San Francisco State University

Engineering 305

Laboratory #8 – Laplace Transform

Purpose

In this laboratory, you will Laplace transforms and their relation to pole-zero plots and impulse responses.

Background reading includes:

- Lathi, Chapter 5
- Holton notes, Unit 7

Background

In this lab we examine important features of the bilateral Laplace transform. The Laplace transform can be considered to be a generalized version of the Fourier transform that is able to describe stable and unstable, causal and non-causal systems.

Definition

The bilateral Laplace transform of a signal $h(t)$ defined as

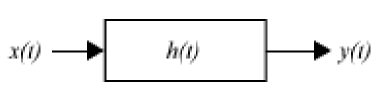
$$H(s) = \int_{t=-\infty}^{\infty} h(t)e^{-st}dt\ . \qquad \text{(L7.1)}$$

For a causal (right-sided) time signal, the region of convergence (ROC) of the Laplace transform, defined as the values of *s* for which the closed form of the Laplace transform exists, is always a right-hemiplane.

$h(t)$	$H(s)$	ROC	
$e^{-\alpha t}u(t)$	$\dfrac{1}{s+\alpha}$	$Re\{s\} > -\alpha$	(L7.2)

Using Laplace transforms to analyze input-output relations of systems

Laplace transforms is particularly useful in solving the input-output relation of systems. Consider a linear time-invariant system with input $x(t)$, output $y(t)$ and impulse response, $h(t)$



As we've seen before, the input-output relation of the system is defined by the convolution relation

$$y(t) = x(t) * h(t) \qquad \text{(L7.3)}$$

which translates in the Laplace domain to

$$Y(s) = X(s)H(s)\,, \qquad \text{(L7.4)}$$

A majority of the systems that are of interest to us are defined by linear constant coefficient differential equations of the form

$$\sum_{n=0}^{N} a_n \frac{d^n y}{dt^n} = \sum_{m=0}^{M} b_m \frac{d^m x}{dt^m} . \tag{L7.5}$$

Recalling the derivative properties of Laplace transforms,

$$\mathcal{L}\left\{\frac{d^n y}{dt^n}\right\} = s^n Y(s) . \tag{L7.6}$$

Taking the derivative of both sides of Equation (L7.5) and applying Equation (L7.6), we get

$$\begin{aligned} \mathcal{L}\left\{\sum_{n=0}^{N} a_n \frac{d^n y}{dt^n}\right\} &= \mathcal{L}\left\{\sum_{m=0}^{M} b_m \frac{d^m x}{dt^m}\right\} \\ \sum_{n=0}^{N} a_n \mathcal{L}\left\{\frac{d^n y}{dt^n}\right\} &= \sum_{m=0}^{M} b_m \mathcal{L}\left\{\frac{d^m x}{dt^m}\right\} \\ \sum_{n=0}^{N} a_n s^n Y(s) &= \sum_{m=0}^{M} b_m s^m X(s) \end{aligned} \tag{L7.7}$$

so, applying Equation (L7.4),

$$H(s) = \frac{Y(s)}{X(s)} = \frac{\sum_{m=0}^{M} b_m s^m}{\sum_{n=0}^{N} a_n s^n} . \tag{L7.8}$$

For systems described by linear constant coefficient differential equations, $H(s)$ can be expressed as a ratio of polynomials in s. The roots of the nominator polynomial are called the *zeros* of $H(s)$. The roots of the denominator polynomial are called the *poles* of $H(s)$.

For example, consider the differential equation

$$y''(t) + 4y'(t) + 3y(t) = 3x'(t) . \tag{L7.9}$$

The Laplace transform is

$$H(s) = \frac{3s}{s^2 + 4s + 3} = \frac{3s}{(s+1)(s+3)} = \frac{-\frac{3}{2}}{(s+1)} + \frac{\frac{9}{2}}{(s+3)} , \tag{L7.10}$$

where we have performed the partial fraction expansion to separate $H(s)$ into the sum of terms. If we stick to causal systems, then the ROC is always given by the half-plane bounded by the right-most pole, i.e. $\mathrm{Re}\{s\} > -1$. The pole-zero plot is shown below.

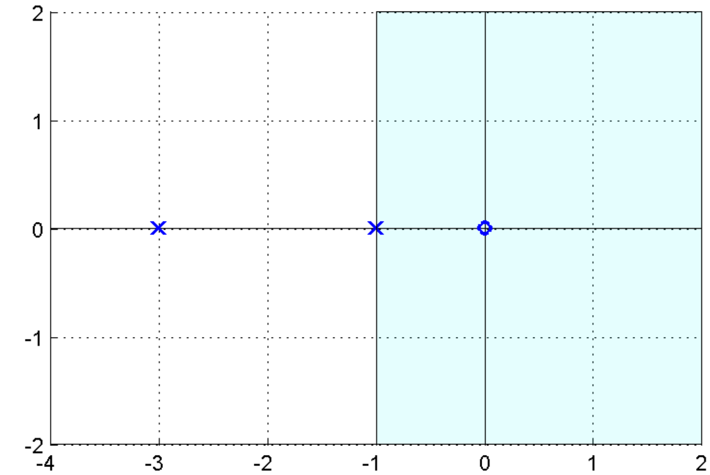


Figure 1

Therefore, the ROCs of the two sum terms must be such that their intersection is also $\mathrm{Re}\{s\} > -1$:

$$H(s) = \underbrace{\frac{-\frac{3}{2}}{(s+1)}}_{\mathrm{Re}\{s\}>-1} \cap \underbrace{+\frac{\frac{9}{2}}{(s+3)}}_{\mathrm{Re}\{s\}>-3},$$

from which we conclude that the inverse Laplace transform is

$$h(t) = -\tfrac{3}{2}e^{-t}u(t) + \tfrac{9}{2}e^{-3t}u(t).$$

This is shown below:

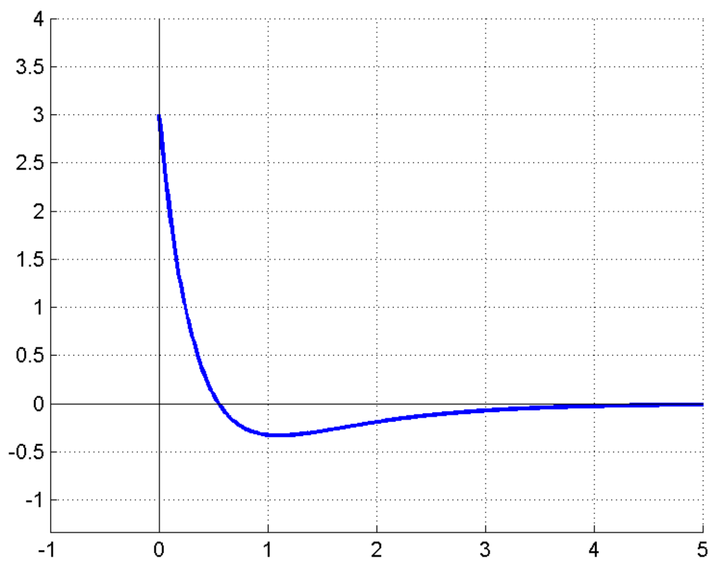


Figure 2

Because the region of convergence includes the $j\omega$-axis, $h(t)$ is absolutely integrable and corresponds to the impulse response of a stable system.

Assignment

In this assignment, we will investigate the pole-zero plot and the inverse transform. In all cases, you will assume that the system is causal, so the ROC is a right-half plane bounded by the right-most pole.

Pole-zero plot

You will write a Matlab function, `pzplot`, that will create a pole-zero plot for $H(s)$, including region of convergence. For any $H(s)$ that is described as the ratio of polynomials, as shown in Equation (L7.8), we can completely specify $H(s)$ by two Matlab vectors corresponding to the coefficients of the numerator polynomial, b_m, and the denominator polynomial, a_n in Equation (L7.8). In the example of Equation (L7.9), the vectors are

```
a = [1 4 3];
b = [3 0];
```

Here's the complete specification of the function

```
function pzplot(b, a)
% PZPLOT  Plot pole-zero plot with region of convergence
%       PZPLOT(B, A) plots the pole-zero plot of the filter B/A, %
with the region of convergence shaded:
%
%                        nb-1         nb-2
%             B(s)   b(1)s     +  b(2)s     + ... +  b(nb)
%       H(s) = ---- = ------------------------------------
%                        na-1         na-2
%             A(s)   a(1)s     +  a(2)s     + ... +  a(na)
%
%       where the numerator and denominator coefficients are given in
%       vectors B and A.
```

The example of the pole-zero above was created by `pzplot(b, a)`.

A few practical details:

- You can determine the roots of the denominator and numerator polynomials using Matlab's `roots` command. You may have complex roots and/or multiple roots (i.e. multiple poles or zeros) that you need to deal with.
- If you have a complex number, `z`, you can plot it in Matlab with `plot(real(z), imag(z), 'o')` or even easier, you can just say `plot(z, 'o')`. The latter won't work the way you expect if `z` is purely real or purely imaginary. However, but you can turn any real or imaginary number into a complex number by simply adding a miniscule real or imaginary part. For example, if `z` is purely real, `z+j*eps` is complex. Of course, you can do something similar for a purely imaginary number.
- You can create a shaded area for the region of convergence with Matlab's `patch` command. For example to create a cyan-shaded rectangle specified by lower-left coordinates (1, 2) and upper right coordinates (3, 4), you would call `patch([1 1 2 2], [3 4 4 3], 'c', 'FaceAlpha', 0.1)`. The `'FaceAlpha'` parameter is necessary to make the portion of the pole-zero plot under the patch transparent, so you can see the axis and possibly singularities (i.e. zeros).
- In order to make the pole-zero plot look pleasing, you should use Matlab's `axis equal` command. You should then use the `axis` command to size your plot such that the edges of the plot are at least

one unit away from the closest singularity, with a minimum range of $-2 < \text{Re}\{s\} < 2$ and $-2 < \text{Im}\{s\} < 2$. For example, for the plot shown in Figure 1, the value of the abscissa on the left side (-3) is determined by the position of the left-most pole minus one. The value of the abscissa on the right-hand side is determined by the minimum allowed value (2). The ordinate is determined by the minimum allowed values ($-2 < \text{Im}\{s\} < 2$).

- If you wish, you can adjust what numbers Matlab plots on the x- and y-axes with `set(gca, 'XTick', …)` and `set(gca, 'YTick', …)`.

Impulse response

You will write a Matlab function that plots the inverse transform of $H(s)$ described by polynomials b_m and a_n in Equation (L7.8) plus a single point in the region of convergence. Here's a specification of the function:

```
function hplot(b, a)
% HPLOT  Plot impulse response
%      HPLOT(B, A) plots the pole-zero plot of the filter B/A:
%
%                            nb-1         nb-2
%             B(s)   b(1)s     +  b(2)s     + ... +  b(nb)
%      H(s) = ---- = ------------------------------------
%                            na-1         na-2
%             A(s)   a(1)s     +  a(2)s     + ... +  a(na)
%
%      B and A are the numerator and denominator coefficients.
```

This function relies on the use of Matlab's `residue` function. You should read the help file for this function and understand what it does. For example, in the example of Figure 1,

```
» [r, p, k] = residue(b, a)

r =
    4.5000
   -1.5000

p =
    -3
    -1

k =
     []
```

The p vector represents the poles, which can be repeated or complex. The `r` vector represents the "residues", that is, the scalar values that form the numerator of the two terms derived from partial fraction expansion. In this laboratory, we will assume that the order of the numerator is always less than the order of the denominator (i.e. the length of `b` is less than the length of `a`). This means that every $H(s)$ can be separated by partial fraction expansion as the sum of terms each of which corresponds to a simple or repeated pole; the `k` vector returned by the `residue` function will be empty (why?).

The example of the impulse response column, above, was created by `hplot (b, a)`.

A few practical details:

- You should use the formula given in Equation (L7.2) to determine the response components for each pole. These formulas work for both simple roots and repeated roots whether real or complex.

- Even if a roots are complex, you expect the sum of the terms that comprises you inverse transform to be real.
- You should adjust the scale of your plot so that the abscissa ranges from. The vertical dimension of the plot should be sized so that the edges of the plot are at least one unit away from the maximum and minimum values of $h(t)$; however, the plot should have a maximum vertical scale of ± 10 and a minimum scale of ± 1.

Response of various systems

We will now use your `pzplot` and `hplot` functions to examine the response of various systems.

- Download this little program, `plotit`, and put it in your working directory. This program just calls your `pzplot` and `hplot` functions with values of `b`, `a` and `roc` that you specify.
- Use this program to provide output for the system defined by Equation (L7.9). Your plots should look similar to those of Figure 1and Figure 2.
- Provide output for systems defined by the following differential equations.
  - $y'' + y = x' - 2x$
  - $y'' + 2y' + 26y = x' + 2x$
  - $y'' + 4y' + 5y = 5x$
  - $y'' + 16y = 16x$ (this system is unstable (why?)

Response of second-order system

Finally, we consider a very important example: the second order system. In lecture and previous labs, we have discussed the classic second-order system defined by the differential equation

$$y'' + 2\zeta\omega_0 y' + \omega_0^2 y = \omega_0^2 x, \qquad \text{(L7.11)}$$

where ω_0 is the undamped (natural) frequency and ζ is the damping factor. In the foregoing, we consider only *causal* systems, whether they are stable or not. What does this say about the region of convergence with respect to the position of the poles?

- In this part of the problem, we will explore what happens when we fix the natural frequency and vary the damping factor. Set $\omega_0 = 2$. Now, produce small plots as you vary ζ from 2 (overdamped), 1 (critically damped), 0.5, 0.1 (underdamped), and 0 (undamped). Note that
  - When $\zeta > 1$, the poles are both real.
  - When $\zeta = 1$, there is a double real pole at $\omega_0 = 2$.
  - When $\zeta < 1$, the poles are complex and lie on a circle of radius $\omega_0 = 2$. As ζ decreases towards zero, the poles move towards the imaginary axis.
  - When $\zeta = 0$, the poles are purely imaginary.
- It is actually possible to implement unstable circuits with a negative damping factor. This is called *positive feedback*. Note what happens when you choose $\omega_0 = 2$ and $\zeta = -0.2$.
- Now we will fix the damping factor at $\zeta = 0.5$ and vary ω_0. Choose values of $\omega_0 = 1$, 2 and 4. Note that the angle of the poles with respect to the real axis doesn't change; only the position of the poles with respect to the point $s = 0$.