FRTN35 System Identification
Laboratory Exercise 1

Michael Lundh
Fredrik Bagge Carlson
Department of Automatic Control
Lund University

1. Introduction

The purpose of this laboratory experiment is to determine the frequency response of a linear process. The frequency response is determined to get insight in the process dynamics and is to be used for controller design. The process will finally be controlled.

Before you proceed, install the `BallAndBeam.jl` package using the instructions provided at gitlab.control.lth.se/processes/BallAndBeam.jl. This installation will take a few minutes if this is your first time using Julia.

It's possible to perform this entire lab against a simulated version of the beam. This can be done in order to be well prepared for the actual lab session. See instructions in Section 4.2 or Algorithm 1.

1.1 Equipment

The process is a beam mounted on a motor shaft, see Figure 1.1. The process input, u, is a reference signal to an internal velocity controller and the output, y, is the beam angle. There are some mechanical resonances in the beam. A PC running Julia[1] will be used both for the initial Frequency Response Analysis and for the latter design and control.

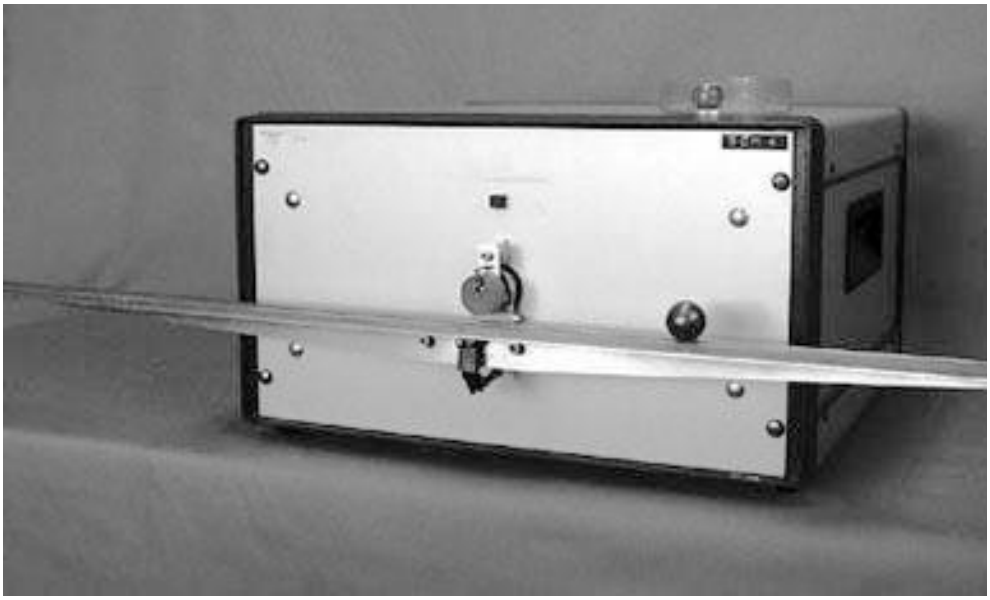

Figure 1 The beam dynamics from velocity reference to beam angle should be identified in this laboratory exercise.

1.2 Preparations

Read Chapter 2 in [Johansson, 1993]. Solve exercises marked with *Preparation* in this guide. Study section 6.6 on compensation in [Åström, 1968] or Sections 5.4–5.5 in [Glad and Ljung, 1989]. Solutions should be presented before the experiments start, and some questions concerning the theory should be answered.

[1] Julia is an open-source, high-level, high-performance dynamic programming language for numerical computing. The syntax of Julia is similar to Matlab but with a few noteworthy differences listed at julialang.org/noteworthy-differences

2. The Method

The system is forced by a sinusoidal input signal $u(t) = u_0 \sin(\omega t)$ where the u_0 and ω are to be chosen. When transients have decayed, the output signal is described by $y(t) = |G(i\omega)|u_0 \sin(\omega t + \arg G(i\omega))$. The experiment is repeated for a number of frequencies ω so that a Bode-diagram may be drawn.

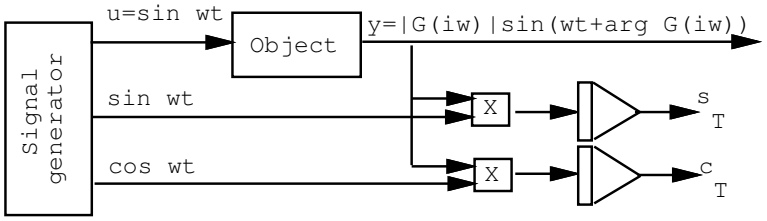

Figure 2 Schema of the Frequency Response Analysis method.

Direct, accurate measurement of the amplitudes of u and y and their phase lag is difficult. A better method is to integrate for a certain time T to get

$$Y_s(T) = \int_0^T y(t) \sin(\omega t) dt$$
$$Y_c(T) = \int_0^T y(t) \cos(\omega t) dt$$

for each frequency ω. If the experiment time is chosen to be a multiple of the period of u, i.e. $T = n \cdot 2\pi/\omega$ we will have

$$Y_s(T) = \frac{T}{2} u_0 \mathrm{Re}\ G(i\omega) = \frac{T}{2}|G|u_0 \cos\phi$$
$$Y_c(T) = \frac{T}{2} u_0 \mathrm{Im}\ G(i\omega) = \frac{T}{2}|G|u_0 \sin\phi$$

From these quantities the magnitude and the phase shift of $G(i\omega)$ is computed.

$$|G(i\omega)| = \frac{2}{Tu_0}\sqrt{Y_s^2(T) + Y_c^2(T)}$$
$$\arg G(i\omega) = \arctan \frac{Y_c(T)}{Y_s(T)}$$

The correlation method can be viewed as filtering $y(t)$ through a band-pass filter with center frequency ω and bandwidth proportional to $1/T$.

3. Experiment Setup

The frequency response analyzer described above is implemented in the Julia package `BallAndBeam.jl` and performs the calculations automatically. First some experiment conditions must be determined, the most important of which are discussed here.

3.1 Frequency Range

Determine the frequency interval where it is essential to know the process. To do this some a priori knowledge is required, maybe from a previous experiment. A reasonable range for the beam is the interval $[0.5, 300]$ rad/s.

Questions: What does this range correspond to in the time domain? Why is this range reasonable?

3.2 Amplitude and Bias

Determine the amplitude u_0 of the input signal. Select the amplitude so that the system remains linear, i.e. signals must in general be sufficiently small. The frequency response analysis may be repeated for different amplitudes to investigate linearity. For the beam let $u_0 \in [0.5, 1.0]$ V.

Since the process is not asymptotically stable (a constant input, u, gives a constant angular velocity and increasing angle) you may need to add a bias term to the input signal, $u(t) = u_b + u_0 \sin(\omega t)$. This is necessary for the beam to remain in approximately the same position during long experiments.

3.3 Measurement Time

The noise level determines the measurement time required to obtain desired accuracy. A long measurement time implies low noise sensitivity. In these experiments we chose the measurement time to be multiple of periods of the forcing signal.

Preparation: How is accuracy related to measurement time if the measurement of y is corrupted by almost white noise?

3.4 Delay

Determine how long to wait before starting measuring. It is the time from forcing with a certain frequency until integration begins. This time depends on how fast the transients decay in the system. Transients of the beam decay in a few seconds. The time given is rounded up to the nearest integer periods.

4. Experiment

In a real situation little is initially known about the process. The identification procedure will therefore be iterative. A new experiment is planned using experience from previous ones. You were given some hints in the last section.

You should do identification experiments until you have such knowledge of $G(i\omega)$ that you are able to plot a Bode-diagram without strange discontinuities. This requires high frequency resolution in some intervals.

Investigate also the linearity of the process, by comparing two identical experiments, except for different signal amplitudes.

The experiments are performed in open loop despite the fact that the process is not asymptotically stable. It is however stable. The frequency response analysis will work since the drift is small.

4.1 Connections

The switch Man/Auto on the backside of the process should be in **Auto** and the following connections should be made:

PC	Beam process
Analog Out 1	IN
Analog In 0	angle
ground	ground

4.2 Julia Session

Frequency Response Analysis Install the `BallAndBeam.jl` package using the instructions provided at gitlab.control.lth.se/processes/BallAndBeam.jl. The instructions provides you with a script on which you can base your FRA experiments, outlined below.

Algorithm 1 Example usage. If you want to perform simulations instead of experiments on the real process, change `P = LabProcesses.Beam(0.01, bias)` to `P = LabProcesses.BeamSimulator(0.01)`

```
using BallAndBeam, LabProcesses, ControlSystems, JLD
# @load "workspace.jld" # Run this command to restore a saved workspace

bias = 0 # Change this if your process drifts over time
P    = LabProcesses.Beam(0.01, bias) # Change this line to
 ↪  BeamSimulator(0.01) if you want to simulate
h    = sampletime(P)

settling_time  = 1
nbr_of_periods = 5

# Below we define some frequency vectors (using logarithmic spacing)
# and run three experiments. You may modify the freqency vectors
# any way you want and add/remove experiments as needed.

w1_100 = logspace(log10(1),log10(300),8)
G1     = fra(P, w1_100, amplitude=1, bias=bias,
 ↪  nbr_of_periods=nbr_of_periods, settling_time=settling_time)

w1_200 = logspace(log10(5),log10(50),20)
G2     = fra(P, w1_200, amplitude=2, bias=bias,
 ↪  nbr_of_periods=nbr_of_periods, settling_time=settling_time)

w1_300 = logspace(log10(10),log10(30),20)
G3     = fra(P, w1_300, amplitude=2, bias=bias,
 ↪  nbr_of_periods=nbr_of_periods, settling_time=settling_time)

@save "workspace.jld"

# Concatenate (overlapping) estimates to be used and sort based on freq
G123 = sortfqs([G1; G2; G3])

bopl(G123)
nypl(G123)
```

The frequency response estimates are stored in matrices (`G123` in Algorithm 1) which have two columns, one with frequencies ω_k (in radians/s) and the other with complex numbers $G(i\omega_k)$.

4.3 Discussion

Try to determine the characteristics of the process using the Bode-diagram. What can you say about

- the order of the system?
- poles and zeros?
- static gain?
- linearity?

5. Control design

Here we assume that the main purpose for the identification is to achieve a frequency response $G(i\omega)$, that could be used to design a controller giving desired closed-loop system properties.

We use Julia to design a controller on the form (see Fig. 3)

$$u(s) = \frac{S(s)}{R(s)}\left(\frac{B_{ff}}{A_{ff}}r(s) - y(s)\right)$$

This is a two-degree of freedom controller. First the feedback compensator, $G_{FB} = \frac{S}{R}$, is designed to give the desired loop-transfer function. Then a feedforward filter, $G_{FF} = \frac{B_{ff}}{A_{ff}}$, may be used to shape the response for command signals. The closed loop system will then be

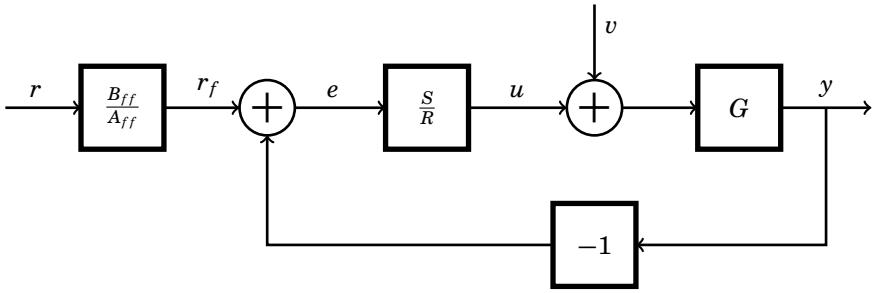

Figure 3 Block diagram of the control loop with feedback compensator $G_{FB} = \frac{S}{R}$ and feedforward filter $G_{FF} = \frac{B_{ff}}{A_{ff}}$

Design goal: Make the closed-loop system as fast as possible and fulfill that the rise time be less than 0.2 s and the maximum overshot less than 10%.

5.1 Feedback Design

A feedback compensator is designed using classical frequency compensation. The feedback is used mainly to get a reasonably fast and well damped closed-loop system by changing the cross-over frequency and phase margin. This is readily done with

```
sysFBc,L,T,C = fbdesign(G, polevect, zerovect, gain)
```

where `sysFBc` is a `StateSpace` object describing the feedback controller, $C = S(i\omega)/R(i\omega)$, $L = G \cdot C$ and $T = L/(1+L)$. This function requires the user to specify gain, poles and zeros of the compensator. Write `?fbdesign` for help. Plots of the uncompensated and the compensated system may be drawn using `bopl(G), bopl!(T)`. See Algorithm 2 for a complete example.

Preparation: Find the transfer functions in the block diagram that have the frequency responses C, L and T.

5.2 Feed-forward Design

The feedback design specifies T. A feedforward filter may then be added to reduce the high frequency content in the reference signal. This is done with

```
sysFFc,YR,FF = ffdesign(T, polevect, zerovect, gain)
```

where the feedforward filter `sysFFc` $= B_{ff}/A_{ff}$ has frequency response FF and the frequency response from r to y is $YR = T \cdot FF$. Gain, poles and zeros are chosen as in `fbdesign`. Plots of T and YR may be drawn using `bopl, nypl`.

If you have time, go over the feedback and feedforward design several times to create different control designs. Then you can compare a faster design with a slower one, etc.

5.3 Real Time Control

The controller design is carried out in continuous time, but a discrete time controller is needed since the controller is implemented in a computer. If the sampling rate is sufficiently high the discrete controller will behave as the continuous one. Discretization of the controllers is easy to perform using the function `c2d`.

5.4 Control experiments

Save your workspace in Julia to have a backup of your design. To perform a control experiment, call the function

```
run_control_2DOF(P, sysFB, sysFF, duration=10, reference =
↪  t->sign(sin(2π*t)))
```

where `reference(t)` is a reference generating function, which defaults to a square wave with frequency $f = 1$Hz. If you have time you can alter this function, try for example a sinusoidal reference. A complete example of the control design procedure is given in Algorithm 2. The controller function can only accept discrete-time linear systems as inputs, hence the feedback and feedforward controllers must be discretized with the correct sample time, use `sysFB = c2d(sysFBc, h)[1]` for this.

Do the controllers and the system behave as expected?

6. Conclusions

The identification has been used for two purposes. Firstly an accurate frequency response was determined to gain insight into the dynamics of the beam. Secondly, the frequency response was used for design of a controller, which was tested on the real process to validate the identified model.

Algorithm 2 Example control design procedure.

```
polevect     = [-10]
zerovect     = []
gain         = 1
sysFBc,L,T,C = fbdesign(G, polevect, zerovect, gain)

polevect     = [-10]
zerovect     = []
gain         = 1
sysFFc,YR,FF = ffdesign(T, polevect, zerovect, gain)

bopl(C, lab="Controller")
bopl!(L, lab="Closed-loop system e->y")
bopl!(FF, lab="Feedforward compensator")
bopl!(YR, lab="Closed-loop system r->y")

sysFB,sysFF  = c2d(sysFBc,h)[1],c2d(sysFFc,h)[1]
y,u,r        = run_control_2DOF(P, sysFB, sysFF, duration=5, reference =
 ↪  t->2sign(sin(2/3*t)))
plot([y u r], lab = ["y" "u" "r"])
```

6.1 Discussion

How would you obtain an estimate of the system transfer function $G(s)$ given the obtained measurements $G(i\omega)$ for $\omega \in \Omega$?

7. Extra

If you have time left, estimate a transfer function model directly from data using an ARX model. Input sequences to excite the process with are, e.g., a PRBS signal or Gaussian noise. The script `FRTN35_lab1.jl` contains code to get you started.

```
prbs     = PRBSGenerator()
duration = 10
y        = zeros(0:h:duration)
u        = zeros(0:h:duration)
for (i,t) = enumerate(0:h:duration)
  @periodically h begin
    y[i]  = measure(P)
    u[i]  = prbs()-0.5
    control(P, u[i])
  end
end
plot([u y])
```

With the code above, we excite the process and store the control signal and measurement sequences in variables u and y. The code

```
na = 4 # Order of A polynomial
nb = 2 # Order of B polynomial
arxtf,  = arx(h, y, u, na, nb) # Estimate transfer function with ARX
 ↪  method
```

```
bopl(G123, lab="Measured transfer function")
bodeconfidence!(arxtf, ,  = logspace(0,3,200))
```

will estimate a transfer function model and plot a Bode diagram of the result as well as the data identified using FRA.

7.1 Discussion

Compare the result using ARX and FRA. Are the estimated Bode diagrams similar?

Noise sensitivity How do the two estimation methods handle measurement noise?

Ease of use Which method is faster to use? How easy is it to use the result of the identification, i.e., using measurements of $G(i\omega)$ for $\omega \in \Omega$ or using a transfer function $G(s)$.

Validation How do you validate the results of the identification when using the different estimation methods?

8. References

[Åström, 1968] Åström, K. J. (1968). *Reglerteori*. Almqvist & Wiksell.

[Glad and Ljung, 1989] Glad, T. and Ljung, L. (1989). *Reglerteknik—Grundläggande teori*. Studentlitteratur, Lund, 2nd edition.

[Johansson, 1993] Johansson, R. (1993). *System modeling & identification*. Prentice-Hall, Englewood Cliffs, NJ.