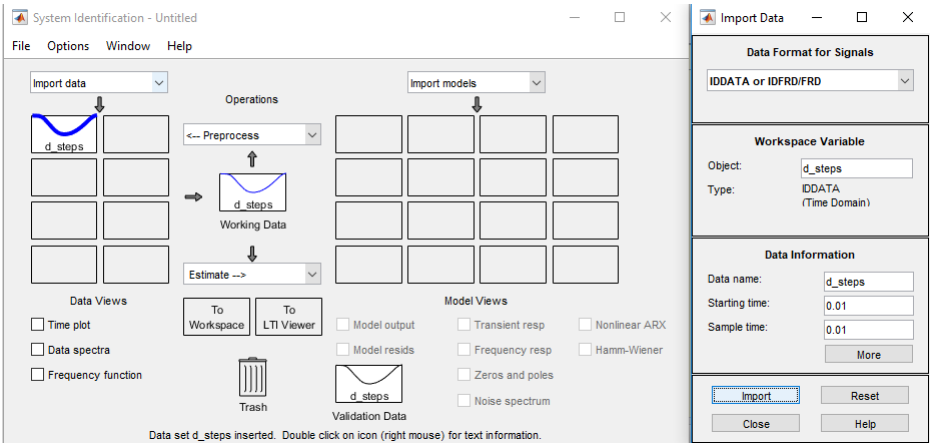

Figure 9: Importing of data

2.4.2 Step 2: Estimation of the Model

Next, the model is to be estimated. The identification toolbox builds the Hankel Matrix (equ. 3) mentioned earlier using singular value decomposition. The method states that the order of a system is the number of non zero singular values. Since the number of measured data is large, the result is guaranteed to yield a number of singular values which is higher than the order of the system. An example is a singular value decomposition that yields the following: $\sigma_1 > \sigma_2 > ... > \sigma_n > \sigma_{n+1} = \sigma_{n+2} = .. = \sigma_p = 0$. The order of the system is "n". However, it can also happen that the singular values of order larger than n are not zero but close to zero. The order here is chosen when a large change in magnitude occur, and the small close to 0 values are due to noise effect.

Step 2.1: In the system identification window, press estimate, then choose state space model. In the new window that shows, choose "pick best value in the range[1:10]", to limit the singular values shown since the correct order of the model is unknown, then tick the "continuous time" button which informs the toolbox that the input data is a continuous time data, and set "Simulation" for "focus" which optimizes the model to use for output simulation. Note that the estimation method is set by default to Subspace identification, which is the method discussed earlier (Fig.11).
A new window opens showing the singular value plot for the Hankel Matrix (Fig.12). The figure shows a drop in 2 magnitudes between the 4^{th} and 5^{th} singular values. This means that the matrix becomes rank deficient at order 5.

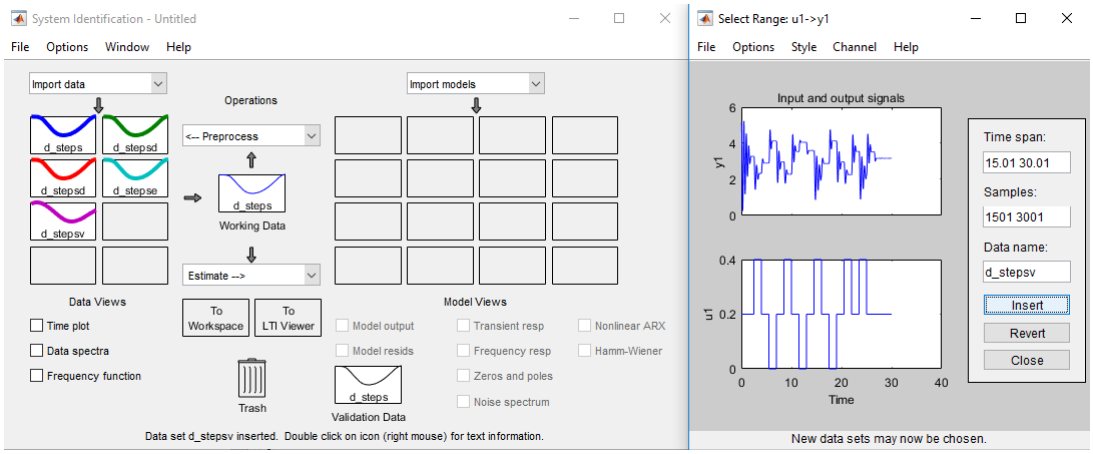

Figure 10: Preprocessing of data

Step 2.2: Select the 4^{th} and close the three windows to go back to the main identification tool window. MATLAB estimates the matrices Φ, Γ and C from equation (5). (Note that the matrix D is set to 0 by default, otherwise it should be set to 0).

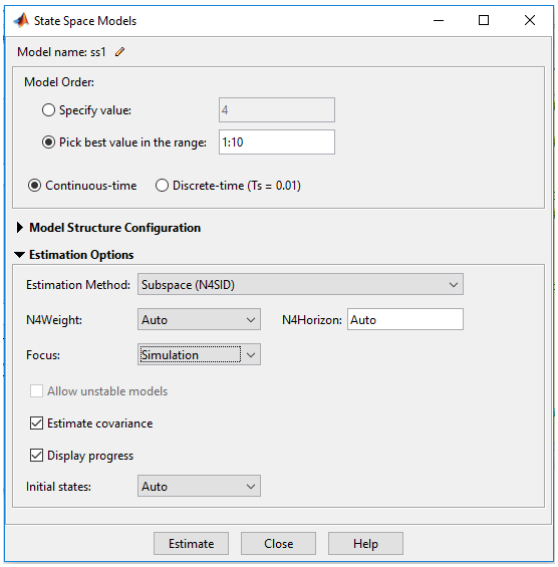

Figure 11: State Space Model Estimation

2.4.3 Step 3: Model Validation

Step 3.1: Tick the "module output" option to compare and validate the estimated model. The plots show the simulated (predicted) outputs of selected models. The models are fed

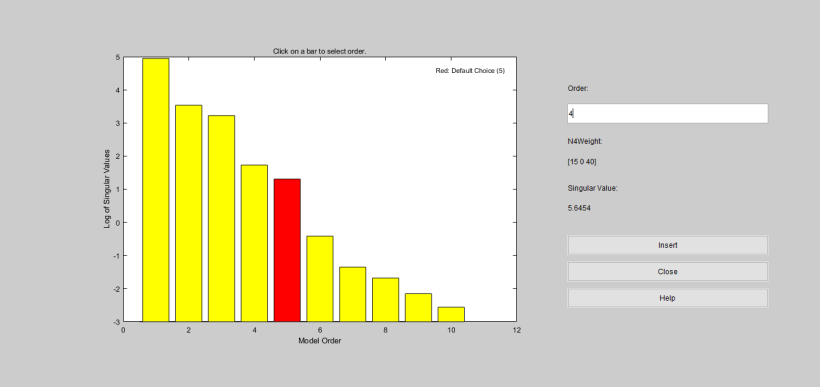

Figure 12: Hankel Matrix and order selection

with inputs from the Validation Data set, whose output is plotted in black. Note that you should also compare "channel two", to make sure that the predicted linear model is matching the whole response of the system beyond the time span of "channel one".
If the estimation is close to the real values of the plant, the response of the estimated system is close to the real one measured (Fig.13).

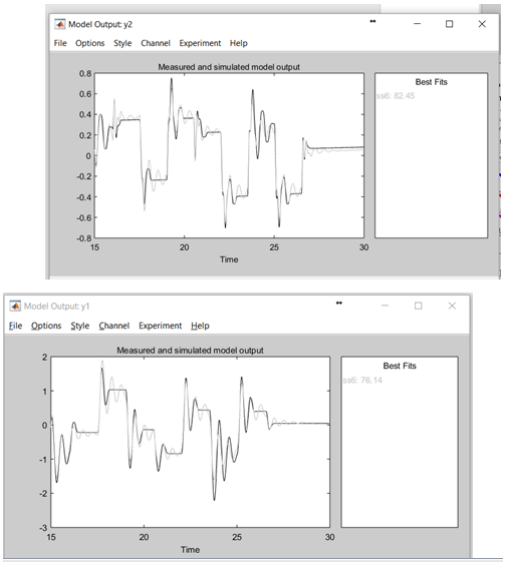

Figure 13: Validation of State Space Estimation

Step 3.2: Export the model to the workspace by dragging and dropping the block "SS6" (in this case).

Note: For more readings on subspace identification, read Chapter 7 of the Control systems course lecture notes, or check the corresponding lecture notes of the Linear and Non Linear System Identification course.

2.4.4 Step 4

Prep. 2.5: Use the provided data sets to identify a state space model with each input signal. In each case, select an appropriate model order. Compare the four models in terms of frequency response and pole zero map (typical examples are shown in Fig. 14 and Fig. 15). Select the model you will use for controller design. (`cstd2_ident.m`)

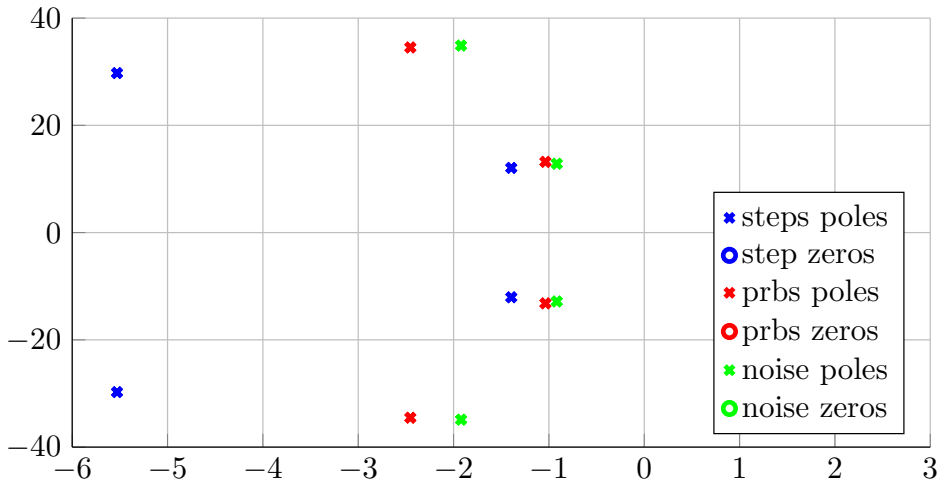

Figure 14: Comparison of the poles and zeros for different models

Prep. 2.6: For which sampling rates would it be ok to perform a continuous-time design and discretize the controller?

Prep. 2.7: Select a suitable sampling rate for the experiments.

3 CONTROLLER DESIGN

The next task will be to design an LQG controller that stabilizes the plant and allows reference tracking. This control scheme consists of a linear quadratic regulator and a

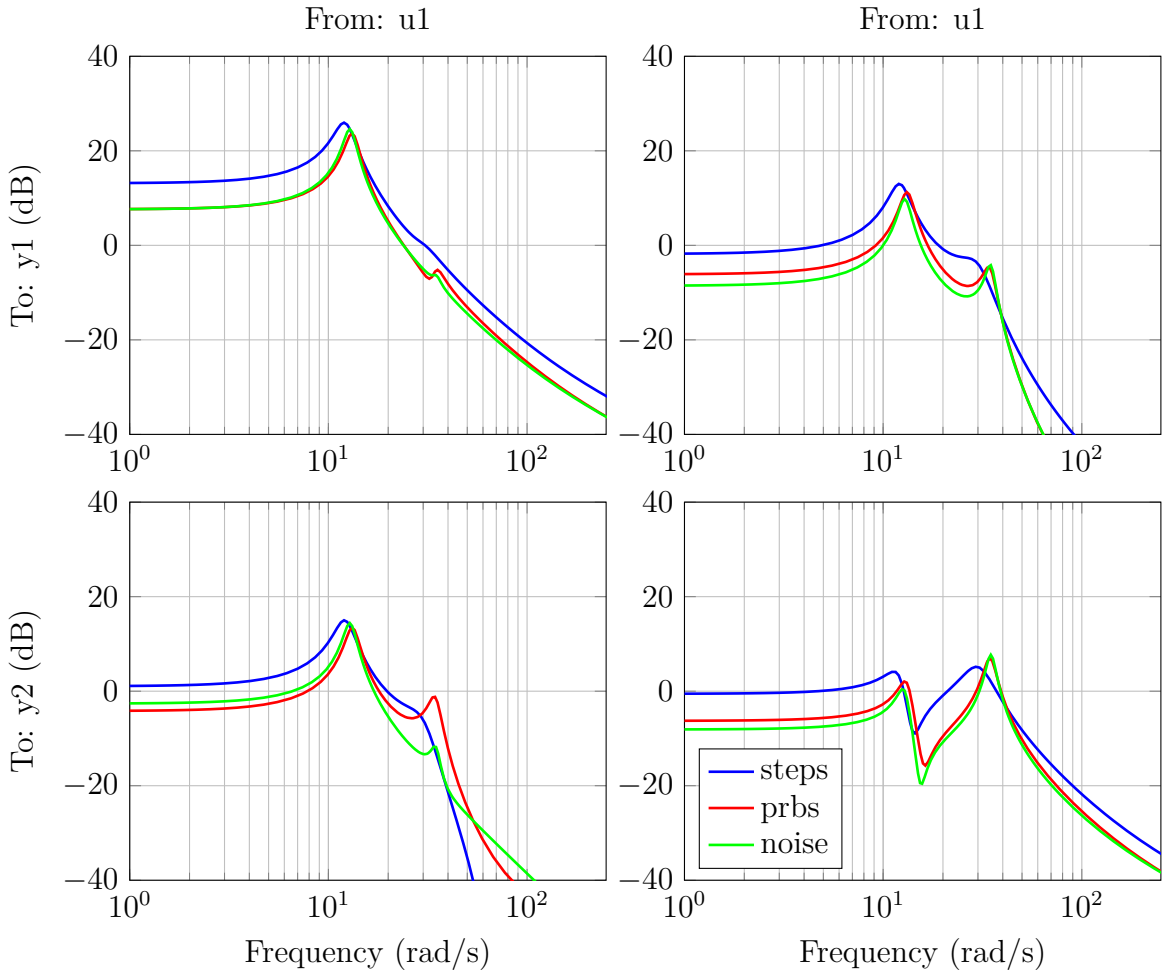

Figure 15: Comparison of the bode magnitude plots for different models

state estimator. A block diagram of this scheme is given in Figure 4.1 of the lecture notes *Control Systems Theory and Design*.

The tasks in this section need to be completed by filling the gaps in in `cstd2_design.m`.

3.1 State Feedback Design

The first step of the design is the computation of the optimal state feedback gain F as discussed in Section 5.9 of the lecture notes.

The relevant Matlab command is

```
F = -lqr(A,B,Q,R);
```

Prep. 3.1: What are the tuning knobs for this design and what are common choices?

Prep. 3.2: Compute an optimal feedback gain for the identified plant.

3.2 State Estimator Design

Figure 3.2 of the lecture notes shows the structure of a state estimator.

Prep. 3.3: How should the observer poles be selected, relative to the poles assigned by the state feedback?

Prep. 3.4: Calculate an optimal estimator gain L and construct the observer state space model.

3.3 Design for Reference Tracking

Section 4.2 of the lecture notes discusses the introduction of reference inputs.

Prep. 3.5: Explain purpose and calculation of the static gain V.

Prep. 3.6: Implement the reference tracking scheme shown in Figure 4.3 of the lecture notes.

Prep. 3.7: Design V for the selected model and simulate a step response with all identified models.

Prep. 3.8: Explain possible difficulties with respect to the steady state error of this design.

3.4 Integral Action

Prep. 3.9: Review the concept of adding integral action into the loop as discussed in Exercise 4.5 of the lecture notes.

Prep. 3.10: Extend the state space model to include integral action.

Prep. 3.11: Compute the optimal state feedback gain F and the integrator gain F_I.

Prep. 3.12: Simulate the closed-loop step response of this design with all identified models.

3.5 Tuning of the controller

Prep. 3.13: Review the relative effects of the entries in tuning matrices on the closed-loop behavior.

Prep. 3.14: Tune the controller in the simulation in such a way that the closed-loop requirements are met.

Prep. 3.15: Review the effect of measurement noise on the design.

Prep. 3.16: Repeat your simulation with noise switched on and retune your controller design if required.

4 EXPERIMENT

Before working with the real plant, students should prepare themselves by going through the preliminary tasks and carefully reading the safety instructions contained in the next section. The main tasks in this section are similar to the preliminary ones, but the object is now the real plant. The real-time system employed in this plant is the Real-Time-Windows-Target Toolbox provided by Matlab.

4.1 Safety Instructions

⚠ **Do NOT start the plant without a safety introduction by the supervisor!**

⚠ **In the event of an emergency, control effort should be immediately discontinued by pressing the red OFF button on front of the control box.**

⚠ **Stay clear of and do not touch any part of the mechanism while it is moving, while a trajectory has been commanded or before the active controller has been safety checked.**

⚠ **Verify that the magnets and glass rod are secured prior to powering up the Control Box.**

4.2 Realtime Model

Fig. 16 shows the Simulink model with the plant block for the communication with the real-time hardware. The switches can be used to activate certain components of the control loop.

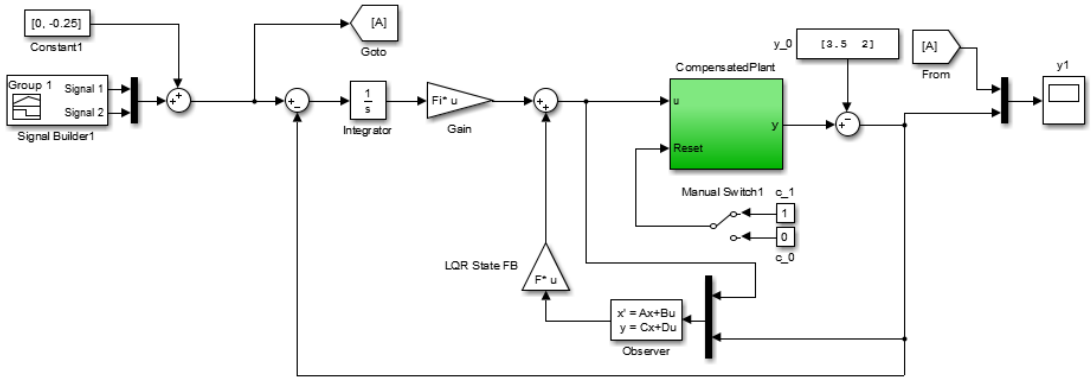

Figure 16: Simulink model used for the experiment

4.3 Experimental Task

The following tasks are to be fulfilled in the laboratory. (The real-time hardware has a maximum bandwidth of 100 Hz)

1. Perform a system identification experiment. Choose a suitable open loop excitation signal and identify a model. Validate your model with input data different from the identification experiment.

2. Use your identified model to synthesize an LQG controller and an LQG controller with integral action. Test the tracking performance of your controller by following the reference trajectories in Fig. 3.

3. Tune your controller on the real plant to meet the performance requirements given in Section 1.1.