Digital Signal Processing Ⅱ

2^{nd} EXPERIMENT

Report

(2nd report of DSP2 course)

Subject	Digital Signal Processing Ⅱ
Professor	Je Hyeong Hong
Submission Date	September 14th, 2021
University	Hanyang University
School	College of Engineering
Department	Department of Computer Science & Engineering
Student ID	**Name**
2019009261	최가온(CHOI GA ON)

Exercises

In this part, there are several exercise questions. Each exercise consists of code and its result. All documents including Matlab code, result, and this report are uploaded in this website :

https://github.com/Gaon-Choi/ELE3077/tree/main/lab_experiment02

Exercise 1

a) Make the function 'impseq' and 'stepseq'.

(Matlab code) impseq.m

```
function [x, n] = impseq(n0, lb, ub)
% Generates x(n) = delta(n - n0); lb <= n <= ub
% -----------------------------------------------------
%
n = [lb:ub]; x = [(n-n0) == 0];
```

(Matlab code) stepseq.m

```
function [x, n] = stepseq(n0, lb, ub)
% Generates x(n) = u(n-n0); lb <= n <= ub
% -----------------------------------------------------
%
n = [lb:ub]; x = [(n-n0) >= 0];
```

b) Plot (stem) each of the following sequences over the indicated interval.

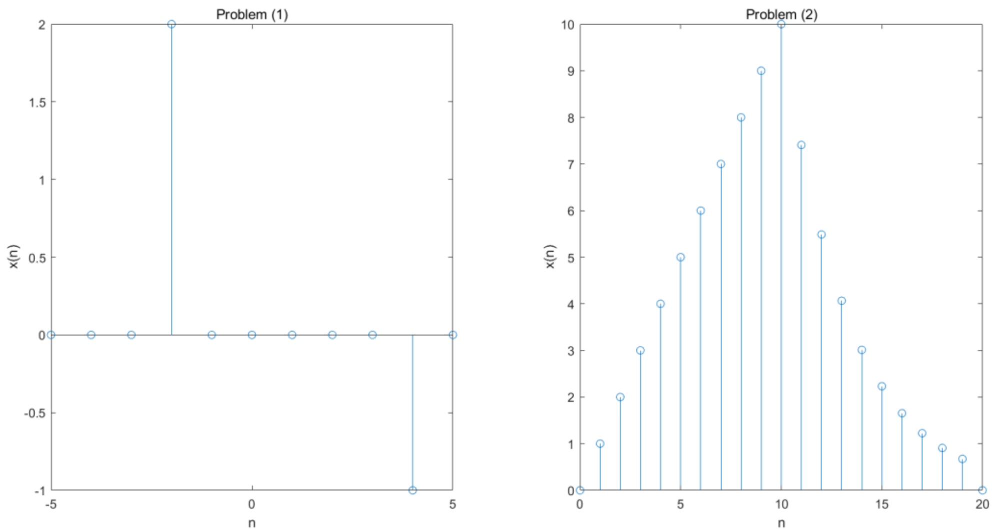
(1) $x(n) = 2\delta(n+2) - \delta(n-4), \quad -5 \leq n \leq 5$

(2) $x(n) = n[u(n) - u(n-10)] + 10e^{-0.3(n-10)}[u(n-10) - u(n-20)], \ 0 \leq n \leq 20$

(Matlab Code) lab2_exercise1_b.m

```
n1 = [-5:5];
x1 = 2 * impseq(-2, -5, 5) - impseq(4, -5, 5);

n2 = [0:20];
x2 = n2 .* (stepseq(0, 0, 20) - stepseq(10, 0, 20)) + 10 *
exp(-0.3 * (n2 - 10)) .* (stepseq(10, 0, 20) - stepseq(20, 0,
20));

subplot(1, 2, 1);
stem(n1, x1); title("Problem (1)");
xlabel("n"); ylabel("x(n)");

subplot(1, 2, 2);
stem(n2, x2); title("Problem (2)");
xlabel("n"); ylabel("x(n)");
```

(Result)



Exercise 2

a) Make the function 'conv_m'.

(Source code) conv_m.m

```
function [y, ny] = conv_m(x, nx, h, nh)
% Modified convolution routine for signal processing
% --------------------------------------------------
% [y, ny] = conv_m(x, nx, h, nh)
% [y, ny] = covolution result
% [x, nx] = first signal
% [h, nh] = second signal
%
nyb = nx(1) + nh(1);    nye = nx(length(x)) + nh(length(h));
ny = [nyb:nye]; y = conv(x, h);
```

b) Generate and plot below signals.

Explain graphically whether it is time-variant or time-invariant.

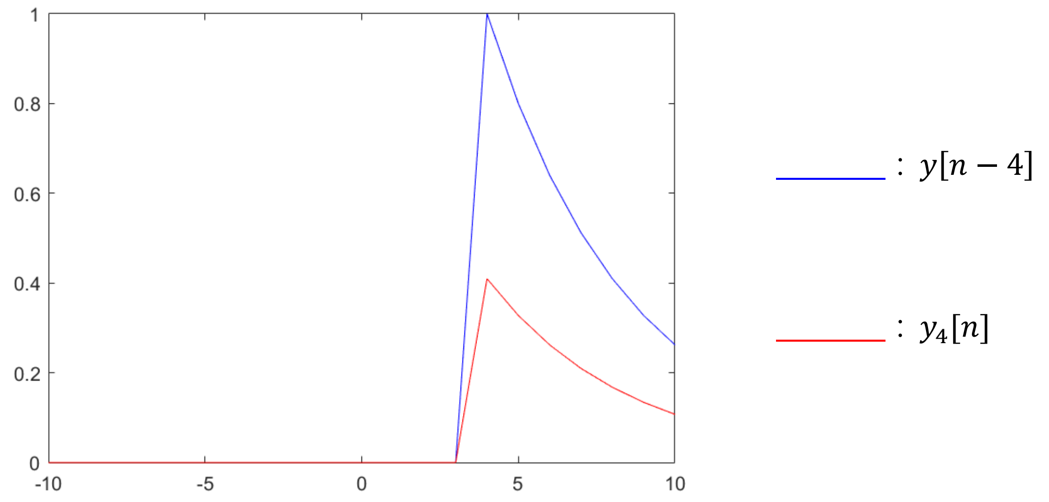
$$x[n] = u[n] \ , \qquad -10 \le n \le 10$$

- $y[n] = 0.8^n x[n]$
- $y[n-4] = 0.8^{n-4} x[n-4]$
- $y_4[n] = 0.8^n x[n-4]$

(Source Code) lab2_exercise2_b.m

```
n = -10:10;

y = 0.8 .^ n .* stepseq(0, -10, 10);
y_1 = 0.8 .^ (n - 4) .* stepseq(4, -10, 10);
y_2 = 0.8 .^ n .* stepseq(4, -10, 10);

grid on
plot(n, y_1, 'b-',n, y_2, 'r-')
xticks(-10:2:10);
xlabel("n");    ylabel("y");
```

(Result)



If $y[n] = 0.8^n x[n]$ is time-invariant, the following equation must be true.

$$y[n-4] = y_4[n]$$

It means that the plots of $y[n-4]$ and $y_4[n]$ must be identical.

The above MATLAB graphing result indicates that the two plots are not the same from each other, which implies that $y[n]$ is time-variant.

c) LTI system

(1) Show numerically the identity $h_1 * \delta[n] = h_1$ is correct.

Use impulse response with '$impseq(0, -5, 5)$'

- signal x

n	-5	-4	-3	-2	-1	0	1	2	3	4	5
x	0	0	4	13	6	-3	-1	2	4	0	0

- LTI system h_1

n	-1	0	1	2	3	4
$\boldsymbol{h_1}$	5	2	3	-5	1	11

- LTI system h_2

n	-1	0	1	2	3	4
h_2	-4	2	0	6	3	1

(Source Code) lab2_exercise2_c1.m

```
[imp, impn] = impseq(0, -5, 5);

h1 = [5, 2, 3, -5, 1, 11];
n1 = -1:4;

left = [-6:9 ; conv_m(h1, n1, imp, impn)]
right = [n1 ; h1]
```

(Result)

```
명령 창
>> lab2_exercise2_c1

left =

    -6    -5    -4    -3    -2    -1     0     1     2     3     4     5     6     7     8     9
     0     0     0     0     0     5     2     3    -5     1    11     0     0     0     0     0


right =

    -1     0     1     2     3     4
     5     2     3    -5     1    11
fx
```

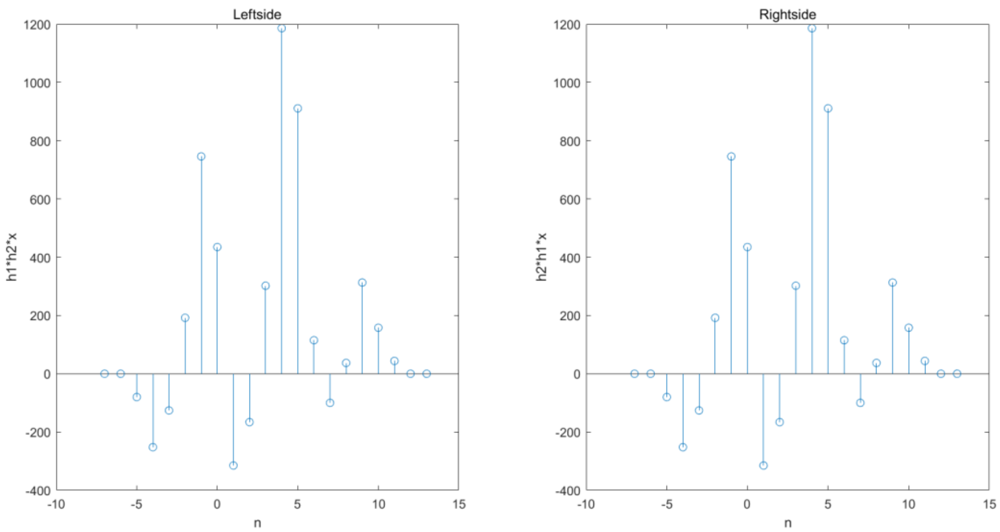
(2) By plotting both $h_1 * h_2 * x$ and $h_2 * h_1 * x$, show numerically they are the same. Write down the error between the two by subtracting one from another.

(Source Code) lab2_exercise2_c2.m

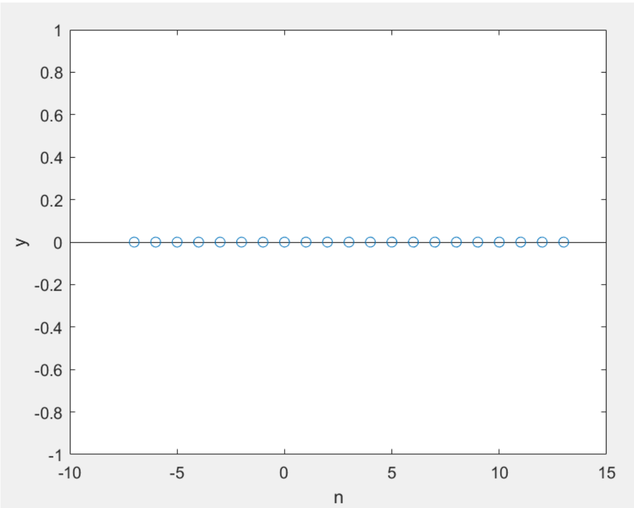
```
h1 = [5, 2, 3, -5, 1, 11];
h2 = [-4, 2, 0, 6, 3, 1];
n1 = -1:4; n2 = -1:4;

x = [0, 0, 4, 13, 6, -3, -1, 2, 4, 0, 0];
nx = -5:5;

[conv1, convn1] = conv_m(h1, n1, h2, n1);
[h1_h2_x, h1_h2_xn] = conv_m(conv1, convn1, x, nx);
[conv2, convn2] = conv_m(h2, n2, h1, n2);
[h2_h1_x, h2_h1_xn] = conv_m(conv2, convn2, x, nx);
```

```
subplot(1, 2, 1);
stem(h1_h2_xn, h1_h2_x);
title("Leftside");
xlabel("n");    ylabel("h1*h2*x");

subplot(1, 2, 2);
stem(h2_h1_xn , h2_h1_x);
title("Rightside");
xlabel("n");    ylabel("h2*h1*x");
```

(Result)




```
title("h1*h2*x - h2*h1*x")
stem(h1_h2_xn, h1_h2_x - h2_h1_x)
xlabel("n"); ylabel("y")
```



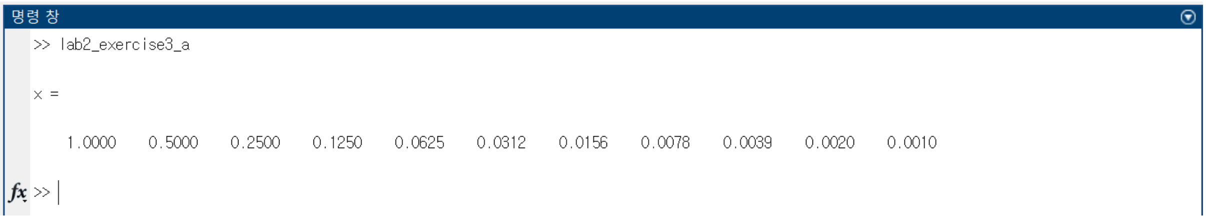
Exercise 3

a) Create a vector from $x[0]$ to $x[10]$ where $x[n] = 0.5^n u[n]$ by using vectorization.

(Source Code) lab2_exercise3_a.m

```
N = 10

n = 0:N;
x = 0.5 .^ n .* stepseq(0, 0, N)
```

(Result)



```
명령 창
>> lab2_exercise3_a

x =

    1.0000    0.5000    0.2500    0.1250    0.0625    0.0312    0.0156    0.0078    0.0039    0.0020    0.0010

fx >>
```

b) Write a MATLAB function for the accumulator which yields a vector of $\{y[n]\}$ where $y[n] = y[n-1] + x[n]$

(Hint: you can use a for loop or the cumsum function.)

(Source Code) lab2_exercise3_b.m

```
n = 0:10;
x = 0.5 .^ n .* stepseq(0, 0, 10);

y = zeros(1, 10);

for i = 1:10
    if i == 1
        y(i) = x(i);
    else
        y(i) = y(i - 1) + x(i);
    end
end

disp("y = ")
disp(y)
```

(Result)

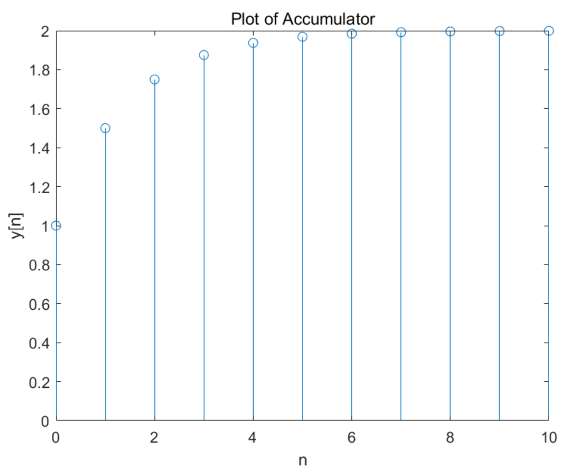
```
명령 창
>> lab2_exercise3_b
y =
    1.0000    1.5000    1.7500    1.8750    1.9375    1.9688    1.9844    1.9922    1.9961    1.9980

fx >> |
```

c) Plot $y[n]$ and compute the value of convergence where $n \to \infty$.

(Source Code) lab2_exercise3_c.m

```
N = 10

n = 0:N;
x = 0.5 .^ n .* stepseq(0, 0, N);

y = zeros(1, N);

for i = 1:N+1
    if i == 1
        y(i) = x(i);
    else
        y(i) = y(i - 1) + x(i);
    end
end

stem(n, y)
title("Plot of Accumulator");
xlabel("n"); ylabel("y[n]");
```

(Result)



From the given equation,

$$y[n] = y[n-1] + x[n]$$

it can be solved by using the solution of differential equation.

$$y[n] - y[n-1] = x[n] = 0.5^n u[n] \qquad \cdots (1)$$

i) Homogeneous solution

$$y_h[n] - y_h[n-1] = 0 \qquad \cdots (2)$$

We can set $y_h[n] = C_1 \cdot \lambda^n$

Then, we can get

$$y[n-1] = C_1 \cdot \lambda^{n-1} \qquad \cdots (3)$$

Substituting (3) into (2),

$$C_1 \cdot \lambda^n - C_1 \cdot \lambda^{n-1} = 0 \qquad \cdots (4)$$

Dividing both sides by $C \cdot \lambda^{n-1}$ from (4),

$$\lambda - 1 = 0 \qquad \cdots (5)$$

The equation (5) is called characteristic equation, and its root is $\lambda = 1$.

$$y_h[n] = C_1(1)^n = C_1 \qquad \cdots (6)$$

ii) Particular solution

$$y_p[n] - y_p[n-1] = (0.5)^n u[n] \qquad \cdots (7)$$

We can set $y_p[n]$ as

$$y_p[n] = C_2 \cdot (0.5)^n u[n] \qquad \cdots (8)$$

Substituting (8) into (7),

$$C_2 \cdot (0.5)^n u[n] - C_2 \cdot (0.5)^{n-1} u[n-1] = (0.5)^n u[n] \qquad \cdots (9)$$

When $n \geq 1$,

$$C_2 \cdot (0.5)^n - C_2 \cdot (0.5)^{n-1} = (0.5)^n$$

Dividing the equation above by $(0.5)^{n-1}$,

$$C_2(0.5 - 1) = 0.5$$

$$C_2 = -1$$

$$\therefore y_p[n] = C_2 \cdot (0.5)^n u[n] = -(0.5)^n u[n] \qquad \cdots (10)$$

The solution of (1) is the addition of homogeneous solution and particular solution.

$$y[n] = y_h[n] + y_p[n] \qquad \cdots (11)$$

Substituting (6) and (10) into (11),

$$y[n] = y_h[n] + y_p[n] = C_1 - (0.5)^n u[n] \qquad \cdots (12)$$

Use boundary condition: $y[0] = y[-1] + x[0] = 1$

$$C_1 - (0.5)^0 u[0] = C_1 - 1 = 1$$

$$C_1 = 2$$

$$\therefore y[n] = 2 - (0.5)^n u[n]$$

As n approaches to positive infinity,

$$\lim_{n\to\infty} y[n] = \lim_{n\to\infty} \{2 - (0.5)^n u[n]\} = 2 - 0 = 2$$

$$\therefore \lim_{n\to\infty} y[n] = 2$$