Project: Band Reject Filter

EE 321
April 24, 2020

Kevin Evans
ID: 11571810

I State model derivation

From the circuit, we can relate the currents at the node v_o using KCL,

$$\begin{aligned}\frac{v_o}{R_L} + i_2 &= \frac{v_i - v_o}{R_1} + i_1 \\ \left(\frac{1}{R_L} + \frac{1}{R_1}\right) v_o &= \frac{v_i}{R_1} + i_1 - i_2 \\ v_o &= \frac{R_L R_1}{R_L + R_1}\left(\frac{v_i}{R_1} + i_1 - i_2\right) \end{aligned} \tag{1}$$

Next, we can relate the relative voltages with KVL,

$$\begin{aligned} v_o &= v_c + v_2 \\ &= v_c + L_2 \frac{\mathrm{d}i_2}{\mathrm{d}t} \end{aligned} \tag{2}$$

$$\begin{aligned} v_o &= v_i - v_1 \\ &= v_i - L_1 \frac{\mathrm{d}i_1}{\mathrm{d}t} \end{aligned} \tag{3}$$

Combining the equations (1) with (2) and (3), then solving for the differential results in

$$\frac{\mathrm{d}i_1}{\mathrm{d}t} = \frac{1}{L_1}\left[\left(1 - \frac{R_L}{R_L + R_1}\right) v_i - \frac{R_L R_1}{R_L + R_1} i_1 + \frac{R_L R_1}{R_L + R_1} i_2\right] \tag{4}$$

$$\frac{\mathrm{d}i_2}{\mathrm{d}t} = \frac{1}{L_2}\left(\frac{R_L}{R_L + R_1} v_i + \frac{R_L R_1}{R_L + R_1} i_1 - \frac{R_L R_1}{R_L + R_1} i_2 - v_c\right) \tag{5}$$

Lastly, we can equate the second inductor current i_2 with the capacitor current,

$$\frac{\mathrm{d}v_c}{\mathrm{d}t} = \frac{1}{C} i_2 \tag{6}$$

At this point, the differentials from the inductors and capacitors have been related and can be assembled into a state matrix, where the state matrix represents the inductor currents i_1 and i_2, as well as the capacitor voltage v_c through time. The elements of the differential matrix $\dot{\mathbf{X}}$ is then equated to eq. (4), (5), and (6), resulting in

$$\dot{\mathbf{X}} = \mathbf{A}\mathbf{X} + \mathbf{B}\mathbf{G}$$

$$\begin{bmatrix} \frac{\mathrm{d}i_1}{\mathrm{d}t} \\ \frac{\mathrm{d}i_2}{\mathrm{d}t} \\ \frac{\mathrm{d}v_c}{\mathrm{d}t} \end{bmatrix} = \begin{bmatrix} -\frac{1}{L_1}\left(\frac{R_L R_1}{R_L + R_1}\right) & \frac{1}{L_1}\left(\frac{R_L R_1}{R_L + R_1}\right) & 0 \\ \frac{1}{L_2}\left(\frac{R_L R_1}{R_L + R_1}\right) & -\frac{1}{L_2}\left(\frac{R_L R_1}{R_L + R_1}\right) & -\frac{1}{L_2} \\ 0 & 1/C & 0 \end{bmatrix} \begin{bmatrix} i_1 \\ i_2 \\ v_c \end{bmatrix} + \begin{bmatrix} \frac{1}{L_1}\left(1 - \frac{R_L}{R_L + R_1}\right) \\ \frac{1}{L_2}\left(\frac{R_L}{R_L + R_1}\right) \\ 0 \end{bmatrix} v_i(t)$$

The output matrix has a singular element of v_o, which is trivially found through the currents passing through the load resistor, effectively by rearranging (1).

$$\mathbf{Y} = \mathbf{C}\mathbf{X} + \mathbf{D}\mathbf{G}$$

$$[v_o] = \begin{bmatrix} \frac{R_L R_1}{R_L + R_1} & -\frac{R_L R_1}{R_L + R_1} & 0 \end{bmatrix} \mathbf{X} + \begin{bmatrix} \frac{R_L}{R_L + R_1} \end{bmatrix} v_i(t)$$

MATLAB numeric estimations

Using MATLAB, we can use the control systems toolbox to find the transfer function of this state matrix numerically. If we input the matrices into variables, the `ss` and `tf` commands may be used to numerically determine the transfer function $H(s)$. The resulting plots are shown below in Figure 1 and 2, using the code from Snippet 1.

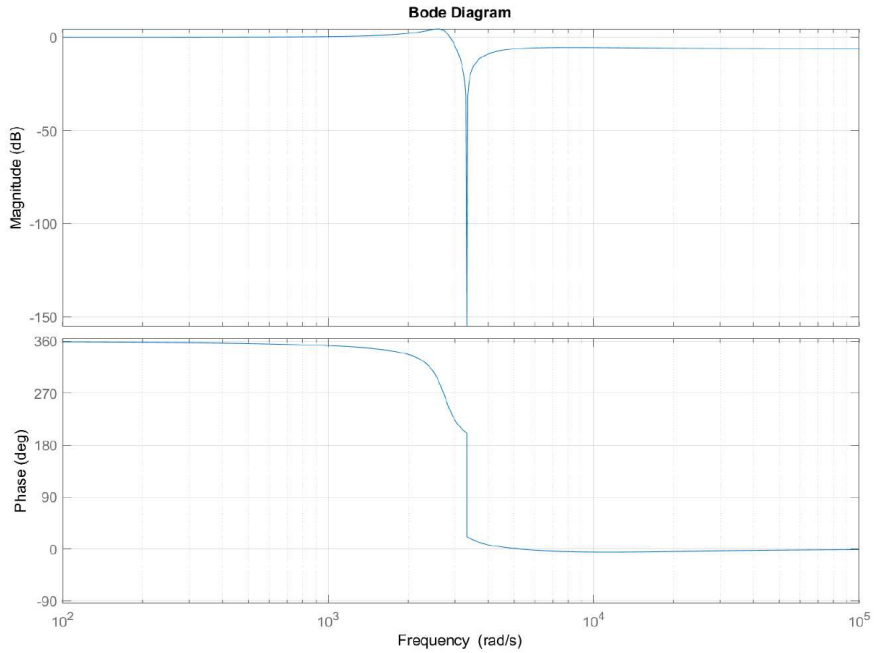

Figure 1: Bode diagram of the transfer function calculated using the state space matrices.

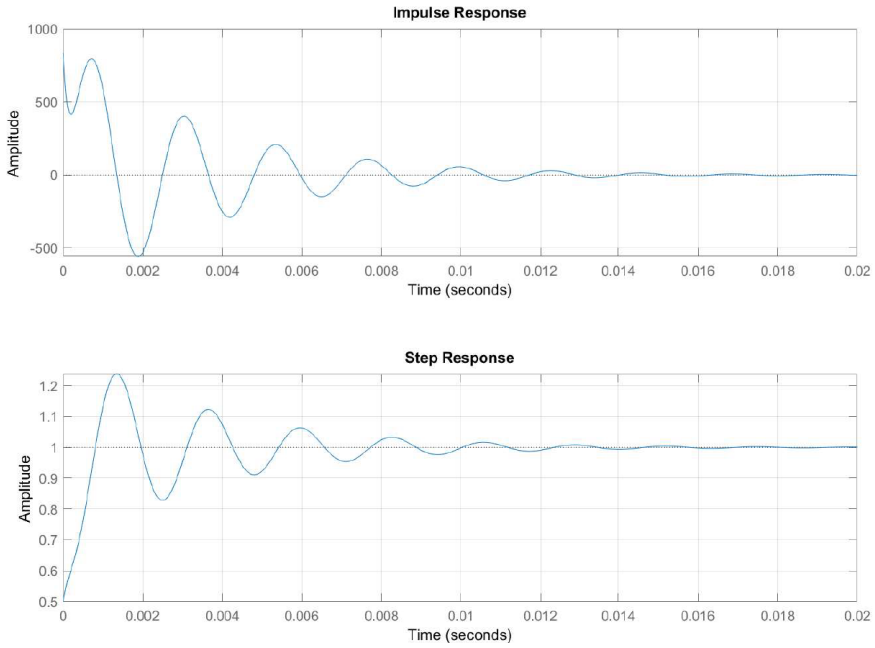

Figure 2: Time domain responses of the transfer function calculated using the state space matrices.

Code Snippet 1: MATLAB code for solving for the transfer function using state space equations.

```
A = [ -4166.67,   4166.67,    0;
       2500.00,  -2500.00, -100;
          0.00, 109890.10,    0];
B = [83.33;
     50.00;
      0.00];
C = [25, -25, 0];
D = [0.5];

% solve system and system->tf
system = ss(A, B, C, D);
H      = tf(system);

% plot a bode plot
bode(H);

% plot: impulse, step, and bode plots
%subplot(2, 1, 1);
figure;
impulse(H);
grid on;

%subplot(2, 1, 2);
figure;
step(H);
grid on;

%subplot(4, 1, [3, 4]);
figure;
bode(H);
grid on;
```

II Impulse and step response

We can convert the circuit to the s-domain and use a simple voltage divider to find the transfer function $H(s)$. First, the components are converted to their s-domain equivalent, as shown in Figure 3(a). Next, the circuit is reduced, allowing us to use a simple voltage divider to calculate the transfer function. This is shown in Figure 3(b). The impedances can be calculated as

$$\begin{aligned} Z_A &= (R_1 \parallel sL_1) = \left[50 \parallel s\left(6 \times 10^{-3}\right)\right] \\ &= \frac{150s}{25000 + 3s} \\ Z_B &= [R_L \parallel (Z_{L2} + Z_C)] \\ &= 50 \parallel \left[s\left(10 \times 10^{-3}\right) + \frac{1}{s\left(9.1 \times 10^{-6}\right)}\right] \\ &= \frac{50\left(s^2 + 1.0989 \times 10^7\right)}{s^2 + 5000s + 1.0989 \times 10^7} \end{aligned}$$

From the voltage divider technique, the s-domain transfer function is found as

$$\begin{aligned} H(s) &= \frac{Z_B}{Z_A + Z_B} \\ &= \frac{0.5s^3 + 4166.67s^2 + 5494500s + 4.578\,75 \times 10^{10}}{s^3 + 6666.67s^2 + 10989000s + 4.578\,75 \times 10^{10}} \end{aligned} \tag{7}$$

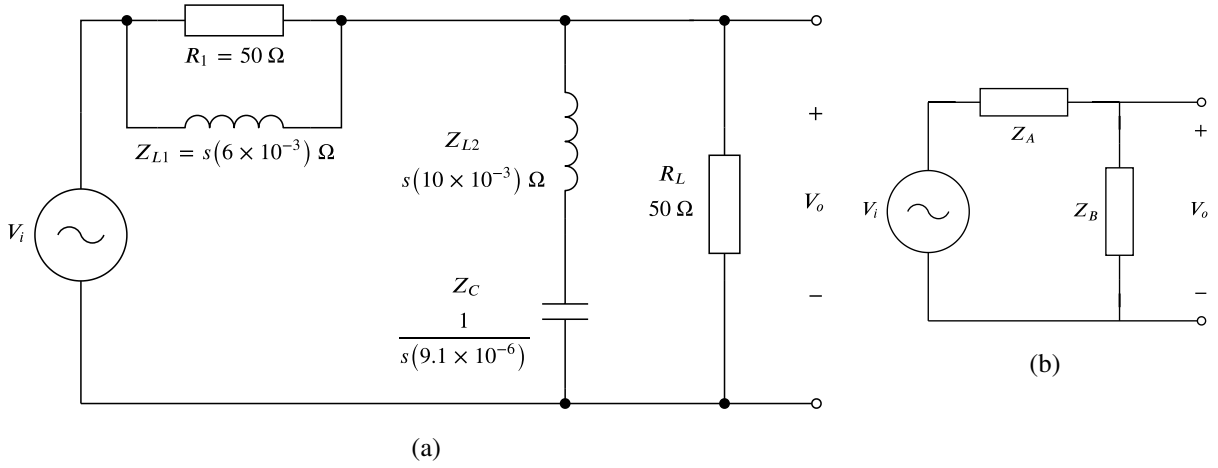

Figure 3: Circuit diagrams of the filter (a) in the s-domain and (b) reduced to two impedances.

Factoring using MATLAB

This transfer function was entered into a MATLAB script to determine the zero and pole locations, as well as the residues. Using the `zpkdata` and `residue` functions, the transfer function was factored using Code Snippet 2 as

$$H(s) = 0.5 + \frac{1307.4886}{s + 6096.1298} - \frac{237.0768 - j418.2}{s + 285.2701 - j2725.7171} - \frac{237.0768 + j418.2}{s + 285.2701 + j2725.7171}$$

From the MATLAB code, the poles and zeros are found to be:

- Poles: $s_{p1} = -6.0961$
  $s_{p2} = -0.2853 + j2.7257$
  $s_{p3} = s_{p2}^* = -0.2853 - j2.7257$

- Zeros: $s_{z1} = -8333.33$
  $s_{z2} = j3315$
  $s_{z3} = s_{z2}^* = -j3315$

Code Snippet 2: MATLAB code for factoring the transfer function.

```
% coefficients of the transfer function:
% order:
% s^    3,       2,        1,          0
N = [0.5, 4166.67,  5494500, 4.57875e10]; % numerator
D = [  1, 6666.67, 10989000, 4.57875e10]; % denominator

% generate transfer function, H(s)
H            = tf(N, D);
[z, p,    k] = zpkdata(H, 'v');  % zeros, poles, gain
[r, ptmp, k] = residue(N, D);    % residues r -> are the K_i values

% print out the residues over the poles
for i = 1 : 3
        fprintf("\n%d -> %s\n\t----------------------\n\t%s\n", ...
                i, num2str(r(i)), num2str(p(i)) );
end
```

2.1 Impulse Response

If we transform the unit impulse function $\delta(t)$ to the s-domain, we can multiply it by the transfer function to determine the response of the circuit. As the impulse function simply is 1 in the s-domain,

$$\begin{aligned} Y_\delta(s) &= H(s)X(s) \\ &= 0.5 + \frac{1307.4886}{s + 6096.1298} - \frac{237.0768 - j418.2}{s + 285.2701 - j2725.7171} - \frac{237.0768 + j418.2}{s + 285.2701 + j2725.7171} \end{aligned}$$

We can take the inverse Laplace transform to get the response in the time domain. The first term, 0.5, becomes the coefficient to another impulse function and is negligible. For the last two terms, which have conjugates in both the denominator and numerator, we can use Table 12.3 from Nilsson/Riedel (p. 469) to transform the terms into a cosine function,

$$\mathcal{L}^{-1}\left\{Y_{\delta(s)}\right\} = 1307.4886e^{-6096.1298t} - 2|237.0768 + j418.2|e^{-285.2701t}\cos\left(2725.7171t + \tan\left(\frac{418.2}{2725.7171}\right)\right)$$

$$y_\delta(t) = 1307.4886e^{-6096.1298t} - 961.45e^{-285.2701t}\cos(2725.7171t + 60.45°)$$

As shown in Figure 4 below, plotting this function using MATLAB, we can see that the impulse response is identical to the response from Part I. The code used to plot this is shown in the Snippet 3 below.

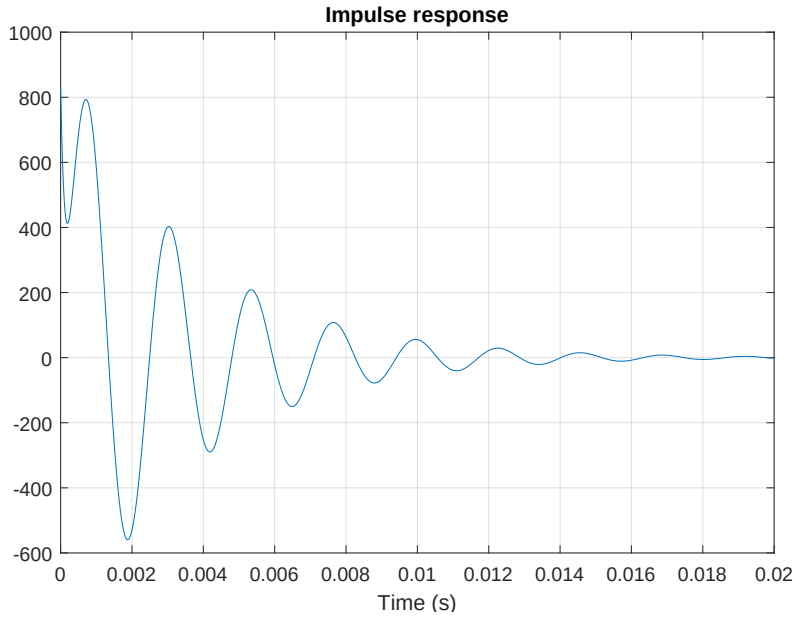

Figure 4: Plot of the impulse response in the time domain.

Code Snippet 3: MATLAB code for plotting the impulse response.

```
dt = 5e-6;
t  = (0:dt:20e-3);
y  = 1307.4886 .* exp(-6096.13 .* t) ...
     - 2 .* 480.725 .* exp(-285.2701.*t) .* cos(2725.7171.*t + deg2rad(60.45));

figure;
plot(t, y);
title('Impulse response');
xlabel('Time (s)');
ylabel('Amplitude (V)');
grid on;
```

2.2 Step Response

Next, the step response can be calculated using the transfer function found earlier. Starting from (7), we can find the step response first in the s-domain,

$$\begin{aligned} Y(s) &= X(s)H(s) \\ &= \frac{1}{s}H(s) \\ &= \frac{0.5s^3 + 4166.67s^2 + 5494500s + 4.578\,75 \times 10^{10}}{s^4 + 6666.67s^3 + 10989000s^2 + 4.578\,75 \times 10^{10}s} \end{aligned}$$

The output function $Y(s)$ remained the same, but with only the order of the denominator incremented. The transfer function was factored using the same MATLAB code as the impulse function, Snippet 2, and was shown to be

$$Y(s) = \frac{1}{s} - \frac{0.2145}{s + 6096.1298} - \frac{0.1428 - j0.1019}{s + 285.2701 + j2725.7171} - \frac{0.1428 + j0.1019}{s + 285.2701 - j2725.7171}$$

If we take the inverse Laplace transform, we can find the output in the time domain $y(t)$. This uses the same transformation pairs listed in Table 12.3 of Nilsson/Riedel.

$$\begin{aligned} y(t) &= \mathcal{L}^{-1}\{Y(s)\} \\ &= 1.0 - 0.2145e^{-6096.1t} + 0.350858e^{-285.3t}\cos(2725.7t + 144.489^\circ) \end{aligned}$$

If we plot the response in the time domain using MATLAB, we can see that this is equivalent to the plots from Part I. This plot is shown in Figure 5 using the code from Snippet 4.

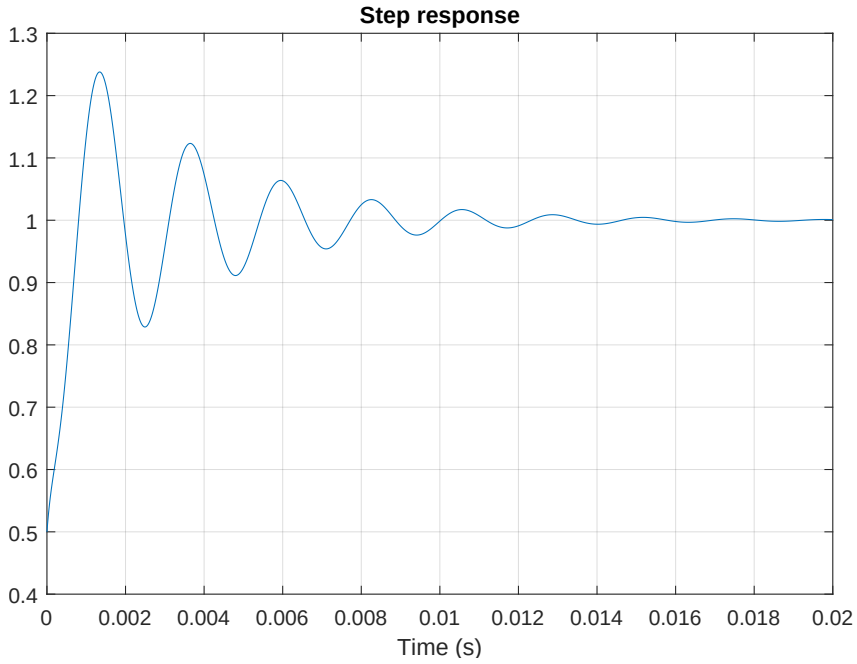

Figure 5: Plot of the step response in the time domain.

Code Snippet 4: MATLAB code for plotting the step response.

```
dt = 5e-6;
t  = (0:dt:20e-3);
y = 1.0 + -0.2145 .* exp(-6.0961e3 .* t) ...
    + 2 .* 0.175429 .* exp(-0.2853e3 .* t) .* cos(2.7257e3 .* t + deg2rad(144.489));
figure;
plot(t, y);
title('Step response');
xlabel('Time (s)');
ylabel('Amplitude (V)');
grid on;
```

2.3 Frequency response

From the original transfer function calculation, eq. (7), we can use the `bode` function in MATLAB to plot this frequency response of the circuit. This was accomplished by appending `bode(H);` to the MATLAB code in Snippet 2, resulting in the plot shown in Figure 6.

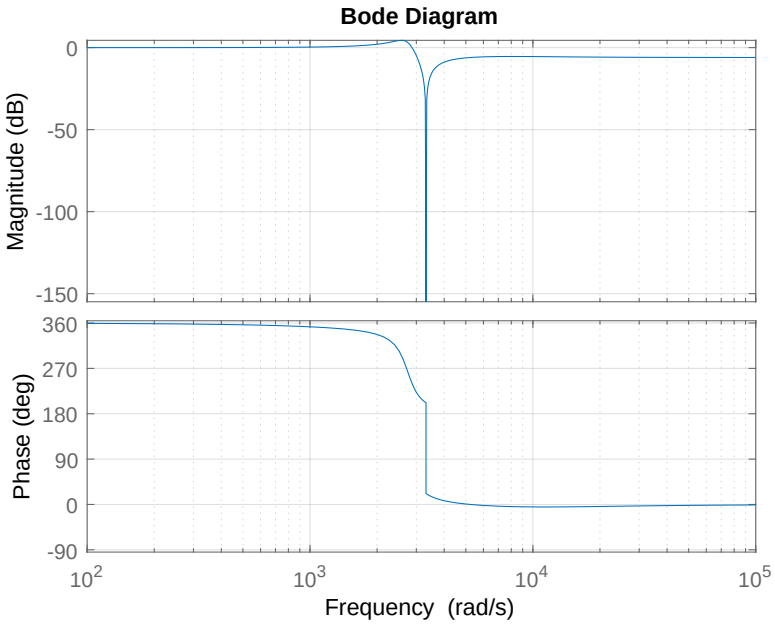

Figure 6: Plot of the step response in the time domain.

III Differential expression

From Part I, it is clear that using the output matrix $\mathbf{Y}$ containing v_o,

$$v_o = \frac{R_1 R_L}{R_1 + R_L}(i_1 - i_2) + \frac{R_L}{R_1 + R_L} v_i$$

Where v_o, i_1, i_2, and v_i are all functions of t. If we take the third derivative of both sides,

$$\dddot{v_o} = \frac{R_1 R_L}{R_1 + R_L}\left(\dddot{i_1} - \dddot{i_2}\right) + \frac{R_L}{R_1 + R_L}\dddot{v_i} \tag{*}$$

If we related the inductor currents to the input and output voltages,

$$\begin{aligned} v_o &= v_i - L_1 \dot{i_1} \\ \frac{v_o}{R_L} + i_2 &= \frac{v_i - v_o}{R_1} + i_1 \end{aligned}$$

Continually differentiating these and rearranging results in

$$\begin{aligned} \dddot{i_1} &= \frac{\dddot{v_i}}{L_1} - \frac{\dddot{v_o}}{L_1} \\ \dddot{i_2} &= \frac{1}{L_2}\left(\ddot{v_o} - \frac{1}{C}\dot{i_2}\right) \\ &= \frac{\ddot{v_o}}{L_2} - \frac{1}{L_2 C}\left[\frac{1}{R_1}\dot{v_i} - \left(\frac{1}{R_1} + \frac{1}{R_L}\right)\dot{v_o} + \frac{v_i}{L_1} - \frac{v_o}{L_1}\right] \end{aligned}$$

If we now substitute $\dddot{i_1}$ and $\dddot{i_2}$ into (*) and rearrange terms,

$$\begin{aligned} \dddot{v_o} + \frac{R_1 R_L}{R_1 + R_L}\left[\left(\frac{1}{L_1} + \frac{1}{L_2}\right)\ddot{v_o} + \frac{1}{L_2 C}\left(\frac{1}{R_1} + \frac{1}{R_L}\right)\dot{v_o} + \frac{1}{L_2 C L_1} v_o\right] \\ = \frac{R_1 R_L}{R_1 + R_L}\left[\frac{1}{L_1}\ddot{v_i} + \frac{1}{L_2 C R_1}\dot{v_i} + \frac{1}{L_2 C L_1} v_i\right] + \frac{R_L}{R_1 + R_L}\dddot{v_i} \end{aligned}$$

Simplifying it some more, it matches the given differential equation,

$$\begin{aligned} \dddot{v_o} + \frac{R_1 R_L}{R_1 + R_L}\left(\frac{1}{L_1} + \frac{1}{L_2}\right)\ddot{v_o} + \frac{1}{L_2 C}\dot{v_o} + \frac{R_1 R_L}{R_1 + R_L}\frac{1}{L_1 L_2 C} v_o \\ = \frac{R_L}{R_1 + R_L}\left[\dddot{v_i} + \frac{R_1}{L_1}\ddot{v_i} + \frac{1}{L_2 C}\dot{v_i} + \frac{R_1}{L_1 L_2 C} v_i\right] \quad \square \end{aligned}$$

IV Numeric convolution

Using MATLAB, we can estimate the convolution using the `conv` function over discrete times. The numeric convolution results in plots that are nearly identical to the plots found earlier. Shown in Figures 7 and 8 are the step and impulse responses found using the numeric convolution. The MATLAB code used to generate these plots is listed in Snippet 5.

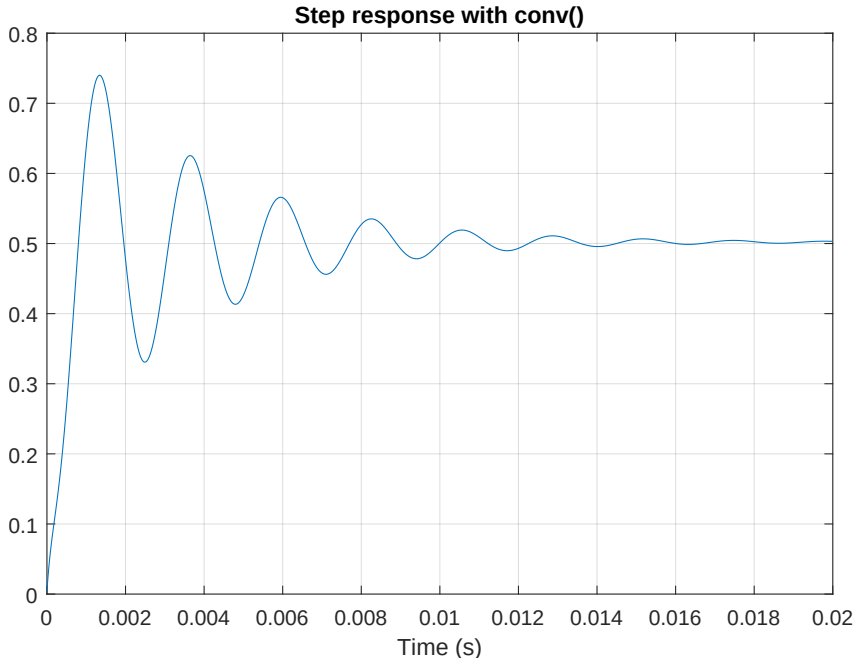

Figure 7: Step response of the transfer function using MATLAB's `conv` command.

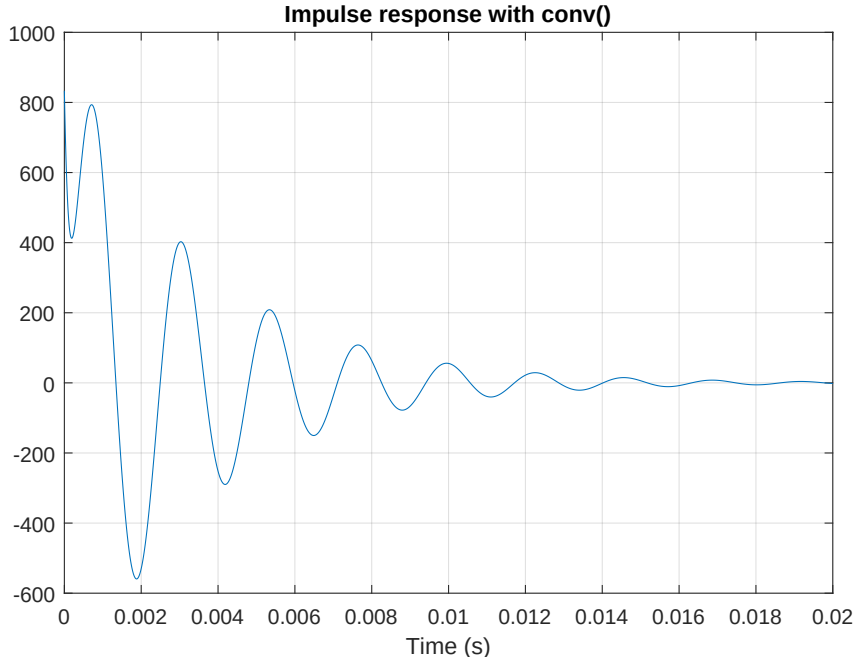

Figure 8: Impulse response of the transfer function using MATLAB's `conv` command.

Code Snippet 5: MATLAB code for the convolutions.

```
% reusing the transfer function from Part II.
dt = 5e-6;
t  = (0:dt:20e-3);
h  = 1307.4886 .* exp(-6096.13 .* t) ...
     - 2 .* 480.725 .* exp(-285.2701.*t) .* cos(2725.7171.*t + deg2rad(60.45));

% input functions:
u  = ones(1, 2 * length(t));       % unit step, filled with 1
d  = [1/dt zeros(1, length(t)-1)]; % impulse, a spike then filled with 0

% numerically find the convolutions for both
s  = conv(h, u) * dt; % unit step
s2 = conv(h, d) * dt; % impulse

figure;
plot(t, s(1:length(t)));
title('Step response with conv()');
xlabel('Time (s)');
ylabel('Amplitude (V)');
grid on;

figure;
plot(t, s2(1:length(t)));
title('Impulse response with conv()');
xlabel('Time (s)');
ylabel('Amplitude (V)');
grid on;
```