332:347 – Linear Systems Lab – Fall 2016

Lab 1 – S. J. Orfanidis

Please review the lab requirements mentioned in the syllabus, and in particular, note the following.

a. The labs have a double-period duration and students are required to attend the *full* double-period sessions while working on the lab assignments. Partial attendance, or arriving late, or leaving early, will result in no credit for the lab. You may leave the lab room early only if you have completed all the lab questions and you have uploaded your final written report to Sakai.

b. Switching of lab sections is not allowed without *prior* approval from Prof. Orfanidis. Cell phones or other electronic devices are not allowed during the lab sessions.

c. Please remember to save your work regularly during the session.

d. Following each lab, students must submit a written *report* within 48 hours of *their lab session*, describing the purposes of the experiments and the methods used, and include all graphs and Matlab code. Probably, the best way to do this is to create an HTML or DOC or PDF file (HTML is best) using MATLAB's publish command from your M-file, and include in it comments about the purposes and methods. Then, you may upload that to Sakai Assignments together with any auxiliary function M-files that you may have written.

   The reports must be uploaded to Sakai under Assignments within the allotted 48-hr period. The written report will count *only if* the student had attended fully their scheduled double-period lab session.

   Please make sure your report's title contains your name and student number.

Problems

In this lab, in addition to the theoretical and symbolic toolbox questions, you must generate four graphs for problem-1 and six graphs for problem-2. Also, please review in advance the week-2 class notes on state-space realizations for first-order systems.

1. This question illustrates the definition of the Dirac delta function, $\delta(t)$, as a limit of ordinary functions. Consider the following four limiting forms, where $u(t)$ is the unit-step,

$$\delta(t) = \lim_{\epsilon\to 0} P_\epsilon(t)\,, \quad P_\epsilon(t) = \frac{1}{\epsilon}\Big[u(t+\tfrac{1}{2}\epsilon) - u(t-\tfrac{1}{2}\epsilon)\Big]$$

$$\delta(t) = \lim_{\epsilon\to 0} \frac{1}{\sqrt{2\pi\epsilon}}\, e^{-t^2/2\epsilon}$$

$$\delta(t) = \lim_{\epsilon\to 0} \frac{1}{\pi}\,\frac{\epsilon}{\epsilon^2 + t^2}$$

$$\delta(t) = \lim_{\epsilon\to 0} \frac{\sin(t/\epsilon)}{\pi t}$$

   where $P_\epsilon(t)$ is a rectangular pulse of height $1/\epsilon$ and support over the interval, $-\frac{1}{2}\epsilon \le t \le \frac{1}{2}\epsilon$.

   For each form, select two successively smaller values of ϵ and evaluate the above functions over 1000 equally-spaced time instants in the range $-1 \le t \le 1$, for example,

```
t = linspace(-1,1,1001);
```

Plot the corresponding functions versus time and note how they resemble the ideal $\delta(t)$ as ϵ gets smaller.

Notes: Vectorize your calculations. Use MATLAB's built-in function `sinc` to evaluate the fourth case (it handles the case $t = 0$). A simple way to implement the unit-step is by the anonymous function,

```
u = @(t) (t>=0);
```

One way to verify the correctness of the above limiting forms is to compute the Fourier transforms of the ordinary functions on the right-hand sides and show that each tends to unity as a function of frequency ω in the limit $\epsilon \to 0$. This is so because the Fourier transform of $\delta(t)$ is $\Delta(\omega) = 1$. We will discuss this approach later, but for completeness, we give the corresponding Fourier transforms below:

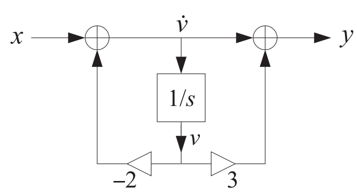
$$P_\epsilon(t) \xrightarrow{FT} F(\omega) = \frac{\sin(\omega\epsilon/2)}{\omega\epsilon/2}, \qquad \frac{1}{\sqrt{2\pi\epsilon}}\, e^{-t^2/2\epsilon} \xrightarrow{FT} F(\omega) = e^{-\epsilon\omega^2/2}$$

$$\frac{1}{\pi}\,\frac{\epsilon}{\epsilon^2 + t^2} \xrightarrow{FT} F(\omega) = e^{-\epsilon|\omega|}, \qquad \frac{\sin(t/\epsilon)}{\pi t} \xrightarrow{FT} = \begin{cases} 1, & |\omega| < 1/\epsilon \\ 0.5, & |\omega| = 1/\epsilon \\ 0, & |\omega| > 1/\epsilon \end{cases}$$

2. Consider the following linear system, driven by the causal input $x(t) = e^{-t}u(t)$, and subject to the initial condition, $y(0^-) = -1$,

$$\dot{y}(t) + 2y(t) = 3\dot{x}(t) + 2x(t) \tag{1}$$

Based on the discussion in week-2 class notes, this system may be replaced by the following equivalent state-space model, shown in block-diagram form below,



$$\boxed{\begin{aligned} \dot{v}(t) &= -2v(t) + x(t) \\ y(t) &= -4v(t) + 3x(t) \end{aligned}} \tag{2}$$

where $v(t)$ is the state and the block $1/s$ represents an integrator.

(a) Verify that the system of equations (2) implies Eq. (1).

(b) Using MATLAB's symbolic toolbox, construct the transfer function $H(s)$ of the system of Eq. (1) and invert it using the **ilaplace** function to obtain an analytical expression for the impulse response $h(t)$.

(c) How is the initial value of the state, $v(0^-) = v(0^+)$, related to the initial value, $y(0^-)$?

(d) For the given input and given $y(0^-)$, use the symbolic toolbox, to obtain the Laplace transform $Y(s)$ of the solution of Eq. (1). Then, expand it in partial fractions, and then invert it to obtain an analytical expression for the solution, $y(t)$.

(e) Determine the exact solution $y(t)$ using the alternative method based on the **dsolve** function.

(f) Repeat parts (d,e) for the state equation in (2) and obtain (i) the Laplace transform $V(s)$ of the state, (ii) its partial-fraction expansion, and (iii) an analytical expression for $v(t)$. Moreover, verify that your solutions for $y(t)$ and $v(t)$ satisfy the second equation in (2), that is, $y(t) = -4v(t) + 3x(t)$.

(g) Plot on the same graph the exact signals $y(t)$ and $v(t)$ over the interval $0 \le t \le 5$.

(h) Solve Eq. (1) numerically for $y(t)$ using the **lsim** function, and also solve Eq. (2) for $v(t)$, over the interval, $0 \le t \le 5$. Plot $y(t)$ and $v(t)$ in the same graph and verify that they agree with the exact solutions of part (g).

As discussed in week-2 class notes, because of the non-zero initial conditions, the transfer functions must be passed into **lsim** as state-space objects (class ss), that is, the usage should be,

```
y = lsim(S,x,t,v0);        % compute y(t), use v0 as initial state
v = lsim(Sv,x,t,v0);       % compute v(t)
```

where `S,Sv` are state-space objects constructed by the commands,

```
num = [b0,b1]; den = [1,a];    % numerator and denominator coeffs of H(s)
[A,B,C,D] = tf2ss(num,den);    % convert to state-space form
S = ss(A,B,C,D);               % class(S) is ss, and can be passed to lsim
```

and similarly for S_v, but now using, `num` = 1.

(i) Next, consider a discrete-time implementation. For the time-step value $T = 0.1$, calculate the zero-order-hold difference equation coefficients a_1, b_0, b_1 from Eq. (10) of the Appendix below. The differential equation (1) is then replaced by the discrete-time difference equation of Eq. (5), and implemented on a sample-by-sample basis by Eq. (7).

Define the following vector of time samples spaced at intervals of T and spanning the time range $0 \le t \le 5$,

```
tn = 0:T:5;
```

Then, iterate the discrete-time difference equation (7), while saving the computed time samples y_n. In addition, iterate the corresponding difference equation for the state, saving the time samples v_n. On the same graph, plot y_n and v_n versus $t_n = nT$. On the graph, place also the exact signals $y(t), v(t)$ determined in part (g).

Note: To be able to visually discern the differences between exact and computed values, the time-step T was intentionaly chosen not to be too small. If you repeat the calculation with $T = 0.01$, the graphs will be virtually indistinguishable.

(j) Repeat parts (c)–(i) when the initial condition is chosen to be $y(0^-)) = 0$.

Appendix: Discretization Schemes for First-Order Systems

For a first-order system, we consider the following differential equation and analog transfer function with given coefficients $\{a, B_0, B_1\}$ and input $x(t)$,

$$\frac{dy(t)}{dt} + a y(t) = B_0 \frac{dx(t)}{dt} + B_1 x(t) \quad \Rightarrow \quad H(s) = \frac{B_0 s + B_1}{s + a} \tag{3}$$

All discretization schemes convert the differential equation into a difference equation, i.e.,

$$y_n + a_1 y_{n-1} = b_0 x_n + b_1 x_{n-1} \tag{4}$$

where $y_n \approx y(nT)$ represents a numerical approximation to the true value of the output $y(t)$ at the discrete time, $t_n = nT$, $n = 0, 1, 2, \dots$, where T a chosen small discretization time interval, and $x_n = x(nT)$ is input signal sampled at time $t = t_n$. The coefficients $\{a_1, b_0, b_1\}$ depend on the value of T and the chosen discretization scheme. The difference equation is iterated by writing it in the following form,

$$\boxed{\begin{aligned} &\text{for } \; n = 0, 1, 2, \dots, \\ &\qquad y_n = -a_1 y_{n-1} + b_0 x_n + b_1 x_{n-1} \end{aligned}} \tag{5}$$

where the starting value y_{-1} may be chosen to be the same as the given initial value $y(0^-)$ of the differential equation, that is,

$$y_{-1} \approx y(0^-) \tag{6}$$

The input $x(t)$ will be assumed to be causal so that, $x_{-1} = 0$. The implementation of the difference equation on a sample-by-sample basis, i.e., without using arrays to hold the input and output signals, but rather computing each output sample y_n as each input sample x_n becomes available, can be formulated with the help of the delayed signals $w(n), v(n)$,

$$w(n) = y(n-1)\,, \quad v(n) = x(n-1)$$

which are updated to time $n+1$ as follows,

$$w(n+1) = y(n)\,, \quad v(n+1) = x(n)$$

The difference quation (5) subject to the initial conditions (6) can be recast in the following repetitive form, in which w, v are temporary variables (internal states) updated at each iteration,

$$\boxed{\begin{aligned} &\text{initialize:} \quad w = y_{-1}\,, \quad v = 0 \\ &\text{for} \;\; n = 0, 1, 2, \dots, \\ &\qquad y_n = -a_1 w + b_0 x_n + b_1 v \\ &\qquad w = y_n \\ &\qquad v = x_n \end{aligned}} \tag{7}$$

Forward-Euler, Backward-Euler, and Trapezoidal Rules

The coefficients $\{a_1, b_0, b_1\}$ were derived in class,

$$\boxed{b_0 = \frac{B_1 p + B_0}{ap+1}\,, \quad b_1 = \frac{B_1 q - B_0}{ap+1}\,, \quad a_1 = \frac{aq-1}{ap+1}} \tag{8}$$

where p, q are defined as follows in the three cases, in terms of the discretization time step T,

$$\begin{aligned} \text{forward Euler:} \quad & p = 0\,, \qquad q = T \\ \text{backward Euler:} \quad & p = T\,, \qquad q = 0 \\ \text{trapezoidal/bilinear:} \quad & p = T/2\,, \quad q = T/2 \end{aligned} \tag{9}$$

Zero-Order Hold

In this case, we saw in class that the coefficients $\{a_1, b_0, b_1\}$ are,

$$\boxed{b_0 = B_0\,, \quad b_1 = B_1\left(\frac{1-e^{-aT}}{a}\right) - B_0\,, \quad a_1 = -e^{-aT}} \tag{10}$$

Typical Outputs