Dyson School of Design Engineering

Imperial College London

DE2.3 Electronics 2

Lab Experiment 2: System Characterisation & Transfer Function

(webpage: http://www.ee.ic.ac.uk/pcheung/teaching/DE2_EE/)

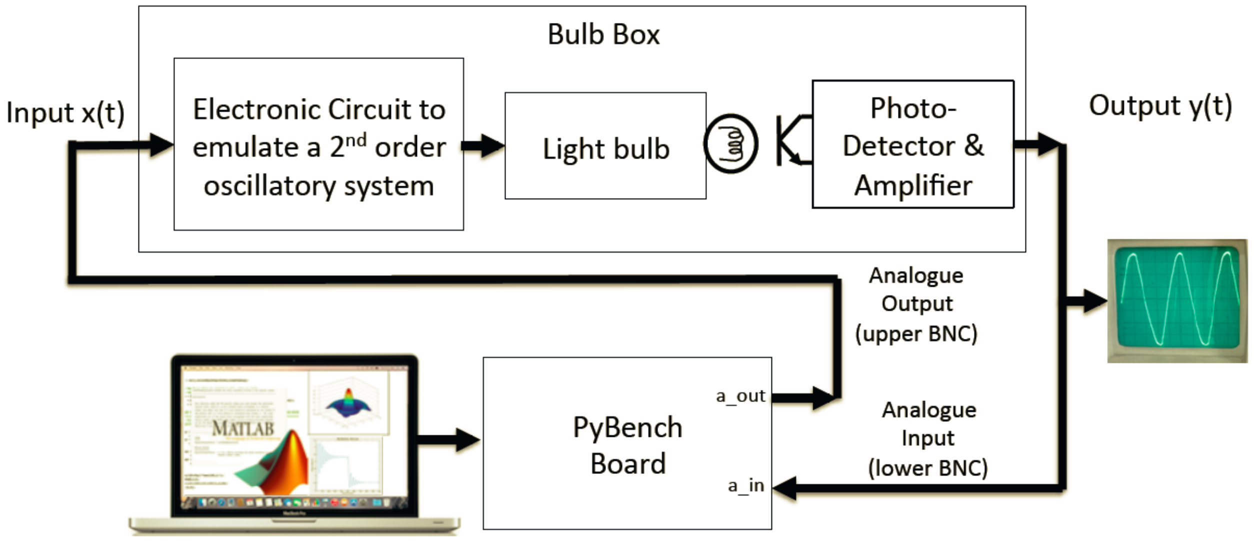
Objectives

By the end of this experiment, you should have achieved the following:

- Obtain the input-output relationship of a system at zero frequency.
- Obtain the frequency response of a system at different frequencies.
- Obtain the step response of a system.
- Understand the meaning of transfer function of a system in the s-domain.
- Use Matlab to predict frequency response of a system from its transfer function.
- Compare the measured and the theoretic behaviour of a system.
- Understand the transient behaviour of a system.

The Experimental System

In this experiment, you will connect the PyBench board to the **Bulb Box**, which is a piece of equipment to emulate (meaning 'mimic' or 'pretend to be') a second-order system. The following figure shows the experimental setup.


Bulb Box
Input x(t)
Electronic Circuit to emulate a 2nd order oscillatory system
Light bulb
Photo-Detector & Amplifier
Output y(t)
Analogue Output (upper BNC)
a_out
PyBench Board
a_in
Analogue Input (lower BNC)
MATLAB

Inside the Bulb Box is an electronic circuit that behaves like a second-order system that is highly oscillatory. It is driven by an input signal x(t). The output of this circuit drives an incandescent light bulb. The intensity of the light emitted from the bulb is measured by a photo-transistor. The photo-transistor produces an electrical current that is approximately proportional to the light intensity, and this current is amplified by another circuit to produce the system output y(t).

Throughout this experiment, you will be driving and measuring signals to and from the Bulb Box using the PyBench board under the control of Matlab.

A simple test to verify if your PC is communicating with the PyBench board properly: Open Window Explorer or Finder. Check that there is a USB drive called PYBENCH. If yes, it is working.

Exercise 1: DC Characteristic of the Bulb Box

- Remove the black shorting link (between the two BNC connectors) so that you are no longer connecting the analogue input and output of PyBench together. Put the link back on one of the pins so that it is not misplaced or lost.
- Connect PyBench to the Bulb Box as shown in the diagram above.
- Open Matlab on your PC, and enter the following:

```
device_name = 'COM4';                     % for PC
pb = PyBench(device_name);
pb.dc(3.0);        % output 3.0v to Bulb Box to x_dc
pause(1);          % delay for 1 sec
pb.get_one()       % read y_dc
```

- You should see the light bulb brightly lid. Now try output a zero voltage and measure the output y. The reading should be close to zero and the light should be OFF.
- Perform a few measurements for y_dc = 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, and measure y_dc. Roughly sketch the **y_dc** vs **x_dc** graph.
- Doing measurement manually like this is tedious. Write a Matlab program **lab2_ex1.m** that performs the characterization automatically. Since we are only interested in the **steady state response** (at DC or zero frequency) of the system to the input drive **x_dc**, you need to put in a delay of at least 0.5 sec after changing **x_dc**. Wait for the output y_dc to settle to its final value before measuring. Otherwise you will be measuring the **dynamic** behaviour of the system.
- Plot **y_dc** vs **x_dc** in Matlab and label your axes and the graph. This is the DC characteristic of the Bulb Box system. Make sure you understand the reason for the shape of this graph. Explain why this system is non-linear.
- What would you expected the **y_dc** vs **x_dc** curve would be like if the system were linear?
- Over what region of input voltage x that the system may be approximately linear?

Optional Challenge (to do at home if you have time)

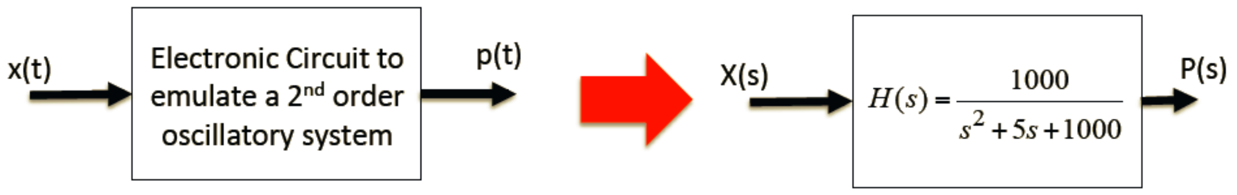
You can model the DC behaviour of the Bulb Box with a non-linear function F:

y_dc = F(x_dc)

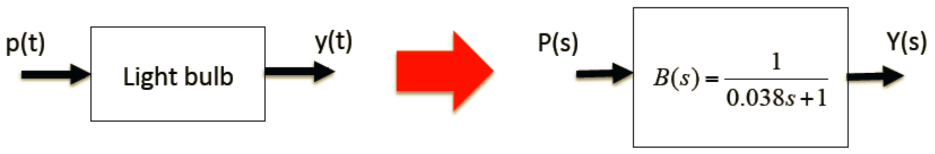
Non-linear systems are hard to compute and analyse. We can "linearize" the system by finding the inverse characteristic F^{-1}, such that F x F^{-1} = 1. How would you derive F^{-1} in Matlab?

Exercise 2: Frequency Response of the Bulb Box system - Simulation

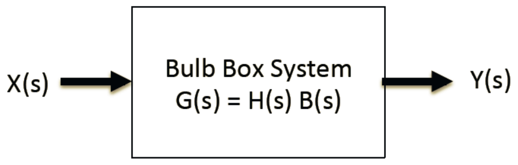
The Bulb Box system can be modeled mathematically as a linear system (ignoring the non-linear DC characteristics for now) as two things connected in series. Firstly, the second order electronic circuit, which is known to have a transfer function as shown:


x(t)
Electronic Circuit to emulate a 2nd order oscillatory system
p(t)
X(s)
$H(s) = \frac{1000}{s^2 + 5s + 1000}$
P(s)

The light bulb, detector/amplifier can be modeled (ignoring the non-linearity) as a first order system as:



The overall system can therefore be modeled in the Laplace s-domain as G(s) = H(s) B(s):

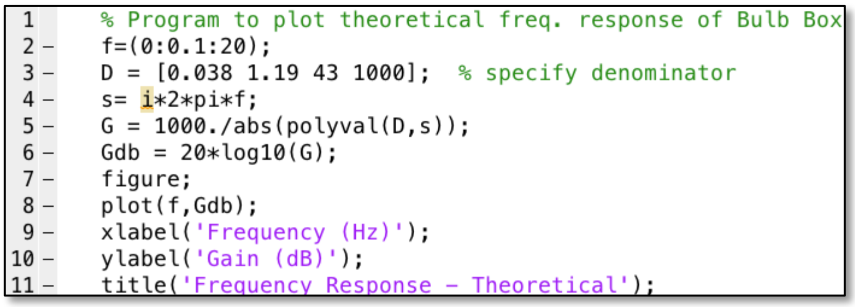


$$G(s) = \frac{1000}{0.038s^3 + 1.19s^2 + 43s + 1000}$$

We are interested in discovering the system's gain G for sinusoidal signals x(t) at different signal frequencies. This is called the **frequency response** of the system. That is we want to find:

$$G(j\omega) = \frac{\text{amplitude of y(t)}}{\text{amplitude of x(t)}} \quad \text{at different } \omega.$$

we can derive the theoretical frequency response G(jω) by evaluating G(s) with s = jω. The following Matlab program will perform the calculation and plot the theoretical frequency response:



```
1      % Program to plot theoretical freq. response of Bulb Box
2 -    f=(0:0.1:20);
3 -    D = [0.038 1.19 43 1000];  % specify denominator
4 -    s= i*2*pi*f;
5 -    G = 1000./abs(polyval(D,s));
6 -    Gdb = 20*log10(G);
7 -    figure;
8 -    plot(f,Gdb);
9 -    xlabel('Frequency (Hz)');
10 -   ylabel('Gain (dB)');
11 -   title('Frequency Response - Theoretical');
```

Line 2: specifies the vector f, from 0Hz to 20Hz in steps of 0.1Hz. This range is selected because it is known that system's oscillatory behaviour is well below 20Hz.

Line 3: creates a vector specifying the coefficients of the denominator polynomial of G(s), in the order of highest power of s to lowest power.

Line 4: s is the vector specifying the frequency of interest. Note that to get the frequency response, we evaluate G(s) at s = jω, where ω = 2πf. Matlab uses i to specify imaginary number instead of j.

Line 5: This calculates G at the different frequencies. Look up Matlab help page for **polyval(.)** function.

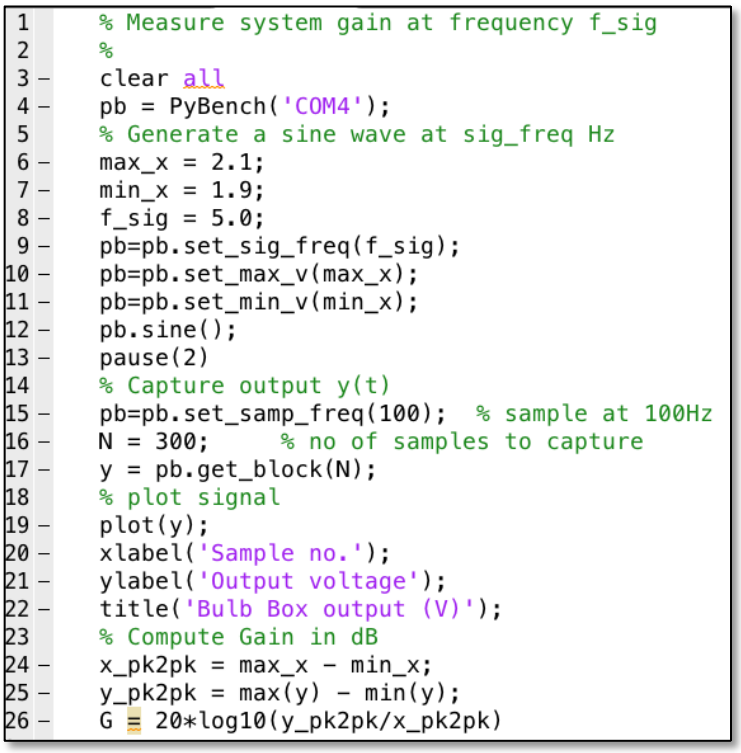
The rest of the Matlab code is self-explanatory.

- Enter the Matlab script, run the script and comment on the results.
- Manually evaluate G(s) $|_{s=j\omega}$ at frequency of 0 Hz and 5 Hz, and check that this agrees with the Matlab prediction. (Remember that ω = 2πf.)

Exercise 3: Measure the Frequency Response of the Bulb Box system using PyBench

In this exercise, you will measure the frequency response of the actual Bulb Box using PyBench and compare your measurements with the theoretical prediction.

Since the system is non-linear, we will only operate the system for input x(t) between 2.1V to 1.9V. Enter and test the following Matlab code to produce a sinusoidal signal at 5Hz and capture 200 samples of the output signal y(t) at a sampling frequency fs of 100Hz, and plot the signal. Finally, it calculates the gain of the system |G(jω)| at 5Hz (i.e ω = 2π*5) and convert this to dB. Make sure that you understand how this works. Remember, you need to substitute 'COM4' here with a suitable name for your USB port for your computer. For Windows 10 systems, it would be COMx, where x can be any number up to 10's. For MacBook users, you need the name like /dev/tty.usbmodemxxxxxx, where xxxx can be a very large number.



```
% Measure system gain at frequency f_sig
%
clear all
pb = PyBench('COM4');
% Generate a sine wave at sig_freq Hz
max_x = 2.1;
min_x = 1.9;
f_sig = 5.0;
pb=pb.set_sig_freq(f_sig);
pb=pb.set_max_v(max_x);
pb=pb.set_min_v(min_x);
pb.sine();
pause(2)
% Capture output y(t)
pb=pb.set_samp_freq(100);  % sample at 100Hz
N = 300;      % no of samples to capture
y = pb.get_block(N);
% plot signal
plot(y);
xlabel('Sample no.');
ylabel('Output voltage');
title('Bulb Box output (V)');
% Compute Gain in dB
x_pk2pk = max_x - min_x;
y_pk2pk = max(y) - min(y);
G = 20*log10(y_pk2pk/x_pk2pk)
```

Repeat the measurement at f = 1Hz, 3Hz, 4Hz, 5Hz, 7Hz and 9Hz. Sketch the measured gain G(jω) vs frequency to get an idea of the shape as compared to that found theoretically in Exercise 2. (You may also mark the measured points on the theoretical frequency response plot.)

Now create a your own Matlab program **lab2ex3a.m** which performs automatic measurement of G(jω) for frequencies between 1Hz and 20Hz. Plot the frequency response (i.e. Gain in dB vs Frequency). Compare the measured frequency response with the theoretical plot. Comment on the results.

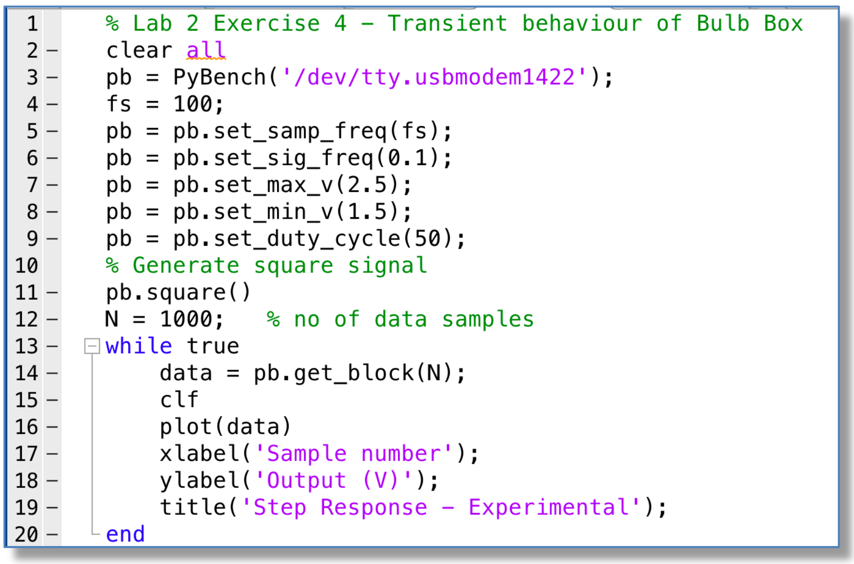
(My solution to lab2ex3a.m is included in the Appendix for those running out of time.)

Exercise 4: Measure the Step Response of the Bulb Box

Frequency response is useful only to measure the "*steady state*" condition of a system. That's why we used the **pause(2)** statement in the previous exercise to wait for the "transient" to die down. In this exercise, you will be examining the transient behaviour of the Bulb Box by driving it with a step function.

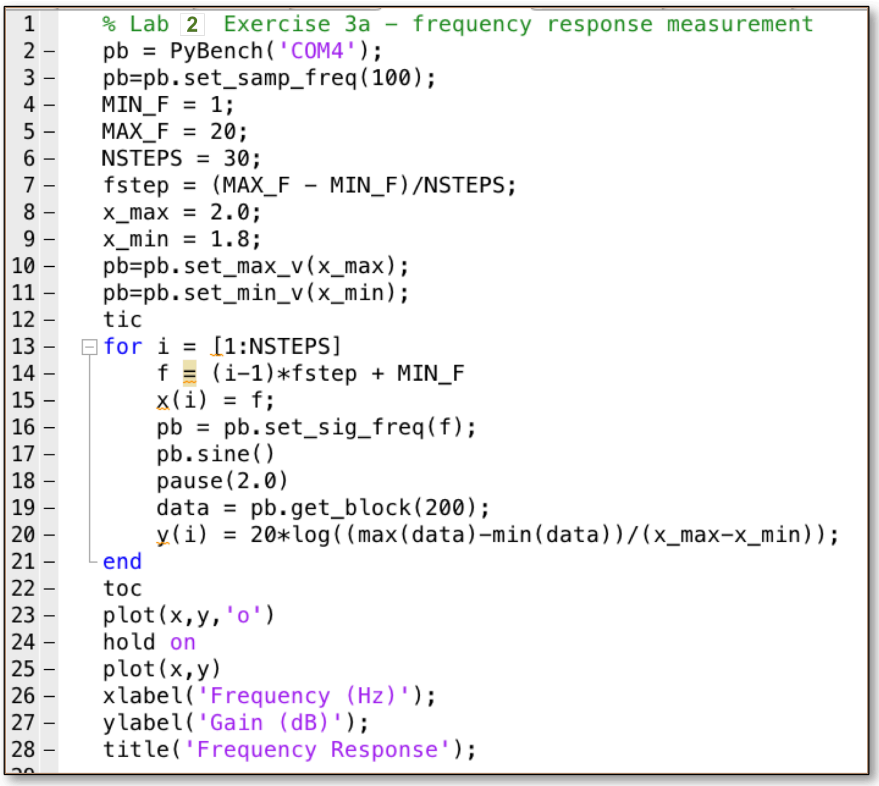
In an experiment, we often use a square wave at a low frequency (say 0.1Hz) to mimic both a rising, followed by a falling, step function. The repetition frequency (in this case, 0.1 Hz) must be chosen to last for long enough such that the transient oscillations have decayed by the end of the half period.

Enter the following Matlab code, and obtain the step response of the Bulb Box. Explain what you see.



```
% Lab 2 Exercise 4 – Transient behaviour of Bulb Box
clear all
pb = PyBench('/dev/tty.usbmodem1422');
fs = 100;
pb = pb.set_samp_freq(fs);
pb = pb.set_sig_freq(0.1);
pb = pb.set_max_v(2.5);
pb = pb.set_min_v(1.5);
pb = pb.set_duty_cycle(50);
% Generate square signal
pb.square()
N = 1000;   % no of data samples
while true
    data = pb.get_block(N);
    clf
    plot(data)
    xlabel('Sample number');
    ylabel('Output (V)');
    title('Step Response – Experimental');
end
```

APPENDIX Solution to Exercise 2a



```
% Lab 2 Exercise 3a – frequency response measurement
pb = PyBench('COM4');
pb=pb.set_samp_freq(100);
MIN_F = 1;
MAX_F = 20;
NSTEPS = 30;
fstep = (MAX_F - MIN_F)/NSTEPS;
x_max = 2.0;
x_min = 1.8;
pb=pb.set_max_v(x_max);
pb=pb.set_min_v(x_min);
tic
for i = [1:NSTEPS]
    f = (i-1)*fstep + MIN_F
    x(i) = f;
    pb = pb.set_sig_freq(f);
    pb.sine()
    pause(2.0)
    data = pb.get_block(200);
    y(i) = 20*log((max(data)-min(data))/(x_max-x_min));
end
toc
plot(x,y,'o')
hold on
plot(x,y)
xlabel('Frequency (Hz)');
ylabel('Gain (dB)');
title('Frequency Response');
```