Automatic Control

Compensation of a position servo

Abstract

The angular position of the shaft of a DC motor is to be controlled. Using frequency analysis, the dynamics of the DC motor is determined. From a step response a parametric model is also estimated. From this model, a lead-lag compensator is constructed, so that given specifications are fulfilled.

Preparation exercises: See page 6–8, and 10. Also, read through the whole instruction carefully.
## Contents

1 Introduction ................................................. 3
   1.1 Objectives .................................................. 3
   1.2 Survey ......................................................... 3
   1.3 Equipment .................................................... 3
   1.4 The process ................................................... 3
   1.5 Initial setup ................................................... 4
   1.6 How to start up the computer ......................... 5
   1.7 Open loop control ........................................... 5
   1.8 Unity feedback control ..................................... 5
   1.9 Lead control ................................................... 6

2 Finding a process model ................................. 7
   2.1 Frequency analysis ........................................ 7
      2.1.1 Introduction ............................................... 7
      2.1.2 Estimation of the frequency properties ................ 8
   2.2 Determination of the servo dynamics by transient analysis ........ 9

3 Lead compensator design .............................. 10

4 Measuring the performance of the closed loop system .... 11
   4.1 Step response ................................................ 11
   4.2 Ramp response ............................................... 11
   4.3 Using the DC servo ......................................... 13
1 Introduction

1.1 Objectives

When you have carried out this laboratory work, you will have learned

- How to determine a Bode diagram, the phase margin and the crossover frequency by using frequency analysis.
- How to estimate a parametric model from a step response.
- How to construct a lead-lag compensator from an estimated Bode diagram.

1.2 Survey

In this laboratory assignment, the angular position of the shaft of a DC motor is to be controlled. The goal is to fulfill some specifications on servo performance (rise time, overshoot and steady-state ramp error). To be able to achieve this you need to have knowledge about the dynamics of the DC motor.

When dealing with control problems, the work typically consists of two stages:

- Modeling of the process. In this laboratory work, you will:
  - Estimate a Bode diagram directly by using frequency analysis.
  - Estimate a parametric model based on physical insight.
- Designing the regulator. Using the frequency properties of the process, you will design a lead-lag filter, which gives the compensated system desired performance.

1.3 Equipment

Besides the process you will use a computer and a signal generator to complete this laboratory work. The computer runs in a LINUX environment and the program to be used is MATLAB.

1.4 The process

In Figure 1, the different parts of the process are displayed. Also, it is indicated how they are connected.

- The input to the process is the voltage $u(t)$ to the amplifier unit AU150B, which affects the angular position of the shaft of the DC motor.
- The output from the process is the voltage $y(t)$ from the potentiometer OP150K. The output $y(t)$ is proportional to the angular position of the shaft of the DC motor. On this shaft, a protractor (“gradskiva”) is attached to simplify the reading of the angular position.
- In Section 2.2 you will also use the angular velocity of the DC motor. This is measured with a tachometer which gives a voltage signal proportional to the angular velocity.
The potentiometer OP150K transforms any angle between $+150^\circ$ and $-150^\circ$ to a voltage. The voltages $+15\text{V}$ and $-15\text{V}$ from the power supply of the servo are connected to OP150K, and the resulting voltage is measured. The relationship between the angle $\theta(t)$ in radians and the voltage $y(t)$ becomes

$$y(t) = K_p \theta(t) \quad [\text{V}]$$

$$K_p = \frac{30}{300} = \frac{18}{\pi} \quad [\text{V/rad}] \quad (1)$$

### 1.5 Initial setup

The initial setup of the servo process is described below. Check carefully so that the cables are connected as described.

- The input signal from the computer should be connected as follows: From DA channel 0 to input channel 4 and 6 on the input attenuator AU150B. The positive pole of the DA channel should be connected to channel 6 on AU150B and the negative pole on the DA channel should be connected to channel 4 on AU150B.

- To measure the output of the servo process, cables should be connected from AD channel 0 to the servo OP150K. The negative pole of the AD channel should be connected to output 0 on OP150K, and the positive pole of the AD channel should be connected to output 3 on the servo.

- To use an external reference signal (i.e. the signal from IP150H), cables should be connected from 3 on the input potentiometer IP150H to the positive pole of AD channel 2 and from 0 on IP150H to the negative pole of AD channel 2.
To measure the angular velocity of the servo motor, the tachometer unit should be connected to AD channel 1.

1.6 How to start up the computer

The assistants will inform you how to start up the computer, if it is not already started.

Observe that this particular MATLAB variant do not allow you to save data. Also observe that it is possible to get help about any MATLAB function by giving the MATLAB command `help function name`.

1.7 Open loop control

The first experiment in this session will be to manually control the servo system. Proceed as follows:

Move the “+ cable” from terminal 0 on the D/A converter to 3 on the potentiometer IP150H. This signal will now serve as the controller output \( u(t) \). Turn on the power and try to control the position of the the servo (OP150K) by turning the knob on the potentiometer (IP150H). Try for example to make a fast step change.

Comments:

1.8 Unity feedback control

The simplest feedback controller that we can use is the Unity Feedback Controller (UFC). The control signal is simply the control error, \( i.e. u(t) = e(t) = r(t) - y(t) \). Try to control the system by using this method. Proceed as follows:

Move the “+ cable” back to 0 on the D/A converter.
Give the command \texttt{ufc} in MATLAB. This will start the unity feedback controller. Determine the rise time and overshoot of the system (overshoot and rise time will be presented in the plot when the program is executed). Note that the reference signal is taken from the computer.

\[ T_r = \quad \text{Overshoot} = \quad \]

**Comments:** (Overshoot, settling time and rise time. Compare with the requirements in 1.9.)

1.9 Lead control

In order to give the system better dynamic properties than those obtained by UFC, you will use a lead filter, see Figure 2.

Your task is to control the process such that:

- the rise time is reduced to 50\% compared to the UFC system,
- the overshoot becomes smaller than 10\%  
- the stationary ramp error is smaller than 0.01V for a ramp with the slope 0.2V/s.

You will have to use a lag compensator in order to fulfill the last requirement, but do not bother about the lag compensator at this stage.

![Figure 2: A DC position servo with compensation link](image)

**Preparation exercise:** To reduce the rise time by 50\%, the bandwidth of the compensated system has to be doubled, \( i.e. \omega_{e,\text{desired}} \geq 2\omega_c \). How large must the phase margin \( \phi_m \) be for the compensated system (open loop), to have an overshoot less than 10 \%, when controlled in closed loop? (Hint: the process can be considered as a second order system, see Figure 5.11 in Glad-Ljung.)
Preparation exercise: The lead parameter $\beta$ is related to the phase margin, see e.g. Figure 5.13 in the course book. Since it is good to have a large phase margin (stability and overshoot) why not use a very small $\beta$? Hint: study the Bode diagram for the lead compensation link.

Preparation exercise: What will a step response look like for a second order system with a large and a small phase margin respectively? Comment on overshoot, settling time and rise time.

2 Finding a process model

Two approaches will be studied to find a process model, frequency analysis and transient analysis.

2.1 Frequency analysis

For the design of the lead compensator you need to do a frequency domain estimation of the process.

2.1.1 Introduction

You will in the following estimate the frequency properties of the process by measuring the gain $|G(i\omega)|$ and the phase shift $\arg G(i\omega)$ of the process for a number of frequencies. Let the input to the system be a sinusoid with amplitude $u_0$ and frequency $\omega$.

When the transients have disappeared, the output of the system is a sinusoid with amplitude $y_0$, the same frequency $\omega$ and phase $\phi$. This situation is indicated in Figure 3.

\[ u(t) = u_0 \sin \omega t \quad \xrightarrow{G(i\omega)} \quad y(t) = y_0 \sin(\omega t + \phi) \]

Figure 3: A sinusoid on the input generates a sinusoid with the same frequency on the output, but with different amplitude and phase.
2.1.2 Estimation of the frequency properties

The frequency analysis is performed by the MATLAB program `freqservo`. The full syntax of this program is `[w,mag,fi]=freqservo`. `w` is here the entered frequency points, `mag` is the output magnitude and `fi` is the phase shifts. *It is important that you do not overwrite the vectors `w`, `mag` and `fi when the experiment has been performed, these vectors will be used later on.* The user will be asked to give a desired frequency. A sine signal with the given frequency generated by the computer will then be used as input signal `u(t)`. The input and output signals will be shown in real time on the screen. When the sine sequence is completed, the program will estimate the magnitude and phase shift of the output signal, and then ask for a new frequency. These measurements will be presented both in a Bode diagram and numerically on the screen. As a complement, the detrended input and output signals are plotted for each frequency. Use this plot as a complement to check that the values of magnitude and phase shift that the program returns are reasonable. When you exit the program, interpolated values of the phase margin and the crossover frequency are shown in the MATLAB window. It is very important that you do many measurements around the crossover frequency to get high accuracy in these parameters. Note that it is also possible to use the command `[w,mag,fi]=freqservo(w,mag,fi)`. The new frequency points you choose are then added to your old vector.

The measured output signal will in many cases not be located symmetrically around zero (it has a bias) which makes the output signal lying above or below the input signal in the display window. If this deviation is “large”, you can correct the measurement by adjusting the position of the scale on the output potentiometer OP 150K by hand before each new measurement.

**Preparation exercise:** Which (frequency-) points are most essential for the design of the lead compensator, in order to fulfill the requirements stated in Section 1.9?

Select proper frequencies when you perform the experiment (preferably in the interval $\omega \in [1, 8]$ rad/s). Use the table below to write down the values that are essential for the lead design.

| $\omega$ [rad/s] | $|G(i\omega)|$ | arg $G(i\omega)$ [°] |
|-----------------|-----------------|------------------|
|                 |                 |                  |
|                 |                 |                  |

8
2.2 Determination of the servo dynamics by transient analysis

As an alternative to the frequency analysis in Section 2.1.2, the servo dynamics will here be determined by using a step response and some measurements. The transfer function for the servo process can be described by the following model (see example 2.3 in Glad-Ljung)

\[ G(s) = \frac{K_m}{s(1 + \tau s)} = \frac{Y(s)}{U(s)} \]  

(2)

This model describes how the angular position \( y(t) \), measured in volts, (see (1)), depends on the control signal \( u(t) \). The model (2) can be divided in two parts and a state variable \( v(t) \) (angular velocity) can be introduced.

\[ V(s) = \frac{K_{m1}}{1 + \tau s} U(s) \]  

(3)

\[ Y(s) = \frac{K_p}{s} V(s) \]  

(4)

Here, \( v(t) \) is the angular velocity, \( K_p \) is given in (1), and \( K_{m1} \) is an unknown gain. Note that

\[ K_m = K_{m1} K_p \]  

(5)

The parameter \( \tau \) can be determined from a step response. Applying the inverse Laplace transform to (3) for a step response in \( u(t) \) gives

\[ v(t) = K_{m1} (1 - e^{-\frac{t}{\tau}}) u(t) \]  

(6)

At time \( t = \tau \) the velocity \( v(t) \) has reached \( (1 - e^{-\frac{1}{\tau}}) \approx 0.63 \) of the steady-state value.

Use the MATLAB program tachometer that measures the time \( \tau \) it takes for the servo to reach 63% of its steady-state velocity measured by the tachometer\(^1\).

\[ \tau = \ldots \ldots \ldots \ldots \]

Next, the gain \( K_m \) in (2) is estimated. The key idea is to select the input so that the servo is running with \( 2\pi \) rad/s (one revolution per second). Let \( u_s \) denote the input signal which gives \( v(t) = 2\pi \) rad/s. The ratio \( 2\pi /u_s \) gives \( K_{m1} \) defined in (3). In order to get \( K_m \) we use (1) and (5). The gain \( K_m \) is thus given by

\[ K_m = \frac{2\pi}{u_s} \frac{18}{\pi} = \frac{36}{u_s} \]  

(7)

To generate a voltage \( u \) on the motor, enter the MATLAB command proc_da(0,u). Change \( u \) until one round takes 1 s (\( \Rightarrow v(t) = 2\pi \) rad/s). For simplicity, measure the time over 10 rounds.

\[ K_m = \frac{36}{u_s} = \ldots \ldots \ldots \]

Summarize your results below:

\(^1\)The tachometer gives a signal which is proportional to the angular velocity. The value of the proportional constant is not important here since only the \( \tau \) is estimated
\[ G(s) = \frac{K_m}{s(1 + \tau_s)} = \ldots \ldots \quad (8) \]

Use the command `bodecomp` to compare the parametric and non-parametric models in the frequency domain. Use the MATLAB command `help bodecomp` for details on how to run `bodecomp`. Notice that a polynomial \( p(s) \) in MATLAB is written as a vector listing the coefficients in descending powers of \( s \) (For example: \( s^2 + 2s + 3 \) is written as \([1 \ 2 \ 3]\)).

**Comments:** (Comment the result. Try to list some advantages of using parametric models over non-parametric models, and vice versa.)

3 Lead compensator design

Use the essential points of the frequency estimations to find out how much phase increase \( \Delta \phi \) that is necessary in order to fulfill the first two requirements stated in Section 1.9.

\[ \omega_{c, desired} = \Delta \phi = \ldots \ldots \quad (9) \]

The transfer function of the lead compensator is described by

\[ F(s) = K F_{lead}(s) = \frac{\tau_D s + 1}{\beta \tau_D s + 1} \quad (10) \]

**Preparation exercise:** Show that \[ |F_{lead}(i\frac{1}{\tau_D \sqrt{\beta}})| = \frac{1}{\sqrt{\beta}} \]

**Preparation exercise:** What is the appropriate choice of \( K \) to get \( \omega_{c, desired} \)?
Determine the parameters of the lead compensator such that desired performance is achieved. Describe in a few words how you proceed, step by step. Also, motivate your choices of the values of the parameters $K$, $\tau_D$ and $\beta$.

4 Measuring the performance of the closed loop system

4.1 Step response

Now, use your lead compensator and measure the performance. This is done by the MATLAB command `lead(K,beta,Td,'step')`. The last parameter in the function gives a step as reference signal. The other possible options are 'ramp' and 'ext'. The step and ramp reference signals are generated by the computer while 'ext' takes an external reference signal from IP150H. The overshoot and rise time are presented in the plot. Nevertheless, you should be prepared to present the definition of these properties.

Write down and comment on overshoot and rise time below.

$$T_r = \underline{\phantom{000}}$$

$$Overshoot = \underline{\phantom{000}}$$

4.2 Ramp response

Use the MATLAB command `lead(K,beta,Td,'ramp')` to run the servo with a ramp with the slope $0.2V/s$ as reference signal. The ramp error is presented in the plot. Comment on it below.
The ramp error, as you can see, does not fulfill the given controller specifications. Design a lag compensator to decrease the error, and show below how you determined the parameters $\tau_I$ and $\gamma$.

Now, by the command `leadlag(K,beta,Td,gamma,Ti,'ramp')`, run the leadlag controller with a ramp as reference signal. Comment on the ramp error.

If the ramp error is small enough, study the step response. This task is solved with the command `leadlag(K,beta,Td,gamma,Ti,’step’)`. Are the specifications still fulfilled? If they are not it is possible to tune the controller parameters by hand to obtain better performance (usually this is difficult).
4.3 Using the DC servo

As a final exercise, run the process as a servo, i.e. use the potentiometer IP150H to generate the reference signal, that is, use the command `leadlag(...,'ext')`. If all cables are connected according to Section 1.5 and your compensator is correct, the protractor of the output potentiometer will follow the reference signal potentiometer.

Would you like to have this as a steering servo in your car? If not, restart the laboration from Section 2.