Lab 2: Analysis of discrete-time systems described by LCCDEs

PURPOSE: To learn how to use Matlab to find the response, $y[n]$, $n \geq 0$, of an N^{th} order LTI discrete-time system for a given input signal, $x[n]$, $n \geq 0$, and a given set of initial conditions: $y[-1]$, $y[-2]$, ... , $y[-N]$. Systems of interest are those which can be described by a Linear Constant Coefficient Difference Equation (LCCDE), *e.g.*:

$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k]$$

The decomposition of the response $y[n]$ into a sum of zero-state and zero-input components will be demonstrated and the use of the Matlab function **filter** to find the zero-state and zero-input reponses will emphasize the meaning of these responses. Matlab will also be used to find values of the system's impulse response and step response. The fact that the zero-state response can be found as the convolution of the system's input and impulse response will also be demonstrated via use of the Matlab function **conv**. The procedure for finding a closed-form expression for the impulse response is also reviewed and the use of Matlab in finding a closed-form expression for the impulse response is demonstrated.

BACKGROUND: See class notes regarding analysis of LCCDEs.

PRELAB EXERCISES: Let $y_{zs}[n] = h[n] * x[n]$ where $h[n]$ and $x[n]$ are infinite-length ***causal*** sequences. And let $z[n] = h_t[n] * x_t[n]$ where $h_t[n]$ and $x_t[n]$ are truncated versions of $h[n]$ and $x[n]$, as defined below:

$$h_t[n] = \begin{cases} h[n] = 0, & n < 0 \\ h[n], & 0 \leq n \leq n_{\max} \\ 0, & n > n_{\max} \end{cases} \quad \text{and} \quad x_t[n] = \begin{cases} x[n] = 0, & n < 0 \\ x[n], & 0 \leq n \leq n_{\max} \\ 0, & n > n_{\max} \end{cases} \tag{1}$$

Where applicable, your answers to the questions below should be expressed in terms of $n_{\max}$. You should assume that $n_{\max}$ has the same value for both truncated sequences.

1. For which values of n, might we find that $y_{zs}[n]$ is nonzero?
2. For which values of n, might we find that $z[n]$ is nonzero?
3. For which values of n, are we sure to find that $y_{zs}[n] = z[n]$?

Hint for the last question above: Define the functions $h_b[n]$ and $x_b[n]$ as follows:

$$h_b[n] = \begin{cases} 0, & n \leq n_{\max} \\ h[n], & n > n_{\max} \end{cases} \quad \text{and} \quad x_b[n] = \begin{cases} 0, & n \leq n_{\max} \\ x[n], & n > n_{\max} \end{cases}$$

and note that since $h[n]$ and $x[n]$ are causal, we have that:

$$h[n] = h_t[n] + h_b[n] \quad \text{and} \quad x[n] = x_t[n] + x_b[n]$$

Hence:

$$\begin{aligned} y_{zs}[n] &= h[n] * x[n] = \big(h_t[n] + h_b[n]\big) * \big(x_t[n] + x_b[n]\big) \\ &= \underbrace{\big(h_t[n] * x_t[n]\big)}_{z[n]} + \underbrace{\big(h_b[n] * x_t[n]\big)}_{r[n]} + \underbrace{\big(h_t[n] * x_b[n]\big)}_{s[n]} + \underbrace{\big(h_b[n] * x_b[n]\big)}_{v[n]} \end{aligned}$$

Thus, $y_{zs}[n]$ will be equal to $z[n]$ for those values of n such that $r[n] = s[n] = v[n] = 0$. For which values of n will we find that $r[n] = s[n] = v[n] = 0$?

PROCEDURE/EXERCISES:

1. For use throughout the lab, create two .m files: one named **dt_delta.m** and the other named **dt_step.m**. The file named **dt_delta.m** will implement the Kronecker Delta function, $\delta[n]$, defined for integer-valued arguments as follows:

$$\delta[n] = \begin{cases} 1, n = 0 \\ 0, n \neq 0 \end{cases}$$

It should include the following lines:

```
function y = dt_delta(n)
y = double(n == 0);
i_nonint = find(round(n) ~= n); % finds the indices of any noninteger elements of n
y(i_nonint) = NaN; % NaN stands for 'not a number'; it's the matlab way of saying
                   % that the value is undefined
```

The file named **dt_step.m** will be a matlab function which implements the discrete-time unit step function, $u[n]$, defined for integer-valued arguments as follows:

$$u[n] = \begin{cases} 1, n \geq 0 \\ 0, n < 0 \end{cases}$$

It should include the following lines:

```
function y = dt_step(n)
y = double(n >= 0);
i_nonint = find(round(n) ~= n); % finds the indices of any noninteger elements of n
y(i_nonint) = NaN; % NaN stands for 'not a number'; it's the matlab way of saying
                   % that the value is undefined
```

2. Consider the causal LTI system whose input-output relationship is characterized by the following LCCDE:

$$y[n] - 3y[n-1] - 4y[n-2] = x[n] + 2x[n-1] \tag{2}$$

On the class handout entitled *Analysis of Causal Recursive LCCDE's*, a closed-form expression for the response, $y[n]$, $n \geq 0$, of System (2) when: $x[n] = 4^n u[n]$, $y[-1] = 1$, and $y[-2] = -1$, was found to be:

$$y[n] = \frac{6}{5}n(4)^n + \frac{26}{25}(4)^n - \frac{26}{25}(-1)^n, \quad n \geq 0 \tag{3}$$

The total response $y[n]$, $n \geq 0$ was found as the sum of the zero-input response, $y_{zi}[n]$, and the zero-state response, $y_{zs}[n]$, where:

$$y_{zs}[n] = \frac{6}{5}n(4)^n + \frac{26}{25}(4)^n - \frac{1}{25}(-1)^n \tag{4}$$

and:

$$y_{zi}[n] = -1(-1)^n = (-1)^{n+1} \tag{5}$$

As discussed in class, System (2) is not stable since it has characteristic roots with magnitude greater than or equal to 1. Since both the specified input, $x[n] = 4^n u[n]$, and the resulting system response, $y[n]$, grow quickly with increasing $n \geq 0$, we will limit our simulation of this system to fairly small values of n.

a) Execute the Matlab commands below (making sure that you understand what each command does) to find the values of $y[n]$, for $n = 0, 1, 2, 3$ given that: $y[-1] = 1$, $y[-2] = -1$, and $x[n] = 4^n u[n]$, as stated above.

```
% define the system
a = [1  -3  -4]; % coeffs from LHS of difference equation
b = [1  2]; % coeffs from RHS of difference equation
% define the array of indices for which we are interested in observing the output
nmax = 3;
n = 0:nmax;
% define the input signal of interest
x = 4.^n  .*  dt_step(n);
% translate the stated initial conditions into corresponding initial state values for the
% transposed direct-form II structure used by Matlab
ics = [1 , -1];
istates = filtic(b,a,ics);
% use the matlab filter function to find the response y[n]
y = filter(b,a,x,istates)
% for comparison purposes, calculate the associated values of the closed-form
%  expression found in the Handout; see equation (3) above.
ycf = (6/5)*n.*(4.^n) + (26/25)*(4.^n) - (26/25)*((-1).^n)
```

Compare the elements of the vector **y** to those of the vector **ycf** to verify that the values of $y[n]$, $n = 0, 1, 2, 3$, as found by the Matlab **filter** function are in agreement with those resulting from the closed-form expression of equation 3.

b) Use the Matlab function **filter** to find the zero-state component, $y_{zs}[n]$, $n = 0, 1, 2, 3,$ of the response, $y[n]$, to System (2) when: $y[-1] = 1$, $y[-2] = -1$, and $x[n] = 4^n u[n]$. Then compare the values you find using Matlab to the values found from the closed-form expression of equation (4).

Hint: Recall that the zero-state component of the response is the response of the system to input $x[n]$, $n \geq 0$ when the initial states of the system are set equal to zero. Thus with the Matlab variables **b,a,n,x,** and **istates** as defined by the code in part (a), the values of $y_{zs}[n]$, $n = 0, 1, 2, 3$ can be easily found as follows:

```
% use the matlab filter function to find the response yzs[n], n = 0, 1, 2, 3
yzs = filter(b,a,x, zeros(size(istates)))
% calculate the associated values of the closed-form expression for yzs[n]; see eqn (4).
yzscf = (6/5)*n.*(4.^n) + (26/25)*(4.^n) - (1/25)*((-1).^n)
```

c) Use the Matlab **filter** function to find the zero-input component, $y_{zi}[n]$, $n = 0, 1, 2, 3,$ of the response when: $y[-1] = 1$, $y[-2] = -1$, and $x[n] = 4^n u[n]$, as stated above. Then compare the values you find using Matlab to the values resulting from the closed-form expression for $y_{zi}[n]$ in equation (5).

Hint: Recall that the zero-input component is the response of the system when the initial state values are as determined by the IC's $y[-1]$ and $y[-2]$ and the input is set to zero. Thus with the Matlab variables **b,a,n,x,** and **istates** as defined by the code in part (a), the values of $y_{zi}[n]$, $n = 0, 1, 2, 3$ can be found as follows:

```
% use the matlab filter function to find the response yzi[n], n = 0, 1, 2, 3
yzi = filter(b,a,zeros(size(x)), istates)
```

% calculate the associated values of the closed-form expression for $y_{zi}[n]$; see eqn (5).
```
yzicf = (-1)*((-1).^n)
```

d) Use the Matlab **filter** function to find the impulse response, $h[n]$, $n = 0, 1, 2, 3$, of the system described by the LCCDE of equation 2.

Recall: the impulse response is a special name given to the zero-state response of the system when $x[n] = \delta[n]$. Thus with the variables **b, a, n,** and **istates** as defined by the code in part (a), the values of $h[n]$, $n = 0, 1, 2, 3$ can be found as follows:

% use the matlab filter function to find the impulse response h[n], $n = 0, 1, 2, 3$
```
h = filter(b,a,dt_delta(n), zeros(size(istates)))
```

Verify that the values of the impulse response found in Matlab agree with the closed-form expression found in class (see eq. 6 below):

$$h[n] = \left(\frac{6}{5}(4)^n - \frac{1}{5}(-1)^n\right)u[n] \tag{6}$$

e) Verify that $y_{zs}[n] = h[n] * x[n]$ where $*$ denotes the convolution operator.

Hint: The Matlab function, **conv**, performs the convolution of two finite-length sequences as described below.

If the vector **x** contains ℓ_x elements with values $x[n_1^{(x)}]$ ... $x[n_2^{(x)}]$
and the vector **h** contains ℓ_h elements with values $h[n_1^{(h)}]$ ... $h[n_2^{(h)}]$,
then the vector **z** created by the matlab command **z= conv(h,x)**
will contain $\ell_z = \ell_x + \ell_h - 1$ elements whose values are: $z[n_1^{(z)}]$ ... $z[n_2^{(z)}]$
where: $n_1^{(z)} = n_1^{(x)} + n_1^{(h)}$, $n_2^{(z)} = n_2^{(x)} + n_2^{(h)}$, and $z[n] = h_t[n] * x_t[n]$ where:

$$h_t[n] = \begin{cases} 0, & n < n_1^{(h)} \\ h[n], & n_1^{(h)} \le n \le n_2^{(h)} \\ 0, & n > n_2^{(h)} \end{cases} \quad \text{and} \quad x_t[n] = \begin{cases} 0, & n < n_1^{(x)} \\ x[n], & n_1^{(x)} \le n \le n_2^{(x)} \\ 0, & n > n_2^{(x)} \end{cases} \tag{7}$$

Thus, with the vector **x** as defined in part (a) and the vector **h** as defined in part (d), a vector **yzsconv** containing the values $y_{zs}[n]$, $n = 0, 1, 2, 3$, can be found as follows.

```
z = conv(h,x)
yzsconv = z(1:length(n))
```

Confirm that the vector **yzsconv** as created by the Matlab commands above is identical to the vector **yzs** found in part (b), thus verifying that $y_{zs}[n] = h[n] * x[n]$, $n = 0, 1, 2, 3$. Note the length of the vector **z**. How does it compare to the length of the vectors **h** and **x**? We only assigned the first 4 values of the vector **z** to the **yzsconv** array. Since we found that the first four elements of the vector **z** had values $y_{zs}[n]$, $n = 0, 1, 2, 3$, can we assume that the the fifth element of the vector **z** has a value equal to $y_{zs}[4]$? Verify your answer by comparing the value of the fifth element of the vector **z** to the value resulting from equation (4) when $n = 4$. Make use of the results to your prelab exercises to explain your findings.

3. Consider the filter described by the following LCCDE:

$$y[n] - 1.8\cos(\pi/4)y[n-1] + 0.81y[n-2] = x[n] + \frac{1}{2}x[n-1] \tag{8}$$

a) Follow the steps below to find a closed-form expression for the impulse response, $h[n]$, of System (8).

i. Find the characteristic roots, λ_1 and λ_2, of System (8). Are the roots distinct? Find their magnitudes? Is System (8) stable?

Hint: The roots, $\lambda_1, ..., \lambda_N$, of the Nth order polynomial:

$$a_0\lambda^N + a_1\lambda^{N-1} + ... + a_N = 0 \tag{9}$$

can be found using Matlab as follows:

```
a = [a_0, a_1, ... , a_N]
lambda = roots(a) % lambda is a column vector (N×1)
```

Note that the vector **lambda** as created by the Matlab commands above will contain the N roots, $\lambda_1, ..., \lambda_N$. Furthermore, the Matlab function **abs** returns the magnitude of its argument. Thus, the vector returned by **abs(lambda)** will contain the magnitudes of the N roots.

ii. From class, we know that the impulse response to a second-order recursive system with $N > M$ and distinct roots λ_1 and λ_2 has the form:

$$h[n] = \left(C_1\lambda_1^n + C_2\lambda_2^n\right)u[n] \tag{10}$$

Use Matlab to find the values of C_1 and C_2.

Hint: Note that the free parameters C_1 and C_2 of equation (10) can be found if the values of the impulse response, $h[n]$, are known for two values of $n \geq 0$. For example, the constraints of equation (10) for $n = 0$ and $n = 1$ can be expressed in matrix form as:

$$\underbrace{\begin{bmatrix} h[0] \\ h[1] \end{bmatrix}}_{h} = \underbrace{\begin{bmatrix} \lambda_1^0 & \lambda_2^0 \\ \lambda_1^1 & \lambda_2^1 \end{bmatrix}}_{P}\underbrace{\begin{bmatrix} C_1 \\ C_2 \end{bmatrix}}_{C} \quad \Rightarrow \quad C = P^{-1}h \tag{11}$$

Thus with **lambda** as created above, the vector C and hence the values of C_1 and C_2 can be easily found in Matlab as follows:

```
N=2, n = 0:(N-1); % n is a row vector (1×N)
b = [1 ,  0.5]
a = [1, -1.8*cos(pi/4) , 0.81]
h = transpose(filter(b,a,dt_delta(n))) % h is N×1
P = (ones(size(lambda))*transpose(lambda)) ...
        .^((transpose(n))*ones(size(n)))
C = P\h % this implements P^-1 h
```

iii. Use equation (10) together with the values of C_1 and C_2, found in part (ii.), to find and **stem** the values of $h[n]$, $-10 \leq n \leq 100$. Include in your lab report the Matlab commands you used to do this. ***Hint***: you should make use of your **dt_step** function. [note: when making your plot, use the subplot(3,1,1) command so that you

can later add two other plots to this figure.] Add an appropriate title such as 'impulse response based on closed-form expression'.

iv. Check your closed-form expression for $h[n]$ by using the Matlab function **`filter`** to generate the values of the impulse response, $h[n]$, $-10 \le n \le 100$, for System (8). Compare these values for $h[n]$, $-10 \le n \le 100$, to those found in part (iii.). Include in your lab report the Matlab commands you used. Explain your technique for comparing the values found in this part to the values found in part (iii.).

v. The impulse response of a stable system is always transient in nature (it becomes negligible after a finite number of samples). Based on your plot of $h[n]$, does $h[n]$ appear to be transient? Would you say that $h[n]$ is negligible after 100 samples?

b) The step response of a system refers to the zero-state response of the system when the input to the system is the unit step sequence, $u[n]$. Use the matlab function **`filter`** to generate and plot the step response, $y_{\text{step}}[n]$, of the system for $-10 \le n \le 100$. Superimpose the input, $u[n]$, on the same plot with the output, $y_{\text{step}}[n]$.

i. Based on your plot, to what value, K, does $y_{\text{step}}[n]$ converge as n gets large?

ii. Note that when $x[n] = u[n]$, the forced or particular solution has the form $y_p[n] = Ku[n]$. Use the approach presented in class to find the value of K such that $Ku[n]$ is the particular solution for the difference equation (8) when $x[n] = u[n]$.

iii. The DC gain of a system refers to the value, K, to which the step response converges as n gets large. In general, the DC gain of a system described by a LCCDE can be found as the ratio:

$$\frac{b_0 + b_1 + \ldots + b_M}{a_0 + a_1 + \ldots + a_N}$$

With vectors **b** and **a** defined as in part, verify that the following Matlab command yields the DC gain K.

```
sum(b)/sum(a)
```

c) Demonstration of the relationship between a system's impulse and step response.

Since $\delta[n] = u[n] - u[n-1]$, *i.e.*, since the unit impulse is the first difference of the unit step, we expect the impulse response of a LTI system to be the first difference of the step response, *i.e.*, we expect: $h[n] = y_{\text{step}}[n] - y_{\text{step}}[n-1]$. You can use the matlab function **`diff`** to verify this (see description of the **`diff`** function under the caution below). Apply the **`diff`** operator to the step response you obtained in part (b). Stem the result, being sure to use an appropriate index vector and a descriptive title such as 'first difference of ystep'. How does the current plot compare to the stem plot of the impulse response you made in part (a)?

Caution: The **`diff`** function reduces the size of a vector by one. For example:

if $\mathbf{u} = \left\{u[n_1], u[n_1+1], u[n_1+2], \ldots, u[n_2-1], u[n_2]\right\}$

and if **delta = diff(u)**, then

$\mathbf{delta} = \left[u[n_1+1] - u[n_1], u[n_1+2] - u[n_1+1], \ldots, u[n_2] - u[n_2-1]\right]$

i.e., $\mathbf{delta} = \left\{\delta[n_1+1], \delta[n_1+2], \ldots, \delta[n_2-1], \delta[n_2]\right\}$.

Type **`help diff`** to obtain more information on **`diff`**.