EE2703:
APPLIED PROGRAMMING LAB

Assignment 6

The Laplace Transform

Arjun R
EE20B016

April 5, 2022

Contents

1 Aim

This assignment aims to analyse **Linear Time-Invariant Systems** with the use of *Continuous Time Fourier Transforms*. Here, will be making use of `scipy.signal` module from Python.

```
import scipy.signal as sp
```

2 Problems

2.1 P1. Forced Damped Oscillator (decay=0.5)

The first problem is to solve for time-dependent position function of a forced damped oscillator with zero initial conditions ($x(0) = 0$ and $\dot{x}(0) = 0$):

$$\ddot{x} + 2.25x = f(t)$$

where

$$f(t) = cos(1.5t)e^{-0.5t}u_0(t)$$

Solving for $X(s)$, we get:

$$X(s) = \frac{F(s)}{s^2 + 2.25}$$

Python function `sp.lti` can be used to generate the transfer function and `sp.impulse` function can be used to find the impulse response of $X(s)$.

```
def transfer_fn(freq, decay):
    den = polymul([1, 0, 2.25], [1, 2*decay, decay**2 + freq**2])
    return sp.lti([1, decay], den)

t, x = sp.impulse(transfer_fn(freq=1.5, decay=0.5), None, linspace(0, 50, 501))
plot(t, x)
xlabel(r'$t\rightarrow$')
ylabel(r'$x(t)\rightarrow$')
grid()
title('Forced Damped Oscillator with decay = 0.5')
show()
```

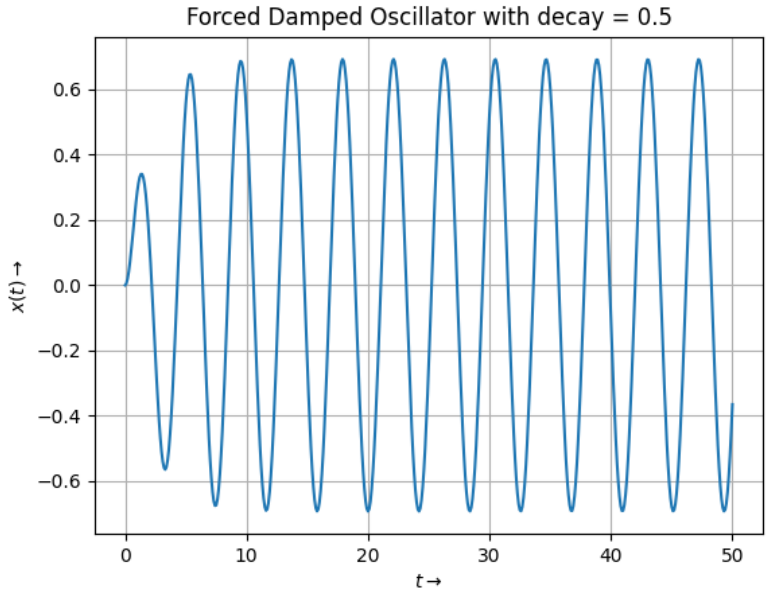


Figure 1: Forced Damped Oscillator with decay = 0.5

2.2 P2. Forced Damped Oscillator (decay=0.05)

Doing the same for a forced damped oscillator with a smaller decay constant gives the plot in figure 2.

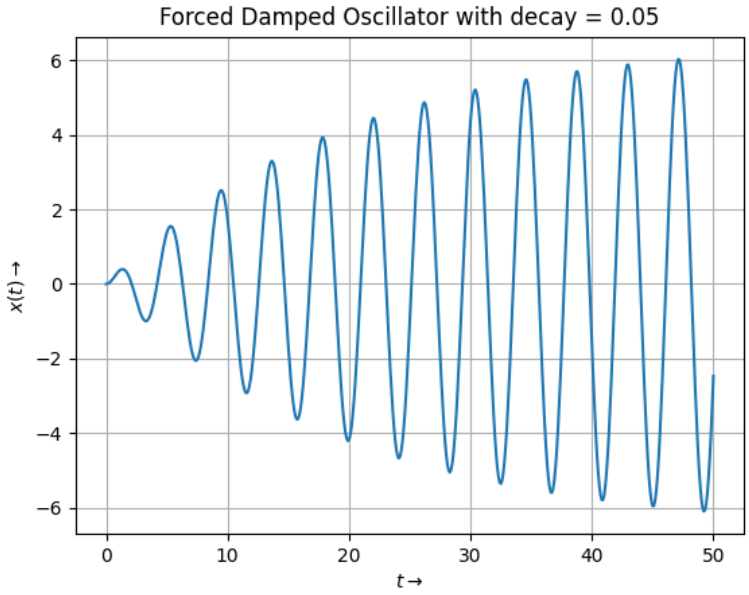


Figure 2: Forced Damped Oscillator with decay = 0.05

2.3 P3. Forced Damped Oscillator on varying Driving Frequency

We can reframe the forced damped oscillator problem by considering it to be an LTI system with input $f(t)$ and output $x(t)$. We can then solve for $x(t)$ by convolution of $f(t)$ with impulse response of the system, which is obtained by inverse Laplace transform of system

transfer function $X(s)/F(s)$. We can use function `sp.lsim` for this purpose.

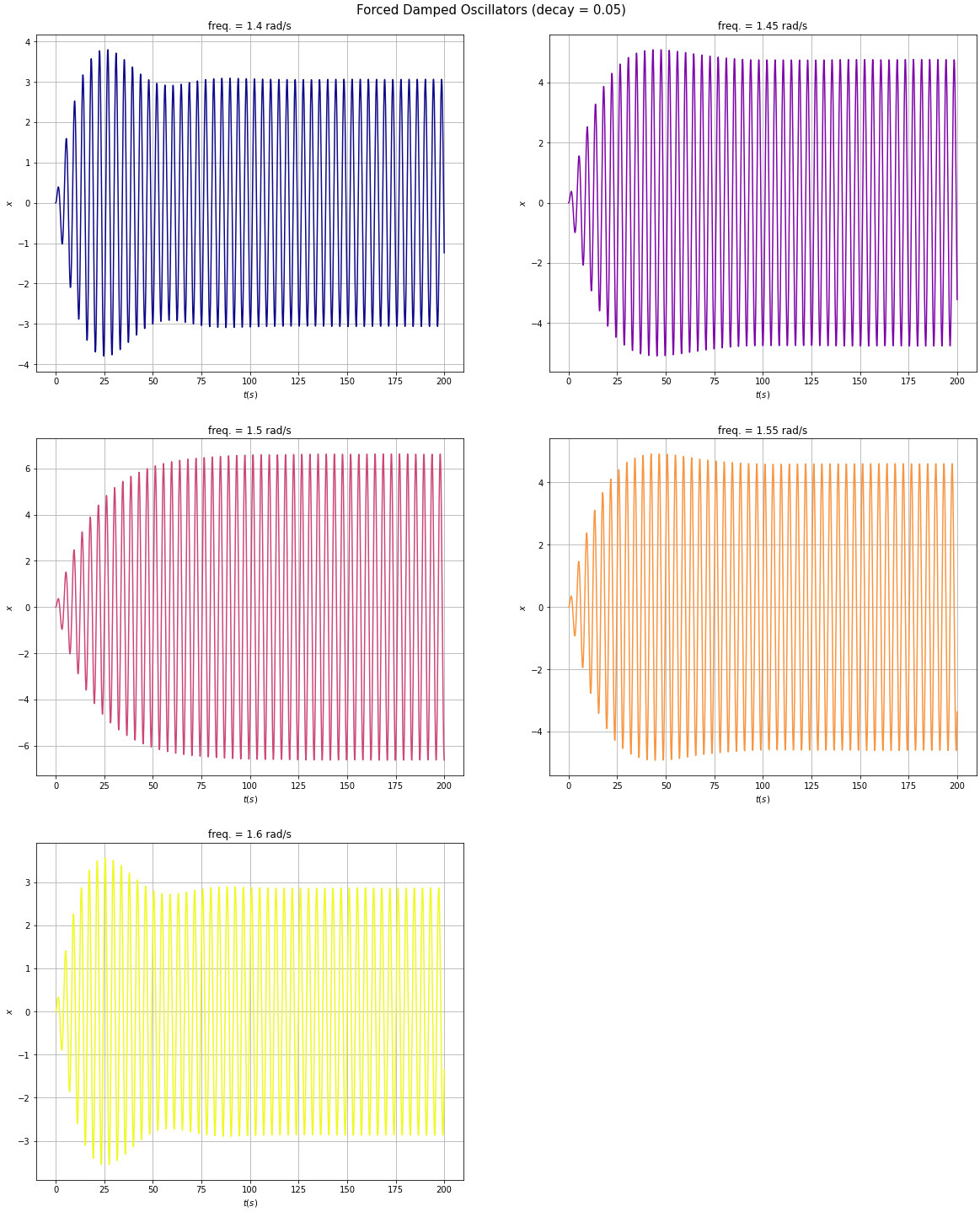


Figure 3: Forced Damped Oscillators (decay = 0.05) on varying driving frequency

```
def f(t, freq, decay):
    return cos(freq*t)*exp(-1*decay*t)*(t>0)

def sys_transfer_fn():
    return sp.lti([1], [1, 0, 2.25])
```

```
t = linspace(0, 200, 2001)
fig, ax = subplots(3, 2, figsize = (20, 25))
fig.suptitle('Forced Damped Oscillators (decay = 0.05)', y = 0.9, fontsize = 15)
freq_range = arange(1.4, 1.65, 0.05)
color = iter(cm.plasma(linspace(0, 1, 5)))
for i in range(5):
    t, x, svec = sp.lsim(sys_transfer_fn(), f(t, freq_range[i], 0.05), t)
    ax[i//2, i%2].plot(t, x, c=next(color))
    ax[i//2, i%2].set_xlabel(r'$t(s)$')
    ax[i//2, i%2].set_ylabel(r'$x$')
    ax[i//2, i%2].set_title(f'freq. = {freq_range[i]} rad/s')
    ax[i//2, i%2].grid()

fig.delaxes(ax[2, 1])
show()
```

The frequency of the cosine in $f(t)$ is varied from 1.4 to 1.6 in steps of 0.05 keeping decay constant as 0.05. Figure 3 gives different system responses on varying the driving frequency.

Observation - 1.5 is the natural frequency of the spring system and we can see that when driven at this frequency, the settling amplitude for the system is maximum. As frequency moves farther away from 1.5, there is significant overshoot and lesser settling amplitudes.

2.4 P4. Coupled Oscillator System

In this problem, we had to solve for the time evolution in the case of a coupled spring system given by:

$$\ddot{x} + (x - y) = 0$$

$$\ddot{y} + 2(y - x) = 0$$

where initial conditions are $x(0) = 1, \dot{x}(0) = y(0) = \dot{y}(0) = 0$.
On solving, we get:

$$X(s) = \frac{s^2 + 2}{s^3 + 3}$$

$$Y(s) = \frac{2}{s^3 + 3}$$

```
X = sp.lti([1, 0, 2], [1, 0, 3, 0])
Y = sp.lti([2], [1, 0, 3, 0])
t = linspace(0, 20, 201)
t, x = sp.impulse(X, None, t)
t, y = sp.impulse(Y, None, t)
```

2.5 P5. RLC Low Pass Filter Circuit

In this problem, we have to find the steady state Transfer function for a two-port network with a series RLC Low Pass filter circuit and plot Bode magnitude and phase plots. The code implementation is as follows:

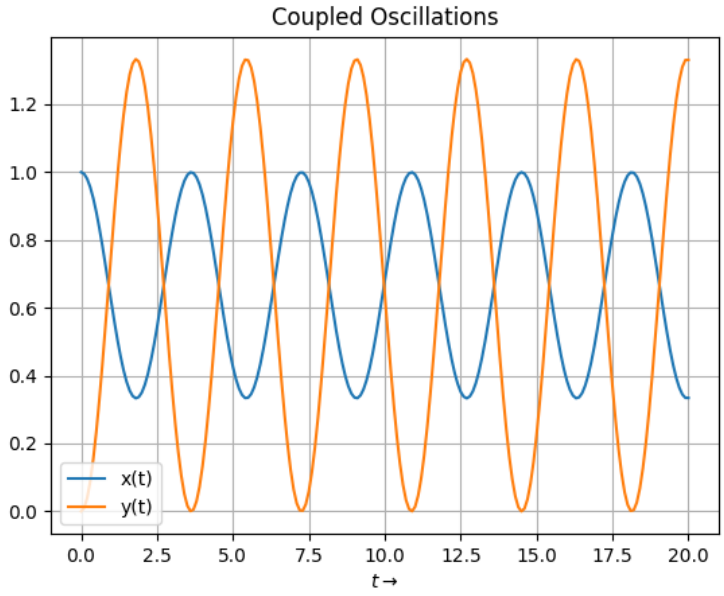


Figure 4: Coupled Oscillator

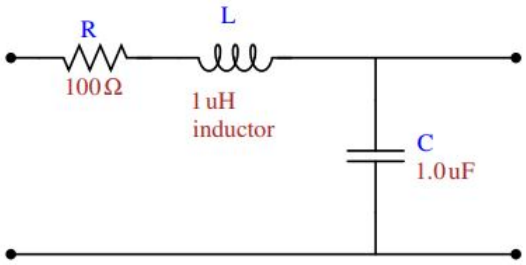


Figure 5: RLC Two-Port Network

```
H_ckt = sp.lti([1], [1e-12, 1e-4, 1])
w, mag, phi = H_ckt.bode()
figure(figsize=(12, 5))
subplot(1, 2, 1); semilogx(w, mag)
subplot(1, 2, 2); semilogx(w, phi)
```

2.6 P6. Time Domain Responses for the Low Pass Filter Circuit

In the final problem, input signal $v_i(t)$ is applied to the previous low pass filter circuit.

$$v_i(t) = cos(10^3 t)u(t) - cos(10^6 t)u(t)$$

We have to obtain output signal $v_o(t)$. Using experience from our previous analysis, we can consider the circuit to be an LTI system and find the convolution of the input signal with the impulse response of the system for finding the output signal.

The transfer function of the LTI system is given by

$$H(s) = \frac{V_o(s)}{V_i(s)} = \frac{1}{s^2LC + sRC + 1}$$

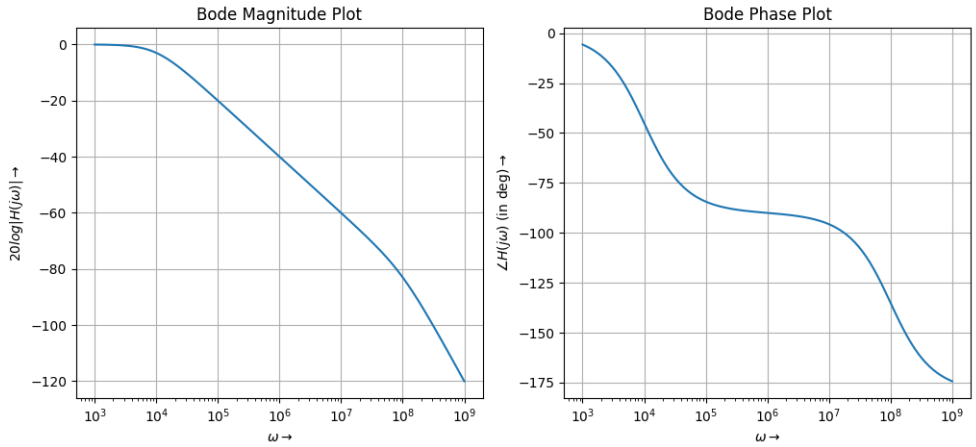


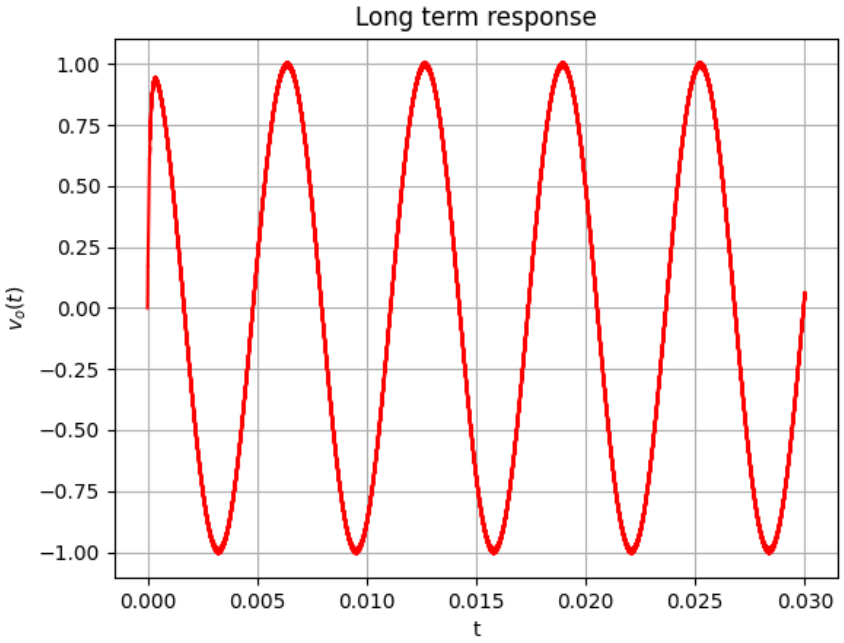
Figure 6: Bode magnitude and phase plots for the Low Pass filter circuit

```
t = linspace(0.5e-6, 3e-2, 60001)
vi = (cos(1e3*t) - cos(1e6*t)) * (t>0)
t, vo, _ = sp.lsim(H_ckt, vi, t)
```

Observations -

- In the short-term response, the $cos(10^3)t$ component remains roughly constant while the $cos(10^6)t$ component oscillates. Thus we get an oscillating response with a slow increase.
- In the long-term response, the $cos(10^3)t$ component remains while the $cos(10^6)t$ component gets filtered out (decrease of 40dB in magnitude response) due to the low-pass filter. Thus we can see that the $cos(10^3)t$ component dominates. Figure 7b can be compared to figure 8.

(a) Short term response



(b) Long term response

Figure 7: Time domain responses for the Low-Pass Filter Circuit

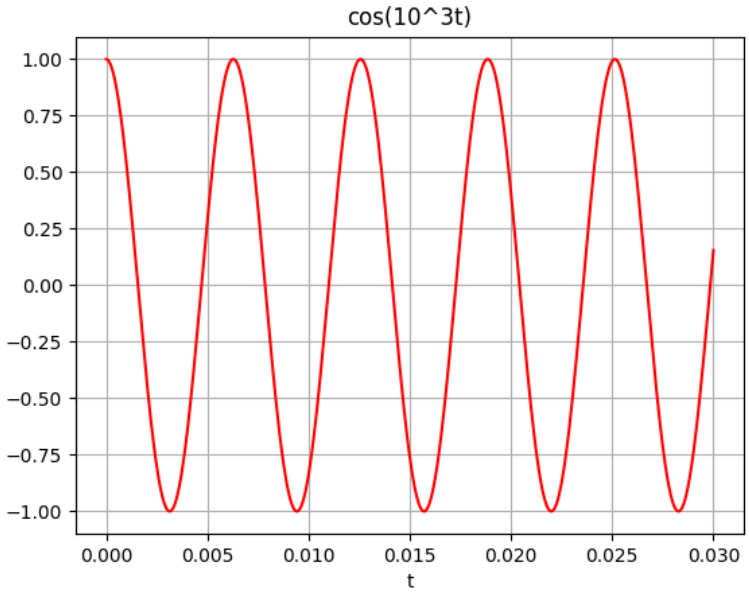


Figure 8: $cos(10^3 t)$

3 Conclusion

LTI systems are very important for electrical engineers. In this assignment, we took a step in this direction using the capabilities of `scipy`'s signal processing library. In particular, we analysed forced damped oscillators, coupled oscillators and electric filter circuits.