Digital Signal Processing Ⅱ

12^{th} EXPERIMENT

Report

(WEEK13 report of DSP2 course)

Subject	Digital Signal Processing Ⅱ
Professor	Je Hyeong Hong
Submission Date	November 27th, 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_experiment12

Exercise 1

exercise1-a)

Generate low-pass filter by using FDA tool referring to the specifications below.

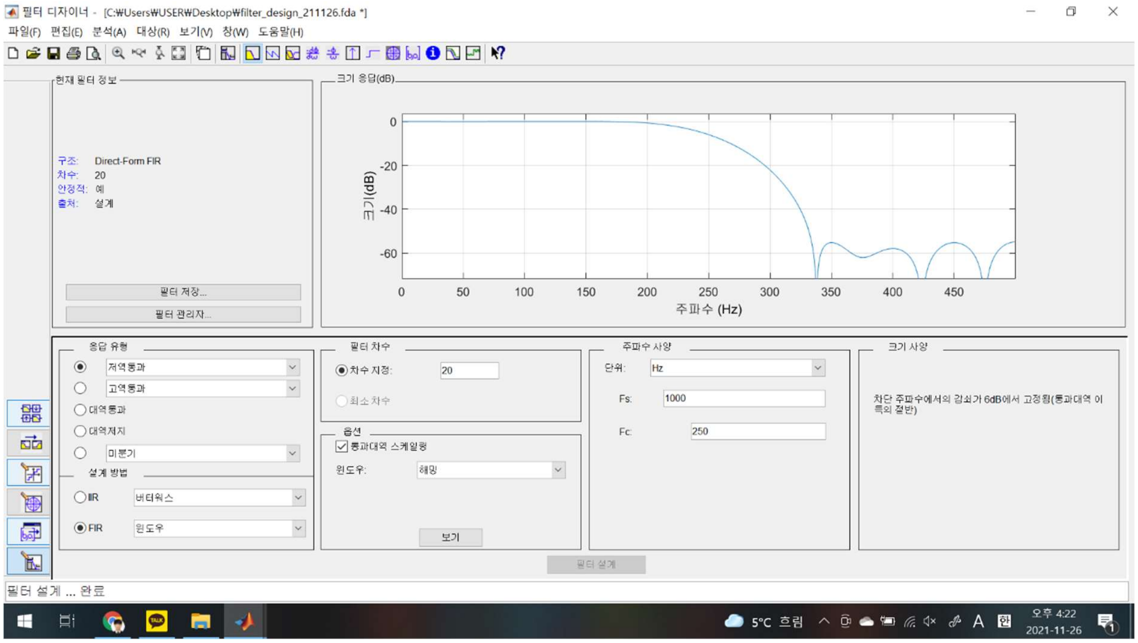
- FIR filter (Design method : Hamming window)
- num of order : 20
- sampling frequency : 1000Hz
- cutoff frequency : 250Hz

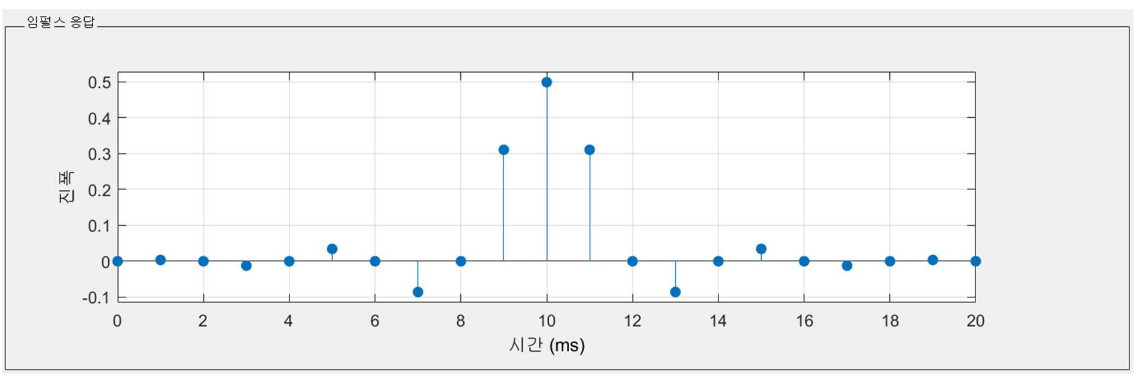
and save this filter's impulse response in the worksheet.

(MATLAB Code) lab13_exercise1_a.m

```
b = [2 2.76 2.622 2.6740, 1.8];
k = tf2latc(b);

delta = [1 0 0 0];

output_dir = filter(b, 1, delta)
output_lat = 2 * latcfilt(k, delta)
```

(Results)






```
명령 창
>> lab13_exercise1_a

output_dir =

    2.0000    2.7600    2.6220    2.6740


output_lat =

    2.0000    2.7600    2.6220    2.6740

fx >>
```

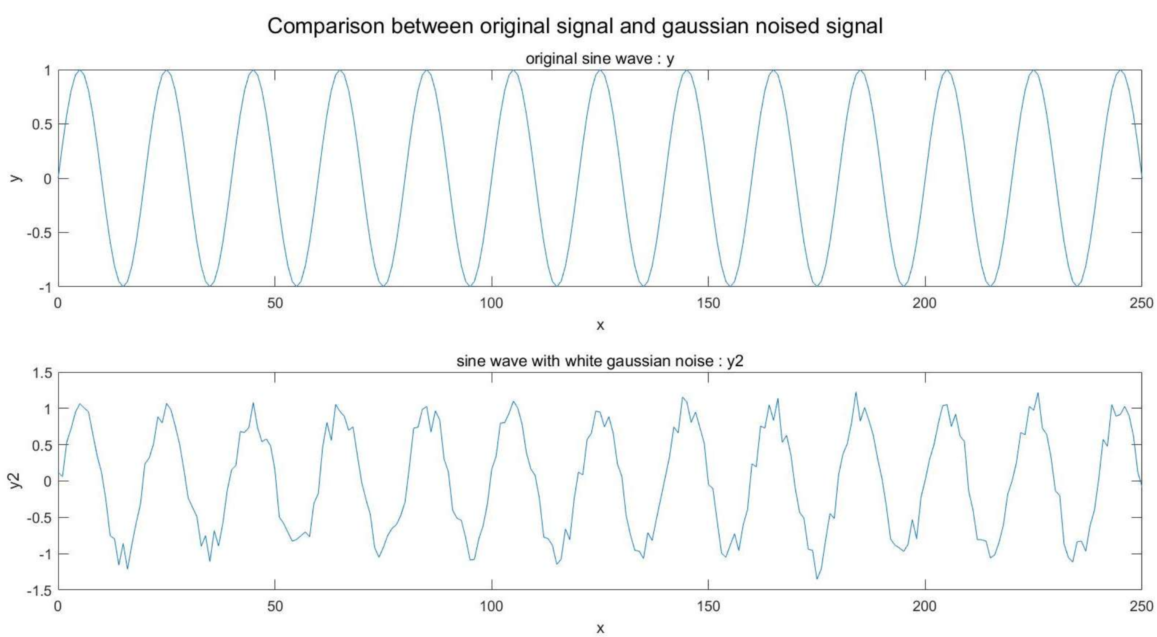
exercise1-b)

Generate a (1/20)Hz sine wave (x=0~250) and add white gaussian noise that has 15dB SNR.

(MATLAB Code) lab13_exercise1_b.m

```
x = 0:250;
y = sin(2 * pi * (1/20) * x);
y2 = awgn(y, 15, 'measured');
% noise = 0.4 * randn(1, length(x));
% y2 = y + noise

subplot(2, 1, 1)
plot(x, y); title("original sine wave : y");
xlabel("x"); ylabel("y");
subplot(2, 1, 2)
plot(x, y2); title("sine wave with white gaussian noise : y2");
xlabel("x"); ylabel("y2");
sgtitle("Comparison between original signal and gaussian noised signal")
```

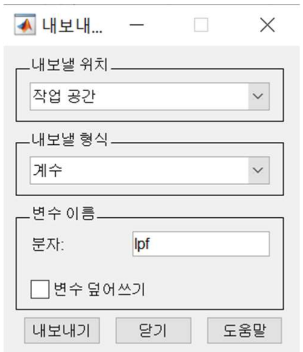
(Results)



exercise1-c)

Filter this signal with the filter that was generated in Q1-a, so that it has no delay.

Hint: you can use ‘filtfilt’ function



(MATLAB Code) lab13_exercise1_c.m

* Also refer to filter_design_211126.fda and load it, then variable “lpf” will appear.

```
x = 0:250;
y = sin(2 * pi * (1/20) * x);
y2 = awgn(y, 15, 'measured');

y_filtered = filter(lpf, 1, y2);
y_nodelay = filtfilt(lpf, 1, y2);

% filtfilt compensates delay caused by filtering
% the difference between filter and filtfilt is: time shift
```

(Results)

(SKIP)

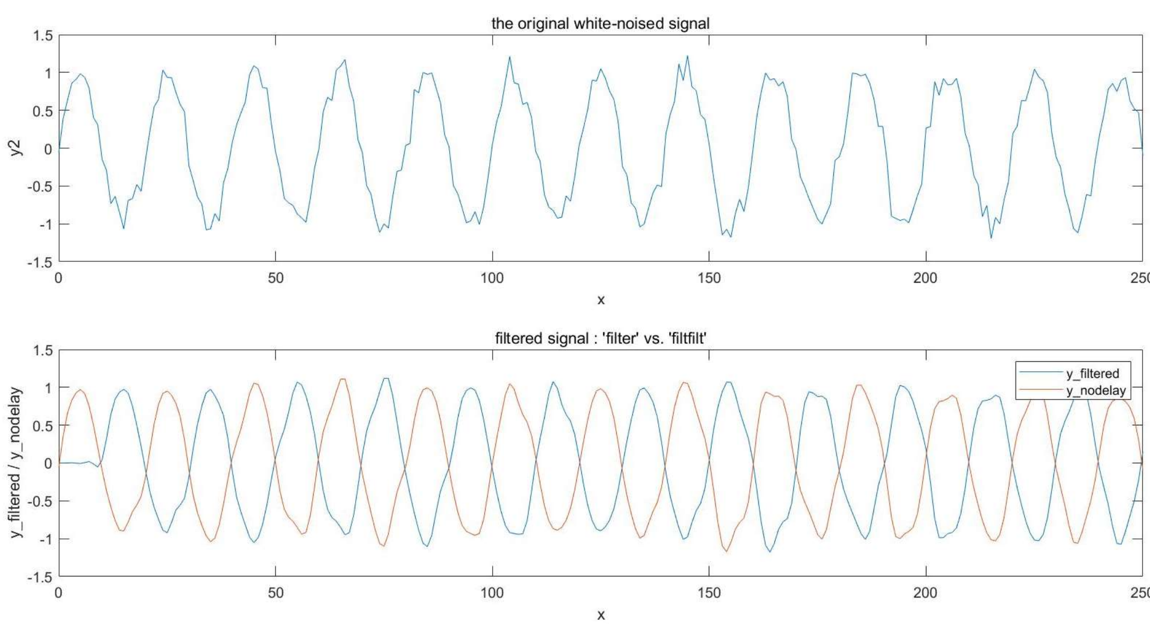
exercise1-d)

Plot the original signal with AWGN and the filtered signal and compare these.

(MATLAB Code) lab13_exercise1_d.m

```
x = 0:250;
y = sin(2 * pi * (1/20) * x);
y2 = awgn(y, 15, 'measured');

y_filtered = filter(lpf, 1, y2);
y_nodelay = filtfilt(lpf, 1, y2);

subplot(2, 1, 1);
plot(x, y2)
title("the original white-noised signal")
xlabel("x"); ylabel("y2");

subplot(2, 1, 2);
plot(x, y_filtered, x, y_nodelay);
title("filtered signal : 'filter' vs. 'filtfilt'")
xlabel("x"); ylabel("y\_filtered / y\_nodelay");
legend("y\_filtered", "y\_nodelay");
```

(Results)



Exercise 2

exercise2-a)

For data sampled at 1000Hz, there is a lowpass filter with no more than 3dB of ripple in a passband from 0 to 200Hz, and at least 60 dB of attenuation in the stopband. Find the filter order when the stop frequency is 300Hz, 350Hz and 400Hz respectively. Compare these and explain briefly.

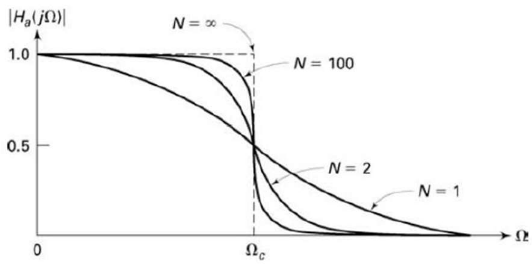
Hint: use 'buttord' function

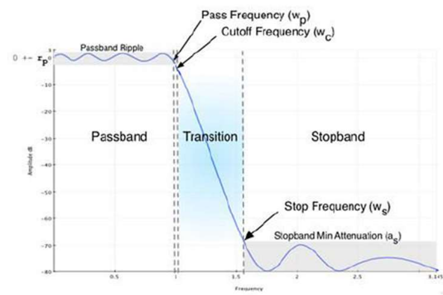
(MATLAB Code) lab13_exercise2_a.m

```
Fc = 200;            % cut-off frequency
Fs = 1000;           % sampling frequency
Wp = 200/(Fs/2);      % normalized cutoff frequency

Ws1 = 300/500;  % stop-frequency_1
Ws2 = 350/500;  % stop-frequency_2
Ws3 = 400/500;  % stop-frequency_3

% What will be order when stop band gets bigger?
% If stop band gets smaller...
   % - transition region will be steeper
   % - order will be increased
   % - getting closer to ideal filter.

% minimum order, about the given condition
[n1, Wn1] = buttord(Wp, Ws1, 3, 60);
[n2, Wn2] = buttord(Wp, Ws2, 3, 60);
[n3, Wn3] = buttord(Wp, Ws3, 3, 60);

disp("n1 = " + n1)
disp("n2 = " + n2)
disp("n3 = " + n3)
```

(Results)

As the stop frequency gets smaller(Ws1→Ws2→Ws3), the transition region will be steeper, and then the order will be increased. Consequently, if the stop band gets smaller, it will be more closer to the ideal filter.





(Results)

명령 창

```
>> lab13_exercise2
n1 = 11
n2 = 7
n3 = 5
fx >>
```

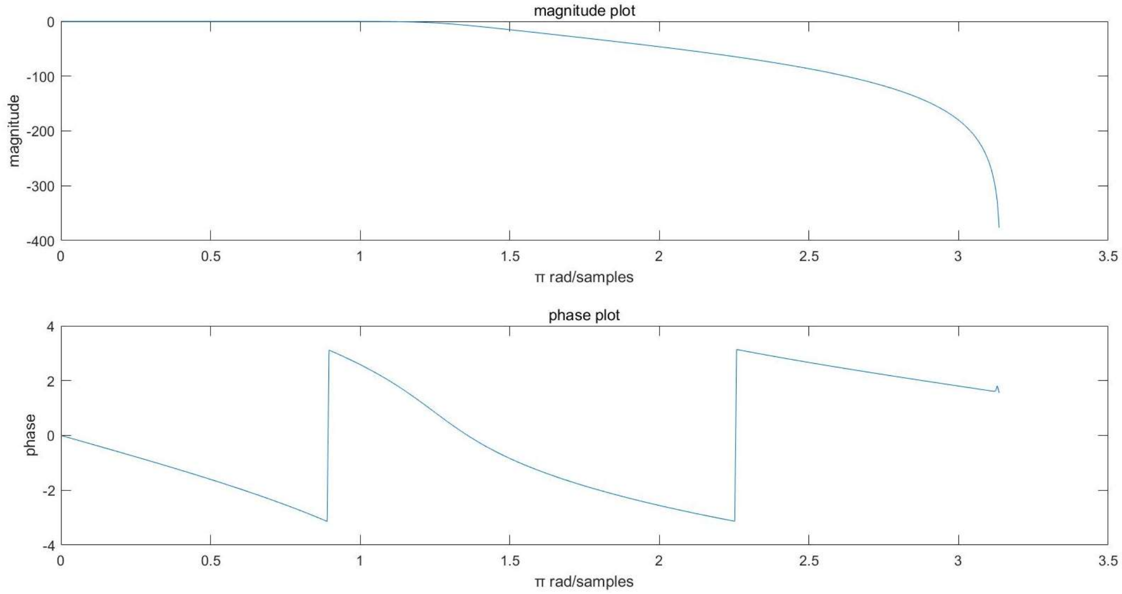
exercise2-b)

Design a 7th-order low-pass Butterworth filter with a cutoff frequency of 200Hz, which, for data sampled at 1000 Hz, corresponds to 0.4π rad/sample. Plot its magnitude (decibel) and phase responses.

(MATLAB Code) lab13_exercise2_b.m

```
Fc = 200;   % cut-off frequency
Fs = 1000;  % sampling frequency

[b, a] = butter(7, 200/500);
% freqz(b, a) % it may be another solution.

[h, w] = freqz(b, a);
magH = abs(h); angH = angle(h);
db = mag2db(magH);

subplot(2, 1, 1);
plot(w, db);
title("magnitude plot"); xlabel("¥ð rad/samples"),
ylabel("magnitude");
subplot(2, 1, 2);
plot(w, angH);
title("phase plot"); xlabel("¥ð rad/samples"),
ylabel("phase");
```

(Results)



exercise2-c)

Plot the pole-zero plot of Q2-b and explain briefly why all of poles is inside the unit circle.

(Explanation)

A system is BIBO(bounded input bounded output) stable.

→ $h[n]$ is absolutely summable.

→ It would have a Fourier transform.

→ Fourier transform is defined only on the unit circle. ($z = e^{jw}$)

→ The ROC extends outward (∵ causal system)

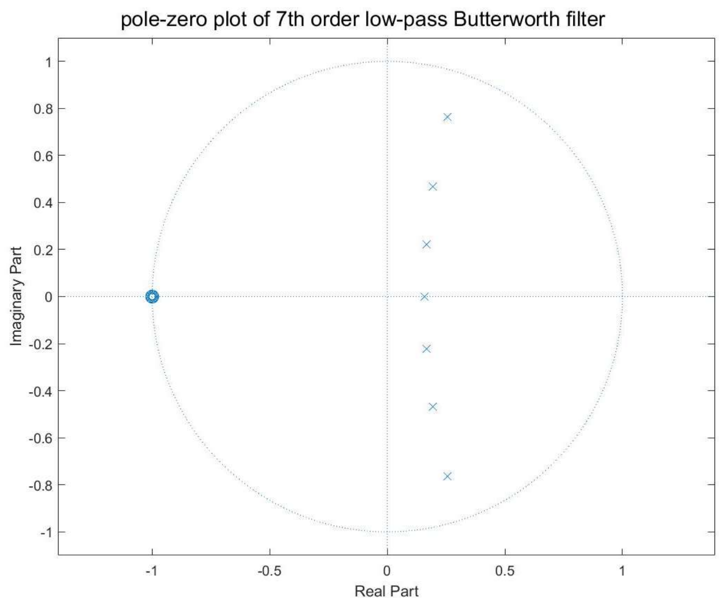
→ ROC does not contain any poles.

Consequently, all poles must be located inside the unit circle.

(MATLAB Code) lab13_exercise2_c.m

```
Fc = 200;   % cut-off frequency
Fs = 1000;  % sampling frequency

[b, a] = butter(7, 200/500);
% freqz(b, a) % it may be another solution.

[h, w] = freqz(b, a);
zplane(b, a)
xlabel("Real Part"); ylabel("Imaginary Part");
sgtitle("pole-zero plot of 7th order low-pass Butterworth filter")
```

(Results)



pole-zero plot of 7th order low-pass Butterworth filter
Imaginary Part
Real Part
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.5
0
0.5
1

exercise2-d)

Design a 6^{th}-order Butterworth band-stop filter with normalized edge frequencies of 0.2π and 0.6π rad/sample. Plot its magnitude (decibel) and phase responses.

(MATLAB Code) lab13_exercise2_d.m

```
[b, a] = butter(3, [0.2, 0.6], 'stop');
freqz(b, a)
sgtitle("pole-zero plot of 6th order Butterworth band-stop filter")
```

(Results)