EE2703: Applied Programming Lab
Assignment 6
The Laplace Transform

Arjun Menon Vadakkeveedu
EE18B104
Electrical Engineering, IIT Madras

March 9, 2020

Contents

1 Introduction

The Laplace Transform is a very powerful tool used to analyse Linear Time Invariant systems (they can also be used to analyse causal Time Invariant systems that have non-zero initial conditions). It is extensively used to solve linear differential equation problems as it converts such problems into algebraic equations in s-domain. The Laplace Transform (for continuous time functions) is obtained from the corresponding time domain signal as:

$$X(s) = \int_{-\infty}^{+\infty} x(t)e^{-st}dt \tag{1}$$

$$\text{where s} = \sigma + j\omega; \sigma \in \mathbb{R}, \omega \in \mathbb{R}$$

For linear constant coefficient Differential Equations, a further simplification is obtained. The transfer function for such systems are rational (= P(s)/Q(s) where P and Q are polynomials in s, Q is not identically 0). One may use scipy functions to then find the inverse of such a rational function:

1. scipy.signal.lti: Define the CT transfer function of an LTI system by specifying the numerator and denominator coefficients.

2. scipy.signal.impulse: Obtain the impulse response of a transfer function-computes inverse of a rational laplace transform

3. scipy.signal.lsim: Takes s-domain transfer function, time-domain input signal and time vector and returns time-domain output signal.

1.0.1 Note on Exponentially Decaying Sinusoids:

For exponentially decaying sinusoidal input functions, ie functions of the form $f(t) = cos(\omega t)e^{-at}u(t)$, the Laplace transform is:

$$F(s) = \frac{s+a}{(s+a)^2+\omega^2} = \frac{Num}{Num^2 + freq^2} \tag{2}$$

A Second Order Linear Differential Equation systems with constant coefficients can be represented in s domain as:

$$s^2X(s) + 2\zeta\omega_o sX(s) + \omega_o^2 X(s) = F(s) \tag{3}$$

where ω_o is the natural frequency of the system and ζ determines the damping:
$\zeta < 1 \Rightarrow$ underdamped;
$\zeta = 1 \Rightarrow$ critically damped;
$\zeta > 1 \Rightarrow$ overdamped

2 Code Structure

2.1 Important blocks used in the program

1: Find response of a second order linear system excited by an exponentially decaying sinusioid:

```
def exp_sin_resp(decay, zeta, freq, tvec):
 num = poly1d([1, decay])
 den = polyadd(polymul(num, num), poly1d([freq*freq]))
 den_x = poly1d([1, 2*zeta*freq, freq*freq])
 den = polymul(den, den_x)
 X = sp.lti(num.coeffs, den.coeffs)
 #num and den are poly1d objects, poly1d.coeffs is an array
 with the polymonial coefficient values
 t, x = sp.impulse(X, None, tvec)
 return 0
```

2: Generate Bode Plots

```
H = sp.lti([1], [1e-12, 1e-4, 1])
# H(s) for the LCR network is 1/(1 + sCR + s^2LC)
w, S, phi = H.bode()
# These may then be plotted using matplotlib.pyplot functions
```

3: Solve linear circuits for the complete response:

```
def LinCktResp(u, t, tname, fname):
 t, x, svec = sp.lsim(H, u, t)
 return 0
# Transient Response:
t_tran = arange(0, 1e-5, 1e-8)
u = cos(1e3*t_tran) - cos(1e6*t_tran)
LinCktResp(u, t_tran, "Transient Response", "./Images/LinCktTranResp.png")
# Complete Response (up to 10ms)
t_comp = arange(0, 1e-2, 1e-8)
u = cos(1e3*t_comp) - cos(1e6*t_comp)
LinCktResp(u, t_comp, "Complete Response", "./Images/LinCktCompResp.png")
```

4: Display the generated plots using OpenCV functions:

```
print("Do you want to view the plots? [y]/[Any other key]:")
press_key = input()
```

```
if (press_key == "y"):
 files = os.listdir('./Images')
 for f in files:
  cv2.imshow(str(f.split('.p')[0]), cv2.imread("./Images/" + str(f)))
  cv2.waitKey(0)
  cv2.destroyAllWindows()
```

2.2 Miscellaneous Details

The program saves the generated plots to the directory 'Images' in the same working directory.

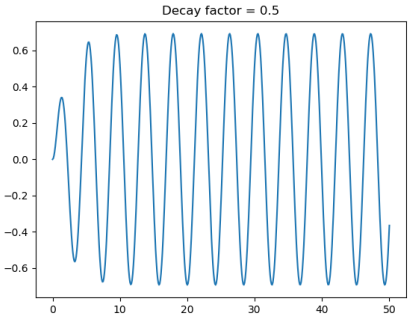
Format for running file on Terminal:

```
python3 EE18B104_Assign6_code.py
```

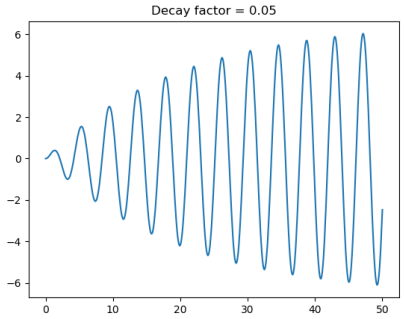
---

3 Plots

3.1 t-domain Response of a 2nd order system to an exponentially decaying sinusoid

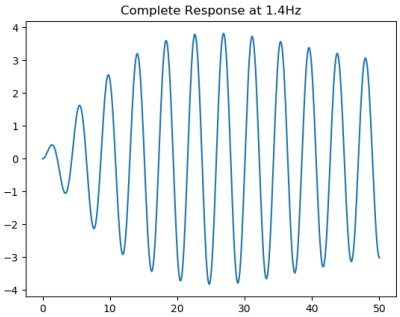


Parameters: $\zeta = 0$, decay factor = 0.5, frequency = 1.5 Hz

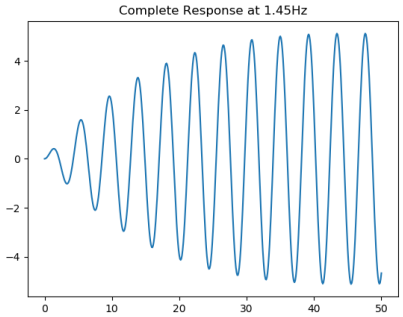


Parameters: $\zeta = 0$, decay factor = 0.05, frequency = 1.5 Hz

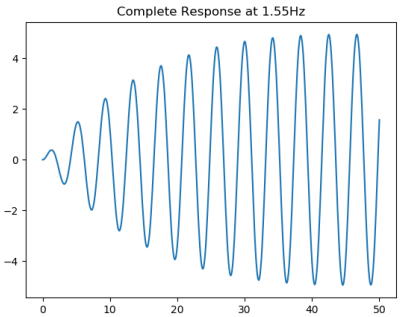
3.1.1 Frequency Sweep



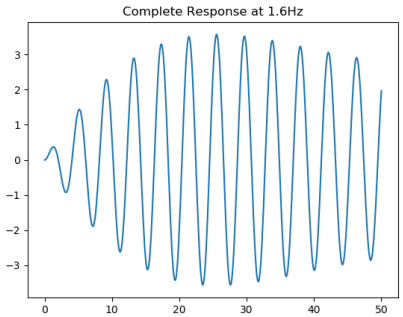
Parameters: $\zeta = 0$, decay factor = 0.05, frequency = 1.40 Hz



Parameters: $\zeta = 0$, decay factor = 0.05, frequency = 1.45 Hz



Parameters: $\zeta = 0$, decay factor = 0.05, frequency = 1.55 Hz



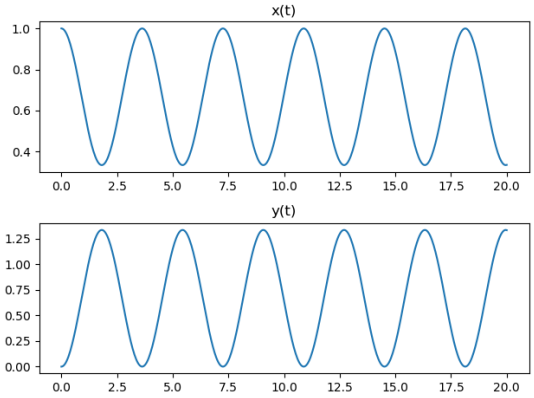
Parameters: $\zeta = 0$, decay factor = 0.05, frequency = 1.60 Hz

3.2 Coupled Spring Problem

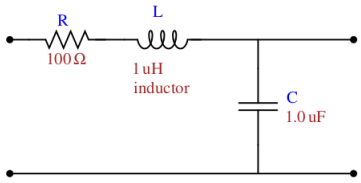
Solution of the following coupled differential equations:

$$\frac{d^2x(t)}{dt^2} + (x(t) - y(t)) = 0$$

$$\frac{d^2y(t)}{dt^2} + 2(y(t) - x(t)) = 0$$

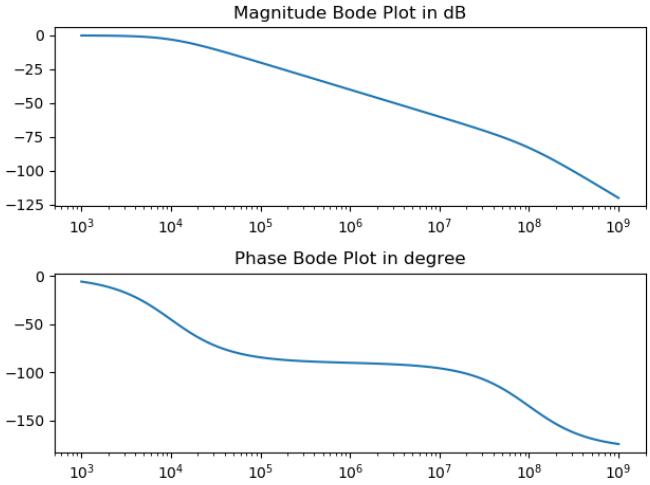


3.3 Analysis of Linear Circuit



Circuit Diagram

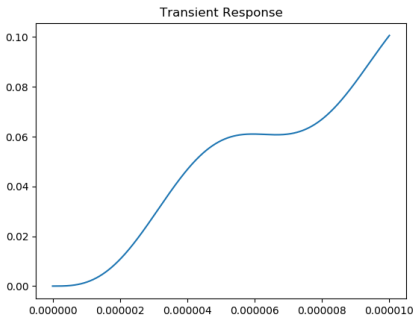
3.3.1 Transfer Function Characteristics



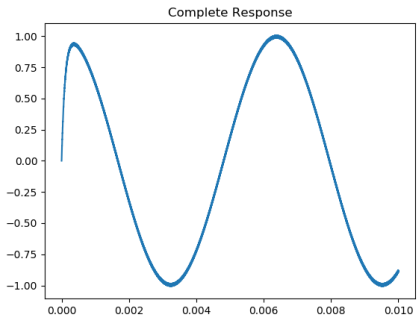
Bode Plot of the Circuit's Transfer Function

3.3.2 t-domain Response

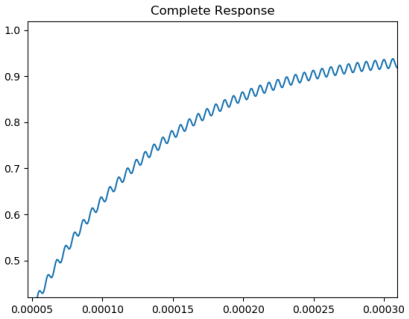
Input signal, x(t) = $\cos(10^3 t)$u(t) - $\cos(10^6 t)$u(t), where u(t) is the unit step function



Transient Response



Complete Response



Complete Response- Zoomed

---

4 Conclusion

4.1 Linear Second Order System

1. The effect of decay is clear from the two plots corresponding to decay = 0.5 and decay = 0.05.

2. Since the input signal decays, the steady state response is the natural (homogeneous/ZIR) response of the system. As the signal with decay = 0.05 takes longer to die down, the corresponding response reaches steady state at a slower pace.

3. The symbolic representation of the time domain response of the system to the given input signal was obtained using the following MATLAB command:

```
>> syms w0, syms s, syms w, syms a;
>> H(s) = (s+a)/((s+a)^2 + w^2);
>> X(s) = 1/(s^2 + w0^2);
>> Y(s) = X(s)*H(s);
>> ilaplace(Y);
```

4. The steady state part of the expression so obtained has the following form: $\frac{f(w0)}{(a^2+w^2-2ww0+w0^2)(a^2+w^2+2ww0+w0^2)}$

5. This explains the reason why the frequency sweep plots have lower steady state amplitude as the frequencies move away from 1.5Hz. Note that the natural frequency of the system is $\omega_o = 1.5$Hz.

6. Also note that for frequencies equidistant from 1.5Hz (1.4 and 1.6 or 1.45 and 1.55), the amplitude is higher for the lower frequency. This is because of the second factor in the denominator.

4.2 Response of the Linear Circuit

1. As is evident from the Bode Plot, the circuit has two poles.

2. A symbolic expression of the time-domain response of the system for the given input was obtained using the following MATLAB code:

```
>> syms s, syms t;
>> H(s) = 1/(1e-12s^2 + 1e-4s + 1);
>> x(t) = (cos(1e3*t)-cos(1e6*t))*heaviside(t);
% heaviside => unit step function
>> X(s) = laplace(x);
>> Y(s) = X(s)*H(s);
>> ilaplace(Y);
```

3. The steady state and transient state responses are respectively (A_1, A_2, B_1, B_2, c_1, c_2 are constants):

$$y_{ss}(t) = \frac{A_1 cos(1000t) + A_2 sin(1000t)}{c_1} - \frac{sin(1000000t)}{c_2}$$

$$y_{ts}(t) = B_1 exp(-5e7t)(cosh(7e6\sqrt{51}t) + B_2 sinh(7e6\sqrt{51}t)$$

4. The transients are short lived as is seen from the plots. Also the steady state response (oscillating at about 1000Hz) has small sinusoidal high frequency variations (of about 1000000 Hz). Both these phenomena are self-evident from the time domain response expression.

$\star\star\star$