REPORT

-결과 보고서-

제출일 : 2018/11/30
과목 : 임베디드 신호처리 시스템
교수님 : 김 수민 교수님
학과 : 임베디드 시스템
학번 : 2014146042
이름 : 주 성 민

제 3 절 실습

3.1 시스템 이산화

(1) 실습 다음은 cutoff 주파수가 1 rad/sec인 3차 아날로그 Butterworth LPF의 전달함수이다.

$$H(s) = \frac{1}{s^3 + 2s^2 + 2s + 1}$$

이 시스템에 bilinear transformation을 적용하여 디지털 필터로 변환하라. 이 때 다음과 같은 표본화 주파수를 사용하라. (bilinear 함수 사용)

$$f_s \in \{5,\ 10\}\ \text{Hz}$$

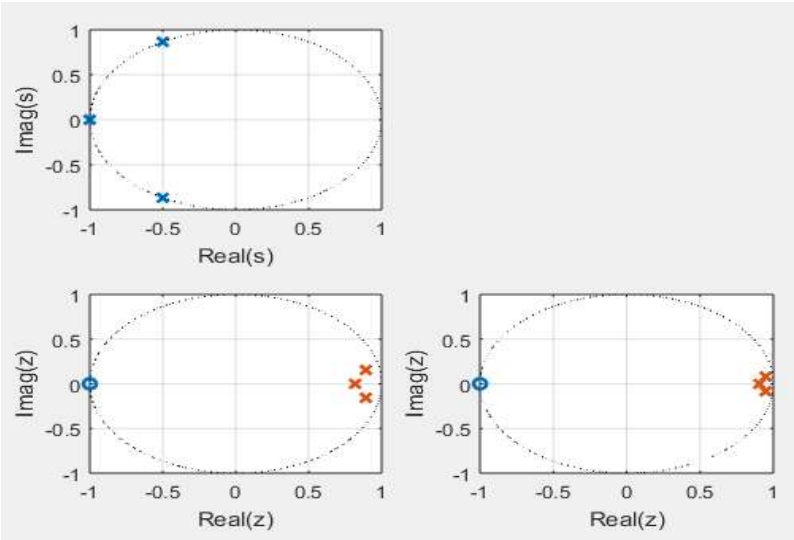
(2) 실습 DEMO (1)의 아날로그 필터와 디지털 필터의 pole-zero plot을 표시하라. 각 시스템의 안정성을 판단하라. (반지름 1인 원과 함께 표시하라.) (tf2zp 함수 사용) (그림 6 참고)

먼저 디지털 필터로 변환 시 사용된 소스코드 입니다.

```
clc;clear
theta = 0:0.01:2*pi;
[z, p, k] = buttap(3);
[num, den] = zp2tf(z,p,k);
[numd1, dend1]=bilinear(num, den, 5);%fs = 5
[zd1, pd1, k]=tf2zp(numd1, dend1);
[numd2, dend2]=bilinear(num, den, 10);%fs = 10
[zd2, pd2, k]=tf2zp(numd2, dend2);
```

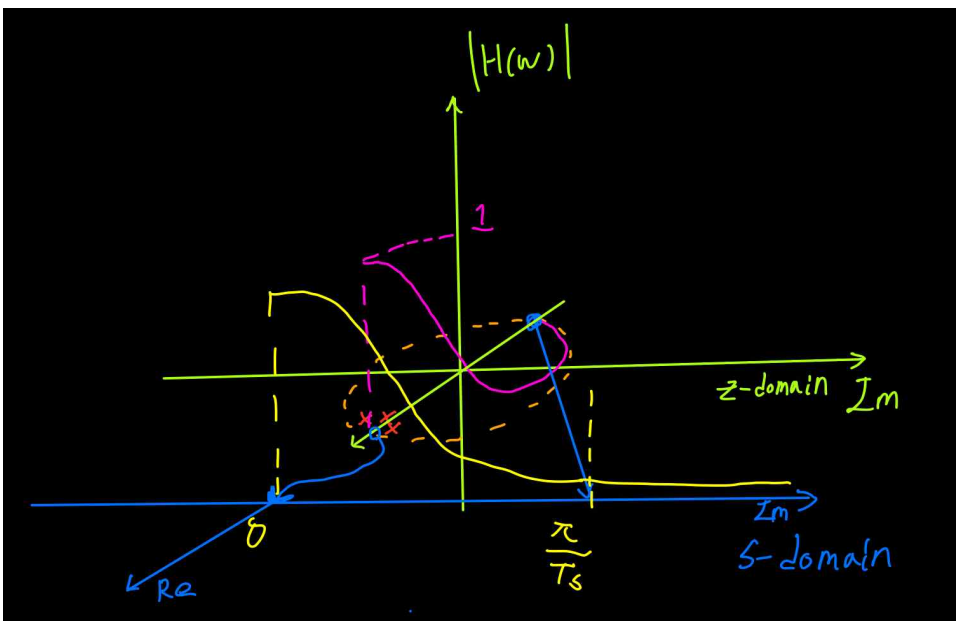
3차 Butterworth LPF를 간단하게 구현하고 bilinear함수의 입, 출력 규격을 맞춰주어 디지털 필터로 변환하였습니다. 1번 문제는 주어진 함수만으로 간단하게 해결할 수 있는 문제였습니다. 다음은 pole 과 zero를 그려주는 코드와 결과 그래프입니다.

```
figure(1)
subplot(2,2,1)
plot(real(z),imag(z),'o',real(p),imag(p),'x','LineWidth',2)
grid on; xlabel('Real(s)');ylabel('Imag(s)')
hold on;plot(cos(theta),sin(theta),':k');axis([-1,1,-1,1])

subplot(2,2,3)
plot(real(zd1),imag(zd1),'o',real(pd1),imag(pd1),'x','LineWidth',2)
grid on; xlabel('Real(z)');ylabel('Imag(z)')
hold on;plot(cos(theta),sin(theta),':k');axis([-1,1,-1,1])

subplot(2,2,4)
plot(real(zd2),imag(zd2),'o',real(pd2),imag(pd2),'x','LineWidth',2)
grid on; xlabel('Real(z)');ylabel('Imag(z)')
hold on;plot(cos(theta),sin(theta),':k');axis([-1,1,-1,1])
```



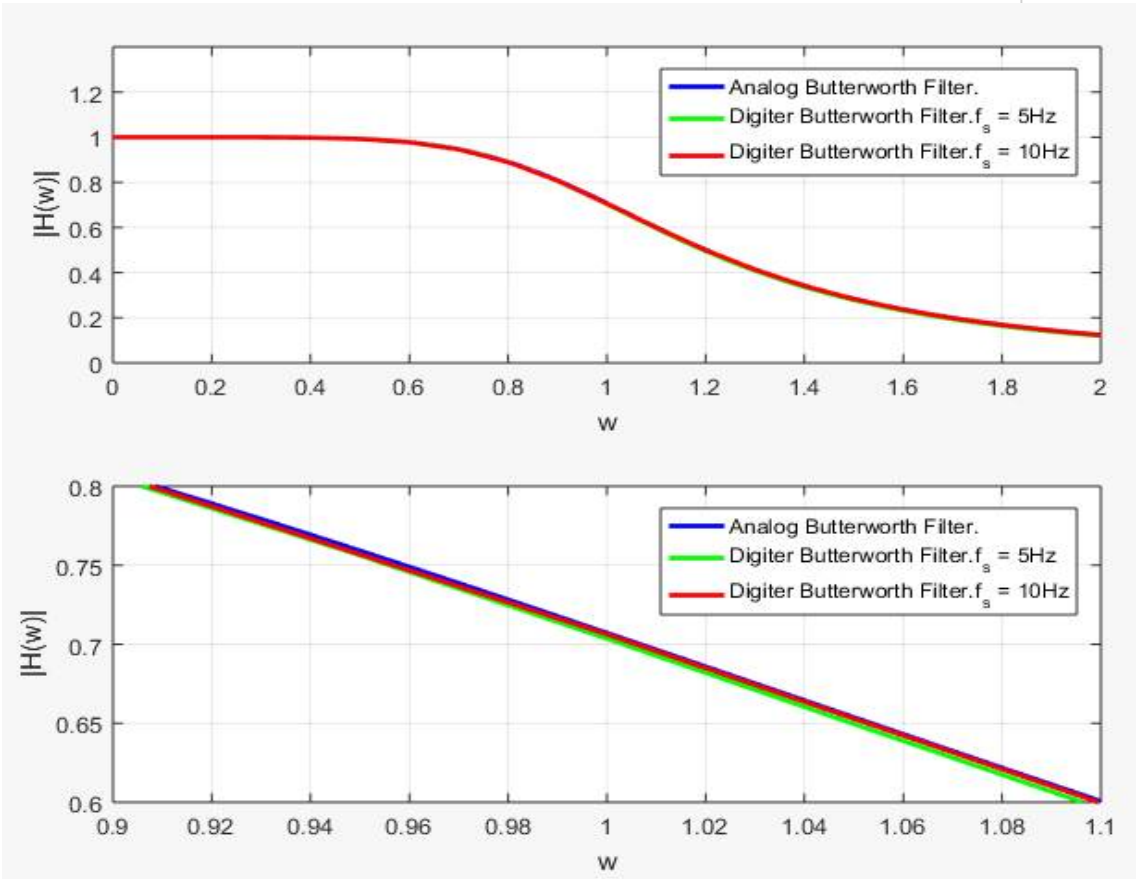
Analog Butterworth LPF의 극점은 좌방평면의 단위 원 위에 있고, 모든 극점이 좌반평면에 존재하기 때문에 안전한 시스템이라고 할 수 있습니다. Digital Butterworth LPF의 pole – zero plot은 아래쪽의 두 그래프이며, 좌측이 5Hz, 우측이 10Hz로 표본화 해준 결과입니다. 두 그래프 모두 극점이 단위원 내부에 존재하기 때문에 디지털 필터 또한 안정한 시스템이라고 판단할 수 있습니다.

아날로그 필터에서 디지털필터로 변환해주는 과정에서 샘플링 주파수는 아주 중요한 역할을 하게 된다고 배웠습니다. s-domain의 무한한 주파수 영역에서 w = $-\frac{\pi}{T_s} \sim \frac{\pi}{T_s}$ 구간의 좌방평면이 z-domain의 단위원에 매핑되는데, 샘플링 주파수가 클수록 아날로그 필터의 더 많은 부분을 가져올 수 있다는 뜻입니다. 실습 결과를 보면, 샘플링 주파수가 5Hz일 때 보다 10Hz일 때 극점들이 더 조밀한 것을 볼 수 있는데, z-domain으로 매핑되는 w 영역이 넓어져 상대적으로 극점의 간격들이 좁아지기 때문인 것 같습니다. (아래는 저희 조의 생각을 그림으로 표현해 보았습니다.)



(3) 실습 DEMO (1)의 아날로그 필터와 디지털 필터의 주파수 응답의 크기를 그래프에 표시하라. (아날로그 주파수 ω에 대해 표시할 것.) (freqs, freqz 함수 사용) (그림 7 참고)

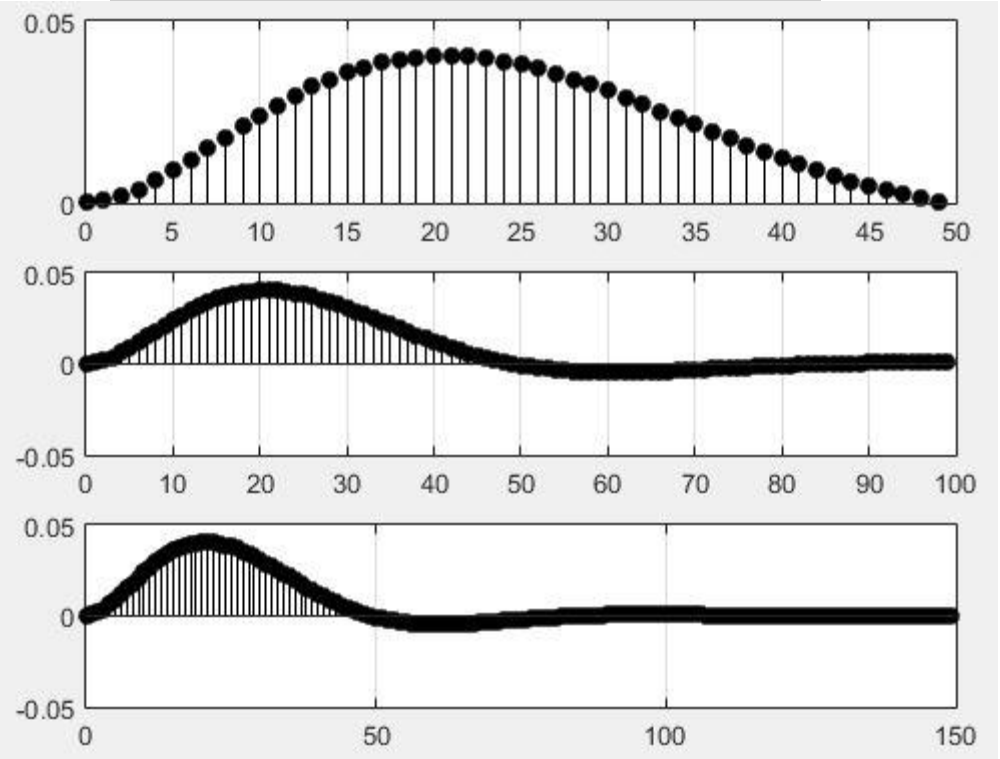
```
w = 0:0.001:2;Omega =  0:0.01:pi;
Ha = freqs(num,den,w);
figure(2);subplot(2,1,1);plot(w,abs(Ha),'b','LineWidth',2)
grid on;hold on;
Hd_1 = freqz(numd1,dend1,Omega);
plot(Omega/(1/5),abs(Hd_1),'g','LineWidth',2)
Hd_2 = freqz(numd2,dend2,Omega);
plot(Omega/(1/10),abs(Hd_2),'r','LineWidth',2)
axis([0,2,0,1.4])
legend('Analog Butterworth Filter.','Digiter Butterworth Filter.f_s = 5Hz','
```



겉으로 보기에는 많은 차이가 나지 않았지만 확대해서 보니 이론 수업때 배운 그대로의 결과물을 확인할 수 있었습니다. 5Hz로 샘플링 하는 것 보다 10Hz로 샘플링할때의 결과가 아날로그 필터와 더 비슷합니다. freqs 함수와 다르게, freqz 함수에는 관찰범위를 라디안으로 넣어주는 점을 고려하여 실습을 진행하였습니다.

(4) 실습 DEMO (1)에서 $f_s = 10$ Hz일때 구한 디지털 필터의 임펄스 응답을 구하고 $n = 0, 1, \cdots, L-1$에 대해 그래프에 표시하라. L값을 달리하며 임펄스 응답을 관찰하라. 이 시스템의 임펄스 응답이 유한한지 무한한지 판단하라. (filter 함수 사용) (그림 8 참고)

```
x1 = [1,zeros(1,49)];n1 = 0:49;
x2 = [1,zeros(1,99)];n2 = 0:99;
x3 = [1,zeros(1,149)];n3 = 0:149;
y1=filter(numd2, dend2, x1);
y2=filter(numd2, dend2, x2);
y3=filter(numd2, dend2, x3);
figure(3)
subplot(3,1,1);stem(n1,y1,'k','filled');grid on
subplot(3,1,2);stem(n2,y2,'k','filled');grid on
subplot(3,1,3);stem(n3,y3,'k','filled');grid on
```



L값을 demo 결과 예시와 동일하게 50, 100, 150 으로 설정하고 실습을 진행하였습니다. 결과를 확대해보면 임펄스 응답이 무한대로 갈수록 0으로 수렴하고 있지만, 유한한 응답은 아닙니다. 따라서 임펄스의 응답이 무한하다는 것과, 0으로 수렴한다는 것을 통해 이전에 pole - zero plot에서 판단한 안정성을 다시 한번 판단할 수 있었습니다.

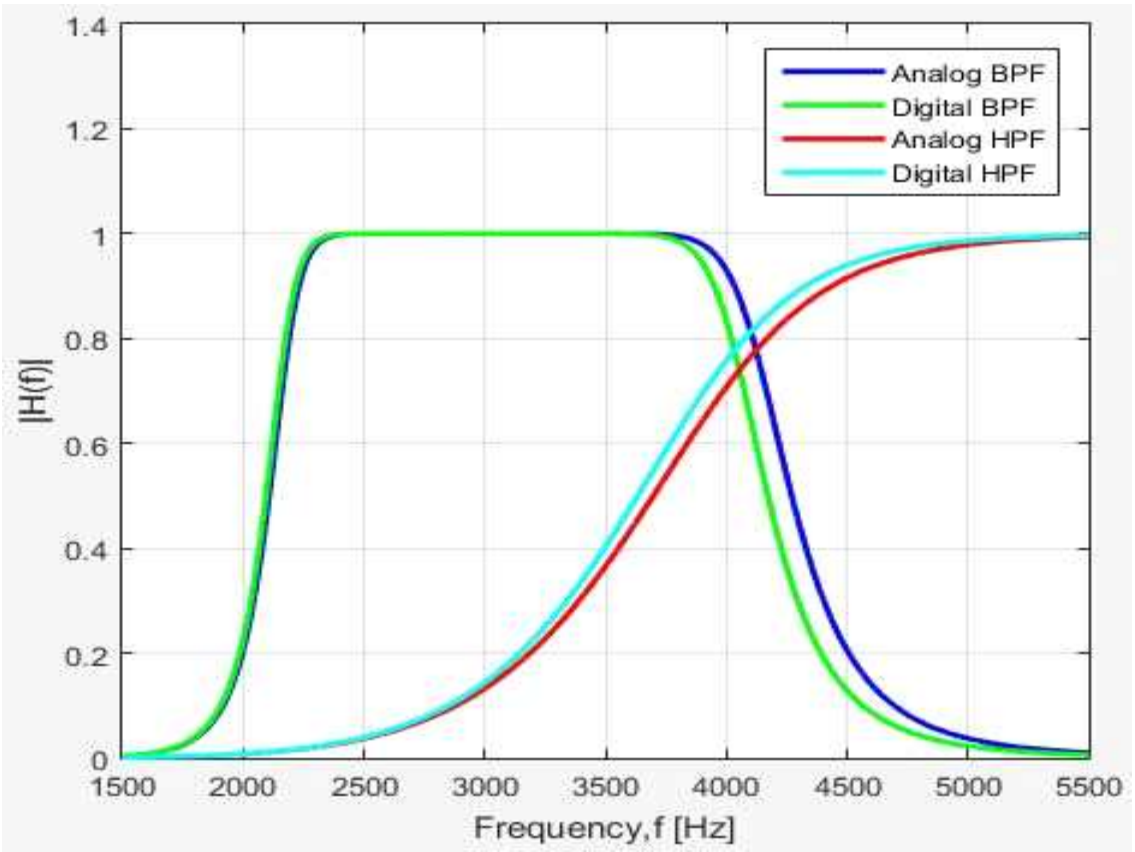
3.2 디지털 BPF, HPF 설계

(1) 실습 7차 아날로그 Butterworth 프로토타입 필터를 이용해 다음과 같은 아날로그 필터를 설계하라. (아날로그 프로토타입 필터 실습의 실습 3.2 (4) 참조) (buttap, lp2bp, lp2hp 함수 사용)

1. BPF, passband = [2000, 4000] Hz
2. HPF, cutoff 주파수 $f_c = 4000$ Hz

(2) 실습 DEMO 표본화 주파수 $f_s = 50,000$ Hz일때 bilinear transformation을 이용해 (1)의 BPF와 HPF를 이산화하여 IIR 필터를 설계한 뒤 주파수 응답을 아날로그 필터와 비교하라. (그림 9 참고)

```
clc;
clear;
[z, p, k] = buttap(7);
[num, den] = zp2tf(z,p,k);
f = 1500:0.1:5500;Omega =  0:0.001:pi;fs=50000;
w = 2*pi*f;

[numt_band, dent_band]=lp2bp(num, den, 3000*2*pi, 2000*2*pi);
H = freqs(numt_band,dent_band,w);
plot(w/(2*pi),abs(H),'b','LineWidth',2)
grid on;hold on;
[num1, den1]=bilinear(numt_band, dent_band, fs);
Hd = freqz(num1,den1,Omega);
plot(Omega/((2*pi)/fs),abs(Hd),'g','LineWidth',2)
axis([1500,5500,0,1.4])

[numt_high, dent_high]=lp2hp(num, den, 4000*2*pi);
H = freqs(numt_high,dent_high,w);
plot(w/(2*pi),abs(H),'r','LineWidth',2)
[numd1, dend1]=bilinear(numt_high, dent_high, fs);
Hd = freqz(numd1,dend1,Omega);
plot(Omega/((2*pi)/fs),abs(Hd),'c','LineWidth',2)
legend('Analog BPF','Digital BPF','Analog HPF','Digital HPF')
xlabel('Frequency,f [Hz]');ylabel('|H(f)|')
```



이전의 아날로그 필터는 f를 기준으로 코드를 구현하였는데, 아날로그와 디지털 필터의 규격을 맞춰주기 위해 입력값들을 w기준으로 주었습니다. freqs 함수의 관찰범위는 $3000\pi \sim 11000\pi$가 되고 freqz 함수의 관찰범위는 대칭성을 고려하여 $0 \sim \pi$로 설정해두었습니다. 두 필터를 f축 기준으로 바꿔서 비교를 진행하기 때문에, 아날로그 필터는 x축을 2π로 나눠주고, 디지털 필터는 $(2\pi * Ts)$로 나누어 주었습니다. 디지털 필터를 f축에 맞춰줄 때 아래와 같은 수식을 사용하였습니다.

$$w = \frac{\Omega}{T_s} = 2\pi f$$

$$\Rightarrow f = \frac{\Omega}{2\pi \cdot T_s}$$

3.3 신호의 필터링

(1) 실습 7차 아날로그 Butterworth 프로토타입 필터를 이용해 다음과 같은 아날로그 필터를 설계하라. (아날로그 프로토타입 필터 실습의 실습 3.2 (4) 참조) (buttap, lp2lp 함수 사용)

- LPF, cutoff 주파수 $f_c = 300$ Hz

(2) 실습 표본화 주파수 $f_s = 3000$ Hz일때 bilinear transformation을 이용해 (1)의 LPF를 이산화하여 IIR 필터를 설계하라.

```
1 -    clc;
2 -    clear;
3 -    [z, p, k] = buttap(7);%%1
4 -    Omega = 0:0.001:pi;f_s = 3000;
5 -    [num, den] = zp2tf(z,p,k);
6 -    [numt, dent]=lp2lp(num, den, 300*2*pi);
7 -    [numd1, dend1]=bilinear(numt, dent, f_s);
8 -    Hd = freqz(numd1,dend1,Omega);
9 -    figure(3)
10 -   subplot(324)
11 -   plot(Omega/((1/f_s)*2*pi),abs(Hd),'k','LineWidth',2);
12 -   hold on;grid on;axis([0,600,0,1])
13 -   xlabel('Frequency f[Hz]');ylabel('Frequency response');
```

cutoff 주파수가 300Hz이기 때문에, lp2lp 함수에는 라디안 값인 600π를 입력 변수로 주었습니다. 표본화 주파수가 3000Hz인 디지털 필터를 설계하기위해 앞서 문제들에서 구현했던 방법을 그대로 사용하였습니다. 마지막 plot함수에서도 Omega값을 수식을 통해 나온 값으로 나눠주며 f축으로 변환하여 그래프에 나타냈습니다. 실습 결과는 demo의 예시처럼 마지막에 하나의 figure에 담았습니다.

(3) 실습 다음과 같은 신호 $x[n]$을 발생하라.

$$x[n] = \cos\left(2\pi\frac{f_1}{f_s}n\right) + \cos\left(2\pi\frac{f_2}{f_s}n\right)$$

- $f_1 = 100$ Hz, $f_2 = 500$ Hz, $f_s = 3000$ Hz
- $n = 0, 1, 2, \cdots, N$, $N = 500$

(4) 실습 (3)의 신호 $x[n]$을 (2)에서 설계한 디지털 필터에 입력하여 출력 $y[n]$을 구하라. (filter 함수 사용)

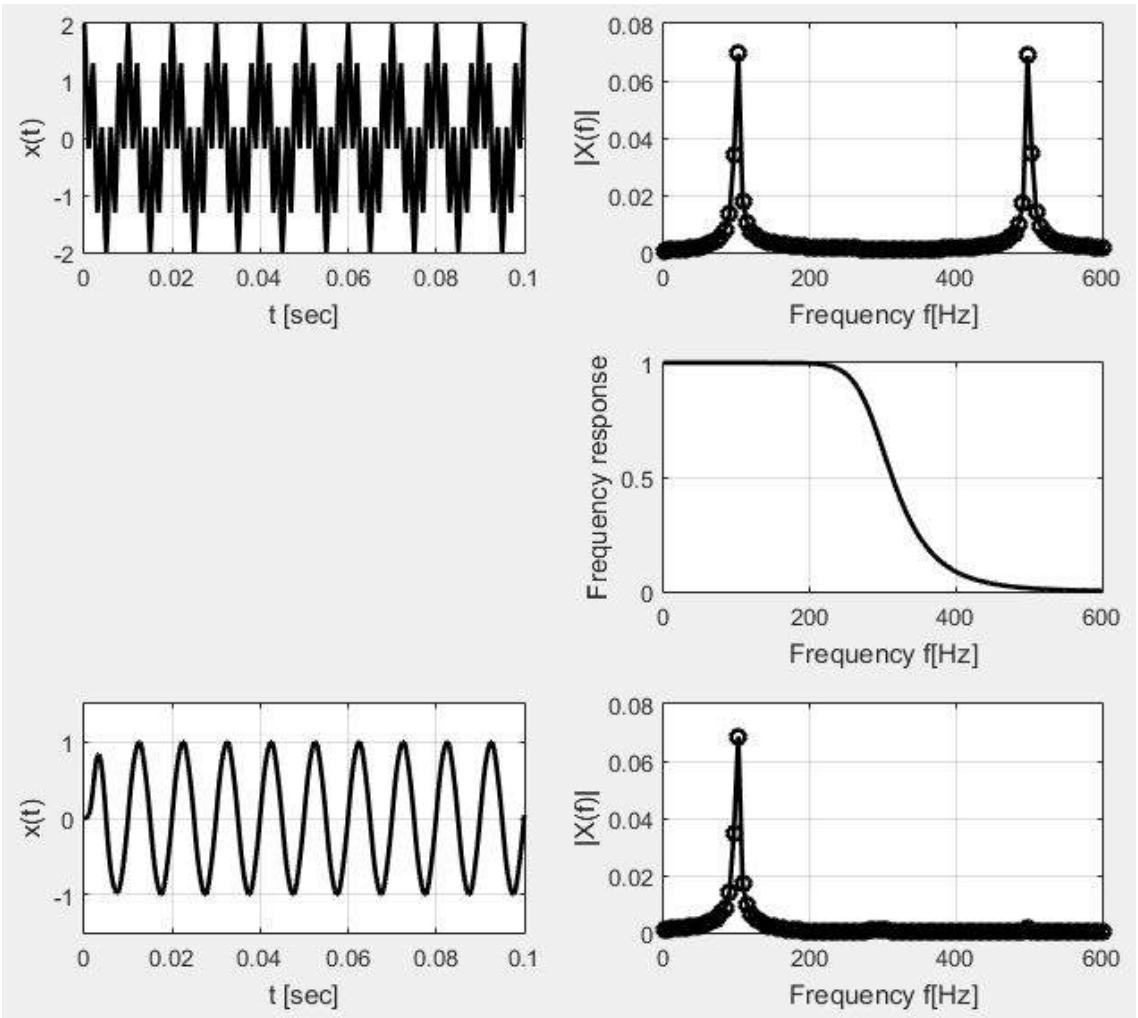
(5) 실습 $x[n]$과 $y[n]$의 크기 스펙트럼을 구하라. (myfun_SA 함수 사용, 표본화 실습 참고)

- $t = nT_s = \frac{n}{f_s}$ 관계를 이용하라.

(6) 실습 DEMO 입력신호 $x[n]$과 크기 스펙트럼, 출력신호 $y[n]$과 크기 스펙트럼, 디지털 필터 $H_d(z)$의 주파수 응답의 크기를 그래프에 표시하라. (그림 10 참고)

```
15 –    f_1 = 100;f_2 = 500;%%3
16 –    n = 0:499;t = n/f_s;
17 –    x = cos(2*pi*f_1*n/f_s)+cos(2*pi*f_2*n/f_s);
18 –    subplot(321)
19 –    plot(t,x,'k','LineWidth',2);axis([0,0.1,-2,2]);grid on;
20 –    xlabel('t [sec]');ylabel('x(t)');
21
22 –    y = filter(numd1,dend1,x); %%4
23 –    subplot(325);
24 –    plot(t,y,'k','LineWidth',2);axis([0,0.1,-1.5,1.5]);grid on;
25 –    xlabel('t [sec]');ylabel('x(t)');
26
27 –    subplot(322)      %%5
28 –    [f , X] = myfun_SA(t,x);
29 –    plot(f,abs(X),'k-o','LineWidth',2);axis([0,600,0,0.08]);grid on;
30 –    xlabel('Frequency f[Hz]');ylabel('|X(f)|');
32 –    subplot(326)
33 –    [f , Y] = myfun_SA(t,y);
34 –    plot(f,abs(Y),'k-o','LineWidth',2);axis([0,600,0,0.08]);grid on;
35 –    xlabel('Frequency f[Hz]');ylabel('|X(f)|');
```

(3)번 문제의 x[n]을 구현하기 위해 n을 0 ~ 499까지 500개를 만들어주고 주어진 수식을 통해 시간 변수 t를 설정 해주었습니다. (4)번 문제의 y[n]을 구현하기 위해 filter함수를 사용하였습니다. (5)번, (6)번 문제에서 x와 y 신호의 크기 스펙트럼을 보기 위해 my_funSA함수를 사용하였습니다.



결과물은 전형적인 LPF의 진행 과정을 단계별로 볼 수 있습니다. 어떠한 시스템에 시간 축 입력에 대한 출력을 확인하기 위해서 일반적으로 주파수 도메인으로 가져가서 처리를 하게 되는데, filter함수를 이용해서 간단하게 시스템의 출력(시간축)을 확인할 수 있었고, 마지막에 스펙트럼으로 주파수축에서 변화를 확인하게 됩니다.

시간 축 입 출력 신호를 분석해보면, x[n]은 두 개의 주파수 성분으로 합쳐진 주기 신호 인데, 출력 y[n]을 보면 마치 한 개의 주파수 성분만 가지고있는 주기 신호처럼 보입니다. 주파수 축 스펙트럼을 확인해보면 원인을 알 수 있는데, 100Hz와 500Hz성분을 가지고 있는 x[n]의 스펙트럼이 LPF를 만나 고주파는 저지되고 저주파는 통과가 되어 y[n]의 스펙트럼과 같은 결과가 나온 것을 볼 수 있습니다. 이상적인 LPF가 아니기 때문에 저지 대역에서 완벽한 저지를 하지 못하고 찌꺼기가 남게 됩니다. 결과적으로, y[n]의 스펙트럼의 500Hz 부분을 자세히 보면 아주 미세하게 아직 500Hz 성분이 남아있는 것을 확인할 수 있습니다.