Lab 2 System Analysis in Time Domain

Objective

- Get to know the zero-input and zero-state response of the LTI system.
- The application of the linear time-invariant property of LTI systems presupposes a zero initial state.
- Understand the definition of impulse responses and step responses and their significance for LTI system analysis.
- Explore the relationship between LTI systems, impulse responses, and system responses.

Content

Differential Equation Describing the System

A linear continuous time-invariant system can always be represented by a linear differential equation with constant coefficients, which is shown as below:

$$\begin{cases} \sum_{i=0}^{N} a_i y^{(i)}(t) = \sum_{j=0}^{M} b_j f^{(j)}(t) & \text{differential equation expression} \\ y^{(k)}(0) = ..., \ (k = 0, \cdots, N-1) & \text{initial conditions, y(t) is the output of the system} \\ f(t) = ..., \ (f(t) = 0 \ when \ t < 0) & \text{input or stimulus to the system} \end{cases}$$

For such a system, there are several ways to classify the system response, one of which is:

$$y(t) = y_{zi}(t) + y_{zs}(t)$$

$y_{zi}(t)$ is the zero-input response of the system, and $y_{zs}(t)$ is the zero-state response of the system. To find $y_{zi}(t)$ and $y_{zs}(t)$, we first need to split the differential equation of the system to characterize the two parts separately, and then find the corresponding solution for each part.
As we just said, $y_{zi}(t)$ means there's no input to the system, and only the initial state of the system acts on the system. So we get the differential equation for $y_{zi}(t)$ like:

$$\begin{cases} \sum_{i=0}^{N} a_i y^{(i)}(t) = 0 \\ y^{(k)}(0_-) = ..., \ (k = 0, \cdots, N-1) \end{cases}$$

Note:

1. The right side of the differential equation equals 0, since there's no input at this moment.
2. We use $y^{(k)}(0_-)$ as the initial condition.

$y_{zs}(t)$ indicates that the initial state is zero, and only the input acts on the system. So we get the differential equation for $y_{zs}(t)$ like:

$$\begin{cases} \sum_{i=0}^{N} a_i y^{(i)}(t) = \sum_{j=0}^{M} b_j f^{(j)}(t) \\ y^{(k)}(0_-) = 0 \xrightarrow[characteristics]{system} y^{(k)}(0_+) = ? \quad (k = 0, \cdots, N-1) \\ f(t) = ..., \ (f(t) = 0 \ when \ t < 0) \end{cases}$$

Note:

1. The right side of the differential equation should be the expression of the stimulus f(t). Because now the stimulus signal has already been input to the system.
2. At this moment $y^{(k)}(0_-) = 0$, since the initial state is zero. But when we solve the differential equation, we should use $y^{(k)}(0_+)$ as the initial condition. Actually $y^{(k)}(0_+)$ comes from $y^{(k)}(0_-)$ by integrating both sides of the differential equation. And in most cases, $y^{(k)}(0_+)$ is equal to $y^{(k)}(0_-)$ unless there is $\delta(t)$ or its derivative on the right side of the differential equation.

About 0- and 0+

We define the instant at which the stimulus signal is input to the system as moment 0. And use 0_- to represent the moment just before the stimulus is added to the system, 0_+ to represent the moment just after the stimulus is added to the system.

So for zero-input response, since the stimulus signal has not been input to the system, we use $y^{(k)}(0_-)$ as the initial condition. While for zero-state response, since the stimulus signal has already been input to the system, we use $y^{(k)}(0_+)$ as the initial condition. What should be noted is that the zero-state response refers $y^{(k)}(0_-) = 0$, not $y^{(k)}(0_+) = 0$.

To get $y^{(k)}(0_+)$ from $y^{(k)}(0_-)$, do integration at both side of the differential equation from 0_- to 0_+. Refer to the following example.

Eg: $\begin{cases} y_f''(t) + 4y_f'(t) + 4y_f(t) = \delta(t) + 2e^{-t}u(t) \\ y_f'(0_-) = y_f(0_-) = 0 \end{cases}$, find out $y_f'(0_+)$ and $y_f(0_+)$.

Do integration at both side of the differential equation from 0_- to 0_+. For a continuous signal or a discontinuous signal with finite signal value, when integrating over an infinitesimal time interval, the result of the integration is zero. While the integral of $\delta(t)$ is one.

$$\underbrace{\int_{0_-}^{0_+} y_f''(t)dt}_{=y_f'(0_+)-y_f'(0_-)} + 4\underbrace{\int_{0_-}^{0_+} y_f'(t)dt}_{=y_f(0_+)-y_f(0_-)} + 4\underbrace{\int_{0_-}^{0_+} y(t)dt}_{=0} = \underbrace{\int_{0_-}^{0_+} \delta(t)dt}_{=1} + 2\underbrace{\int_{0_-}^{0_+} e^{-t}u(t)dt}_{=0}$$

$$[y_f'(0_+) - y_f'(0_-)] + 4[y_f(0_+) - y_f(0_-)] = 1$$

Where

$$y_f(0_+) - y_f(0_-) = 0$$
$$y_f'(0_+) = y_f'(0_-) + 1 = 1$$

So we get

$$\begin{cases} y_f(0_+) = y_f(0_-) = 0 \\ y_f'(0_+) = y_f'(0_-) + 1 = 1 \end{cases}$$

Think about $\begin{cases} y_f''(t) + 4y_f'(t) + 4y_f(t) = 2e^{-t}u(t) \\ y_f'(0_-) = y_f(0_-) = 0 \end{cases}$, what will we get?

Solve Differential Equation with MATLAB

MATLAB symbolic toolbox provides function **dsolve** to solve differential equation with constant coefficients. When using the symbolic method, $\delta(t)$ is represented by **dirac(t)** and $u(t)$ is represented by **heaviside(t)**.

When using **dsolve**:

1. The format of **dsolve** is like $\text{dsolve}([\text{eqn1}, \text{eqn2}, \ldots], [\text{cond1}, \text{cond2}, \ldots])$. Both eqn and cond are strings. Eqn refers to the differential equations. Cond refers to the initial conditions.
2. For versions earlier than MATLAB 2019, when **heaviside** is contained in the right side of the differential equation, take a number a bit greater than 0 to represent the 0+ in the condition. For versions later than MATLAB 2019, when the derivative of **heaviside** is contained in the right side of the differential equation, take a number a bit greater than 0 to represent the 0+ in the condition. This is because when used as a symbolic function, **heaviside** has no exact value at t=0, as we discussed in Lab 1. And **dsolve** doesn't handle this properly yet. This also leads to a slight error in the results, which we will discuss later.
3. Use function **simplify** to simplify the uncertainty model (optional).

Zero-Input Response

To find out the zero-input response, solve the differential equation which describes the system with function **dsolve**.

Eg:$y''(t) + 4y(t) = 0, y(0_-) = 1, y'(0_-) = 1$, find the zero_input response.

1. Represent differentiation by using the **diff** function.
2. Specify a differential equation by using ==.

```
syms y(t)
D2y = diff(y,t,2);
Dy = diff(y,t);
eqn = D2y+4*y==0;
conds = [y(0)==1, Dy(0)==1];
ysol = dsolve(eqn,conds);
yzi = simplify(ysol);
```

Zero-State Response

Since the systems discussed in this course are all LTI systems, we can find out the zero-state response by the following three ways.

1. Solve the differential equation with function **dsolve** as we did for zero-input response.
2. Find zero-state response based on the system properties. Function **lsim** helps to achieve it.
3. By convoluting the system impulse response with the system stimulus.

Method 2 and 3 are similar. Both of them will take advantage of the system's properties. And method 3 is the bridge between time domain analysis and s domain analysis which we will discuss in Lab 6.

Find out Zero-State Response by Solving Differential Equation (Symbolic Method)

By solving the differential equation, we can also find the zero-state response of the system. However, we should take 0+ instead of 0- as the initial condition as we discussed before.

Eg:

$y''(t) - 5y'(t) + 6y(t) = f(t), f(t) = e^{-2t}u(t), y(0_+) = 0, y'(0_+) = 0$, find the zero_state response.

```
syms y(t)
D2y = diff(y,t,2);
Dy = diff(y,t);
eqn = D2y-5*Dy+6*y==exp(-2*t)*heaviside(t);
conds = [y(0)==0, Dy(0)==0];
ysol = dsolve(eqn, conds);
yzs = simplify(ysol);
```

$$\frac{e^{-2t}\,(e^{t}-1)^{2}\,(\mathrm{sign}(t)+1)\,(3\,e^{2t}+4\,e^{3t}+2\,e^{t}+1)}{40}$$

Try to solve the differential equation by yourself, you should get a result as the following:

$$y_{zsr}(t) = -\frac{1}{20}[-e^{-2t} + 5e^{2t} - 4e^{3t}]u(t)$$

Which is the same as $\frac{e^{-2t}\,(e^{t}-1)^{2}\,(\mathrm{sign}(t)+1)\,(3\,e^{2t}+4\,e^{3t}+2\,e^{t}+1)}{40}$. In the result of function **dsolve**, "sing(t)+1" has the same effect as u(t).

Tips: Function **dsolve** can have several equations. When the right side of the differential equation also has n-order derivative, set the input signal as the second equation.

For your reference:

When we take 0.001 as 0+ we will get:

```
syms y(t)
D2y = diff(y,t,2);
Dy = diff(y,t);
eqn = D2y-5*Dy+6*y==exp(-2*t)*heaviside(t);
conds = [y(0.01)==0, Dy(0.01)==0];
ysol = dsolve(eqn, conds);
yzs = simplify(ysol)
```

$$\frac{e^{-2t}\left(4\,e^{5t}e^{-\frac{1}{20}}-5\,e^{4t}e^{-\frac{1}{25}}+1\right)}{20}$$

the result is somewhat different from the following one:

$$y_{zsr}(\mathrm{t}) = -\frac{1}{20}[-e^{-2t} + 5e^{2t} - 4e^{3t}]u(t)$$

This is because of the bias when y(0.01) and Dy(0.01) are used instead of y(0) and Dy(0) as the initial condition. Take $e^{-\frac{1}{20}}$ and $e^{-\frac{1}{25}}$ as 1, then yzs will be the same as the one calculated manually. Try to change 0.01 to 0.0001 to observe the difference in the result.

Use Function lsim to Find out the Zero-State Response (Numerical Method)

In MATLAB, **lsim** can also be used to find out the zero-state response in numerical method. It has the form like:

$$\mathrm{y} = \mathrm{lsim}(\mathrm{sys}, \mathrm{f}, \mathrm{t})$$

t is the time axis. **f** is the input signal. While **sys** represents the model of the LTI system.
sys can be generated by function **tf**:

$$\mathrm{sys} = \mathrm{tf}(\mathrm{b}, \mathrm{a})$$

b refers to the numerator part of the transfer function and **a** refers to the denominator part of the transfer function.
For the differential equation:

$$a_3y'''(t) + a_2y''(t) + a_1y'(t) + a_0y(t) = b_3f'''(t) + b_2f''(t) + b_1f'(t) + b_0f(t)$$

the value of **a** and **b** is:

$$\mathrm{a} = [a_3, a_2, a_1, a_0]$$
$$\mathrm{b} = [b_3, b_2, b_1, b_0]$$

and the transfer function is:

$$\mathrm{sys} = \mathrm{tf}(\mathrm{b}, \mathrm{a})$$

Note if the Nth derivative is missing in the differential equation, the corresponding element in the vector should be set to zero.
Example: $y''(t) - 5y'(t) + 6y(t) = f(t), f(t) = e^{-2t}u(t)$, try find out the zero-state response with function **lsim**().

```
sys = tf([1],[1,-5,6]);
t = 0:0.01:5;
f = exp(-2*t).*heaviside(t);
y = lsim(sys,f,t);
plot(t,y);
xlabel('t');ylabel('y(t)');title('Zero-State Response'),grid on;
```

Subs

Function **subs** is used to replace an old symbolic variable or string in the symbolic expression with a new symbolic or numeric variable or expression. The format of subs is like **subs**(S, old, new). It is useful when calculating $y = y_{zi} + y_{zs}$, where y_{zi} is calculated with symbolic method and y_{zs} with numerical method.
After using function **subs**, you will get a vector or a matrix with symbolic elements. Use function **double** to change the symbolic elements to the numeric ones.

Find out Zero-State Response by Convolution

Unit Impulse Response and Unit Step Response

When we add a unit impulse signal into a system with zero initial state, we get the unit impulse response, and the same goes for the unit step response. Since the impulse response and the step response can reflect the characteristics of the LTI system, they are of great significance to analyze the LTI system.

MATLAB provides function **impulse** and **step** to achieve the unit impulse response and the unit step response for the LTI system. The format of the two functions is shown as below:

$$h = \text{impluse}(sys, t) \text{ or } [h, t] = \text{impluse}(sys)$$
$$g = \text{step}(sys, t) \text{ or } [g, t] = \text{step}(sys)$$

Eg1: $y''(t) + 3y'(t) + 2y(t) = f(t)$, try to find the unit impulse response and the unit step response of the system.

```
t = 0:0.001:4;
sys = tf([1],[1,3,2]);
h = impulse(sys,t);
g = step(sys,t);
subplot(2,1,1);plot(t,h); axis([0,4,0,0.3]);
grid on;xlabel('t');ylabel('h(t)');title('Impulse Response');
subplot(2,1,2);plot(t,g); axis([0,4,0,0.6]);
grid on;xlabel('t');ylabel('g(t)');title('Step Response');
```

or we can do it this way:

```
sys = tf([1],[1,3,2]);
[h,t1] = impulse(sys);
[g,t2] = step(sys);
subplot(2,1,1);plot(t1,h); axis([0,4,0,0.3]);
grid on;xlabel('t');ylabel('h(t)');title('Impulse Response');
subplot(2,1,2);plot(t2,g); axis([0,4,0,0.6]);
grid on;xlabel('t');ylabel('g(t)');title('Step Response');
```

Convolution Operation

Convolution is extensively used in signal processing, communication and control engineering. It can be viewed as a description of the input-output relationship for an LTI system. In addition, it can be considered as an operation performed between two arbitrary signals. Convolution can be used as a tool for analysis the properties of signals or the processes that produced those signals.

The unit impulse response of a system is the response of the system to the unit impulse signal and it has special significance when a system is an LTI system, because it can fully characterize the system. The impulse response provides all of the information needed to determine the output of an LTI system with any input signal.

For a continuous-time system, the output $y(t)$ of a continuous-time LTI system is related to its input $x(t)$ and the system impulse response $h(t)$ through the convolution integral expressed as:

$$y(t) = \int_{-\infty}^{+\infty} x(\tau)h(t-\tau)d\tau = x(t) * h(t)$$

When using a computer to perform the convolution operation of a continuous time signal as above, we need first convert it to a discrete time signal. One way to approximate the continuous signals is to use piecewise constant functions by defining

$$\delta_{\Delta}(t) = \begin{cases} 1 & -\frac{\Delta}{2} \le t \le \frac{\Delta}{2} \\ 0 & otherwise \end{cases}$$

The continuous signal $x(t)$ can then be approximated by a piecewise constant signal $x_{\Delta}(t)$ as a sequence of pulses every Δ seconds with height $x(k\Delta)$, which is :

$$x_{\Delta}(t) = \sum_{k=-\infty}^{\infty} x(k\Delta)\delta_{\Delta}(t-k\Delta)$$

When $\Delta \to 0$, $x_{\Delta}(t) \to x(t)$. Similarly, $h(t)$ can be approximated by

$$h_{\Delta}(t) = \sum_{k=-\infty}^{\infty} h(k\Delta)\delta_{\Delta}(t-k\Delta)$$

The convolution integral can thus be approximated by convolving the two piecewise constant signals as follows:

$$y_{\Delta}(t) = \int_{-\infty}^{\infty} x_{\Delta}(\tau)h_{\Delta}(t-\tau)d\tau$$

Notice that $y_{\Delta}(t)$ is not necessarily a piecewise constant function. For computer representation purposes, the output needs to be generated in a discrete manner. This is achieved by further approximating the convolution integral as indicated below:

$$y_{\Delta}(n\Delta) = \Delta \sum_{k=-\infty}^{\infty} x(k\Delta)h((n-k)\Delta)$$

The discrete convolution sum $\sum_{k=-\infty}^{\infty} x(k\Delta)h((n-k)\Delta)$ can be computed with the MATLAB function **conv**. Then this sum can be multiplied by Δ to get an estimate of the continuous signal $y(t)$ at $t = n\Delta$. Note that as Δ is made smaller to obtain a closer approximation to $y(t)$.

For an LTI system, the output signal $y(n)$ is given by the convolution of the input signal $x(n)$ with the system impulse response $h(n)$.

$$y(n) = \sum_{m=-\infty}^{\infty} x(m)h(n-m) = x(n) * h(n)$$

Function **conv(u, v)** is used to calculate the convolution of vectors u and v. We can also use convolution to find out the zero-state response.

Eg:$f1(t) = u(t) - u(t-2), f2(t) = e^{-3t}u(t)$, find out the convolution of *f1* and *f2*.

(Since *f1* is infinite, while it is impossible for a computer to process infinite signal. So the scale of t should ensure that f2 can be attenuated to be small enough. Here we take t=2.5.)

```
dt = 0.01; t = 0:dt:2.5;
f1 = heaviside(t)-heaviside(t-2);
f2 = exp(-3*t).*heaviside(t);
f = conv(f1,f2)*dt;
n = length(f); tt = (0:n-1)*dt;
subplot(3,1,1); plot(t,f1); grid on;
title('f1(t)'); xlabel('t');ylabel('f1(t)');
subplot(3,1,2); plot(t,f2); grid on;
title('f2(t)'); xlabel('t');ylabel('f2(t)');
subplot(3,1,3); plot(tt,f); grid on;
title('f(t)=f1(t)*f2(t)');xlabel('t');ylabel('f1(t)');
```