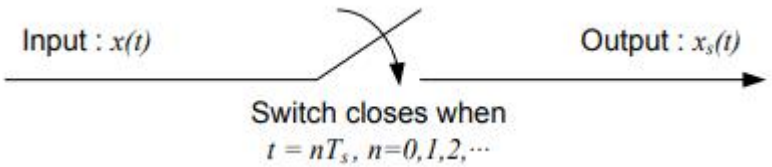
1. 개요

- 샘플링의 이해
- 샘플링한 신호의 복원
- 알리아싱과 나이퀴스트 주파수에 대하여

2. 본문

(1) 샘플링의 이해

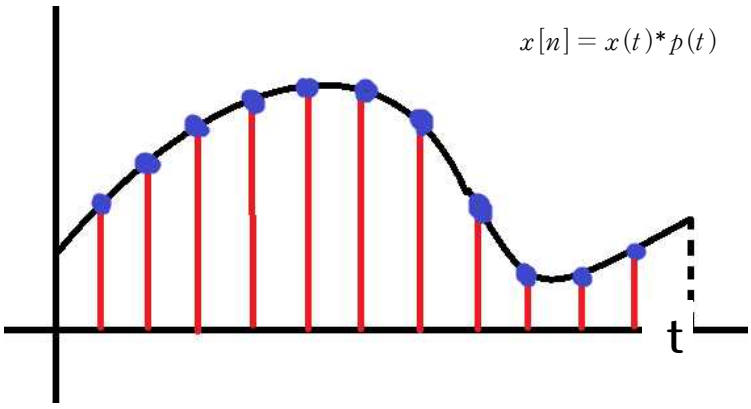
샘플링은 단순하게 생각하면 연속 신호를 비 연속 신호를 만드는 과정이라고 볼 수 있다. 실생활에서 모든 신호는 연속신호이다. 아날로그 신호라고 불리는데, 1초에 몇 개의 신호라고 딱 정의할 수가 없는 말 그대로 연속적인 신호이다. 이러한 아날로그 신호를 잘게 쪼개서 이진(디지털)화시키는 과정을 샘플링(표본화) 과정이라고 한다. 흔히 사용하는 컴퓨터, 스마트폰 등 디지털 세계에서는 모든 신호를 이러한 과정을 통해 나온 샘플링신호를 사용한다. 다음은 샘플링 과정의 알고리즘이다.



이전 실습의 컨볼루션 연산에 비하면 아주 간단한 알고리즘이다. 샘플링 주기의 정수배 마다 대상 신호의 값을 받아오고 받아온 값들을 순서대로 나열해주면 샘플링 신호를 완성할 수 있다. 이러한 샘플링 과정을 수식을 통해 자세히 알아보았다.

$$p(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT_s)$$

다음은 샘플링의 기본이되는 임펄스열이다. 시간축에서 샘플링은 위의 벌스파와 샘플링 대상신호를 단순하게 곱해주기만하면 완성된다.



검정색 신호가 원신호, 빨간색 신호가 임펄스열, 파란색 신호가 시간축에서 샘플링된 신호의 파형이다. 시간축에서 연속신호였던 함수가 이산신호로 바뀌면서 주파수축

에서도 변화가 일어난다. 먼저 연속 주기함수인 임펄스열의 Fourier series 는 다음과 같다.

$$p(t) = \sum_{k=-\infty}^{\infty} \frac{1}{T_s} e^{jk\omega_s t}$$

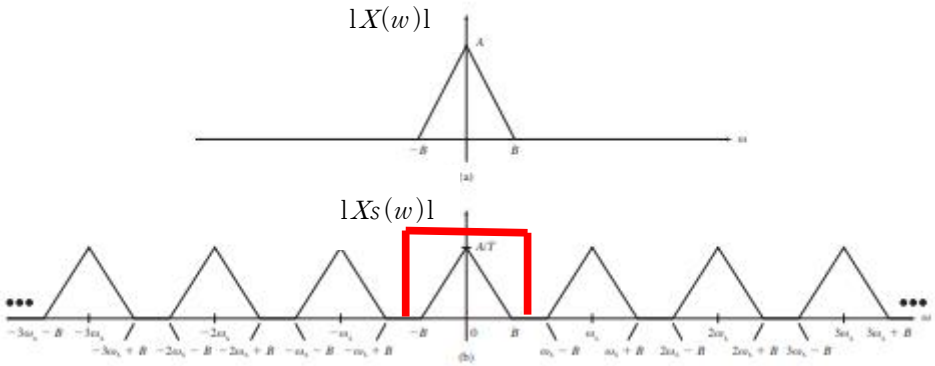
시간 영역에서의 임펄스열은 주파수 축에서도 크기와 주기가 변한 임펄스열이라는 것을 알 수 있다. 위의 식을 시간축 신호에 곱하고 Fourier Transform을 취해주면 다음과같은 결과물이 나온다.

$$\begin{aligned} X_s(\omega) = \mathcal{F}\{x_s(t)\} &= \mathcal{F}\left\{\frac{1}{T_s}\sum_{k=-\infty}^{\infty} x(t)e^{jk\omega_s t}\right\} \\ &= \frac{1}{T_s}\sum_{k=-\infty}^{\infty} \mathcal{F}\left\{x(t)e^{jk\omega_s t}\right\} \\ &= \boxed{\frac{1}{T_s}\sum_{k=-\infty}^{\infty} X(\omega - k\omega_s)} \end{aligned}$$

푸리에 변환의 주파수 이동 성질로 인해 원신호의 푸리에 변환된 신호가 샘플링 주기마다 반복되는 것을 볼 수 있다. 우리는 이러한 신호를 후에 나오는 실습문제의 결과를 통해 확인해볼 것이다.

(2) 샘플링한 신호의 복원

샘플링된 신호를 다시 실생활에서 쓰였던 신호로 만들려면 복원과정이 필요하다. 복원과정의 설명을 위해 먼저 위에서 설명한 과정이 완료된 결과물 사진을 보겠다.

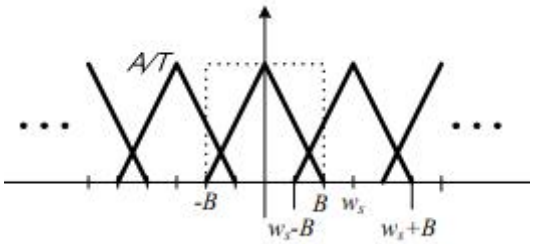


수없이 반복되는 스펙트럼 중 복원에 필요한 부분은 딱 한주기만큼이다. 빨간색 네모 안의 신호만 뽑아내고 줄어든 크기만큼 증폭해준다면 이론상으로는 다시 원래 아날로그신호의 스펙트럼과 같은 결과가 될 것이다. 샘플링 신호의 복원은 이러한 알고리즘으로 이루어지며, 그림의 빨간 네모박스를 구현하기 위해 Low Pass Filter를 사용하게 된다. 이론상으로는 완벽하게 복원이 가능하지만, 실제로는 그림과 같은 직

사각형 모양의 이상적인 필터를 구현하지 못한다. 현실적인 복원 결과물은 실습문제의 결과물을 통하여 복원되는 스펙트럼을 확인 할 수 있다.

(3) 알리아싱과 나이퀴스트 주파수에 대하여

알리아싱이란 신호가 왜곡되는 현상을 말한다. 직전 그림에서는 샘플링은 충분히 큰 주파수로 샘플링하였기에 샘플링 후 신호의 스펙트럼을 보면 파형 간 간격이 충분한 것을 볼 수 있다. 하지만 그림에서 원 신호의 정보가 담긴 최대 대역폭인 주파수 B 보다 낮은 주파수로 샘플링을 한다면 다음 그림과 같은 현상이 발생한다.



그림과 같은 현상이 일어난 상태에서 Low Pass Filter를 사용하면 원신호의 파형과는 많이 다른 결과가 나올 것이다. 이러한 현상을 방지하기 위해 Nyquist라는 사람이 샘플링 복원을 위한 최소 주파수를 정의했다.

$$\omega_s > 2B$$

원 신호의 스펙트럼에서 최대의 대역폭 보다 최소 2배는 넘는 주파수로 샘플링해줘야 복원과정에서 왜곡현상이 일어나지 않는다는 정의이다. 사실 샘플링 주파수는 높으면 높을수록 복원 시 원 신호와 매우 비슷해 지지만, 계산횟수, 용량 등의 문제 때문에 최소 주파수가 정의된다고 한다.

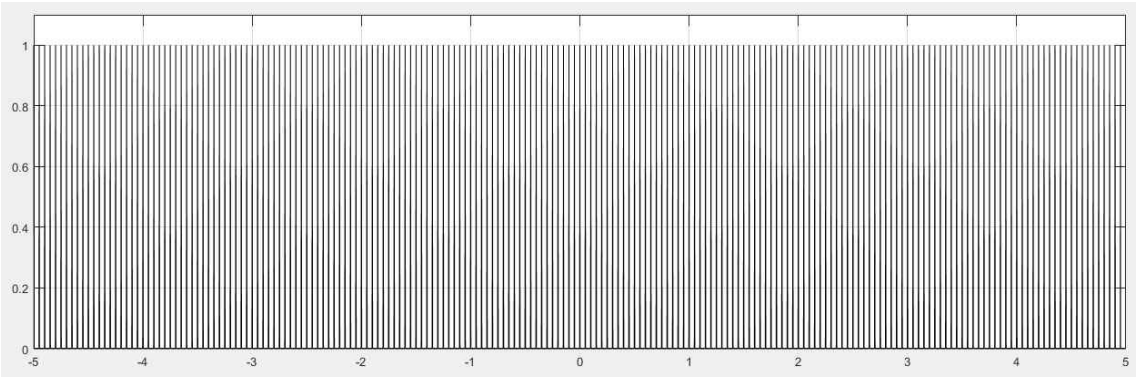
3. 실습 결과

- **실습** 구현한 임펄스열을 다음 파라메터에 대해 구하고 그래프에 표시하라.
  - $t_1 = -5,\ t_2 = 5$
  - $f_s = 20$ Hz
  - $N = 4096$

임펄스열 함수의 소스코드와 결과 사진입니다. 원래는 아래의 네 줄짜리 함수를 사용하였지만, 데모 답안 그래프처럼 보기 좋게 나오지 않아서 좀 더 세밀한 분석을 통해 함수를 구현하였습니다. 자세한 설명은 주석으로 적어놨습니다.

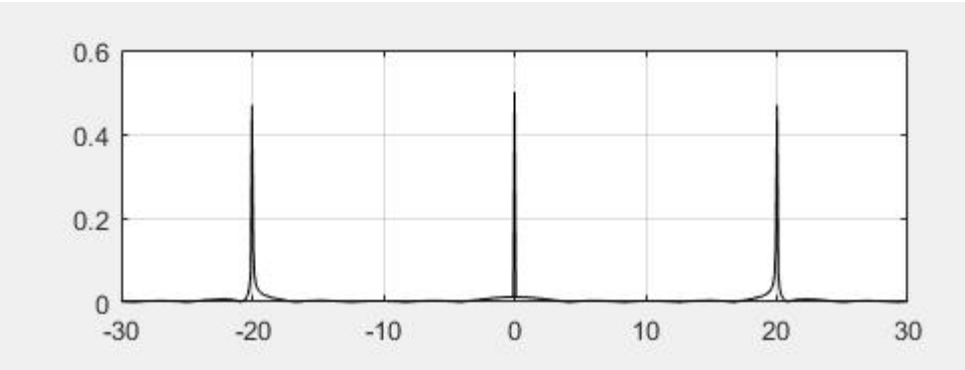
```
function [t, P] = t3_pulse_train(t1, t2, N, fs)
if(fs == 10 || fs == 15 || fs == 20)

    T = round(1/fs,4); % 먼저 주기를 계산해서 소수 다섯째자리에서 반올림
    num_1 = fs * (t2-t1);%펄스파열에서 1의  갯수
    num_zero = fix((N - num_1)/(num_1-1)); %1과1사이 의 0 갯수
                                %ㄴ> 이부분 설명 예시는 150개의 펄스파가 있다고가정하면
                                %    149개까지만 계산하고 나머지 하나의 펄스파에 남는 0을 추가

    t = t1:T/(num_zero+1):t2-T;% t2에서 한주기전까지 시간을 계산
    P = zeros(1,N);
    for i = 1:length(t) %% T까지 주기에 대응하는 시간값마다 1을 넣어줌
        if(mod(round(t(i),4),T) == 0)
        P(i) = 1;
        end
    end
    if(length(t) <N) %% 부족한 N만큼 뒤에 시간을 추가해줌(00이 들어갈 시간)
        t = [t, linspace(t(end),t2-T/(num_zero+1),N-length(t)+1)];
        t = unique(t);%%위계산에서 중복시간값이 하나 나와서 빼줌
    end
    %시간축을 따로 계산한 이유는 한주기를 먼저 빼고 반올림등을 사용해서 알맞은 시간값에 1을 넣어주고
    %나중에 t2를 초과하지않고 N의갯수가 부족한 만큼의 0을 추가해주는 방식

else %% 주파수가 10,15,20 이 아닐때 단순하게 짬. 위에꺼는 10,15,20만 잘됨
t = linspace(t1,t2,N+1);
t = t(1:end-1);
P = zeros(1,N);
P(1:round(N/((t2 -t1)*fs)):end) = 1;

end
```

(A)



- **실습** 임펄스열 $p(t)$의 Fourier Transform을 (손으로) 계산하라.

- **실습** 임펄스열 $p(t)$의 크기 스펙트럼을 (MATLAB으로) 구하여 그래프에 표시하고 손으로 계산한 Fourier Transform과 비교하라.



-매트랩으로 구현한 Fourier Transform

(별다른 알고리즘 없이 앞에서 구현하였던 함수에 파라메터를 대입, 결과를 `myfun_SA()` 함수와 `abs()`한수를 사용하여 구현하였습니다.)

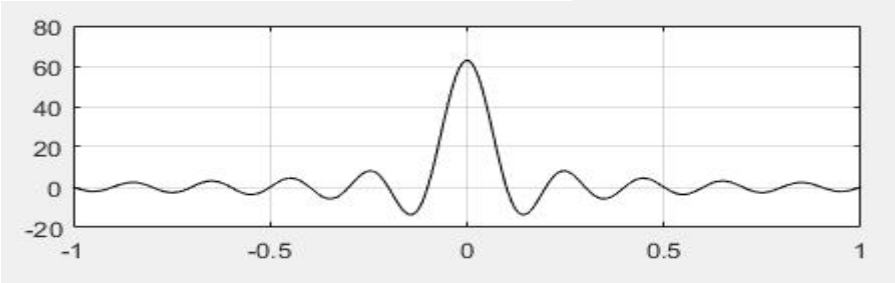
3.2 Sinc함수 발생

- 실습 다음과 같은 신호 $x(t)$를 발생시키고 그래프로 표시하라.

$$x(t) = \tau \, \mathrm{sinc}\left(\frac{\tau t}{2\pi}\right)$$

  - t : 실습 3.1에서 구한 t를 사용
  - $\tau = 20\pi$

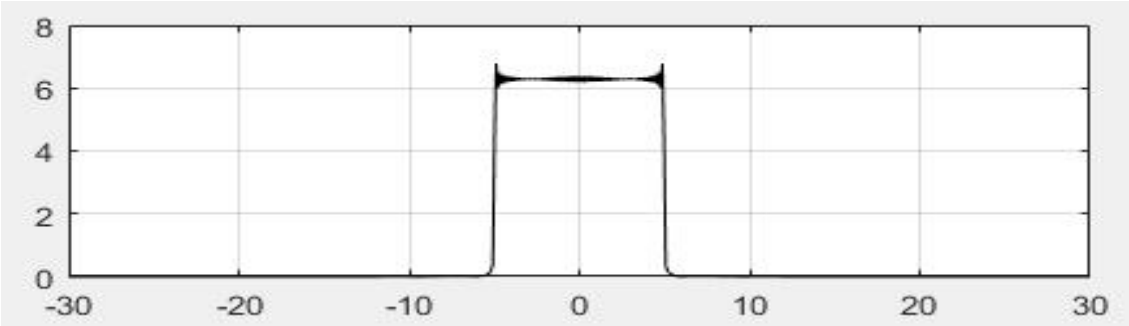
```
t1 = -5;
t2 = 5;
[t , p] = t3_pulse_train(t1, t2, N, f);

tau = 20*pi;
y = tau*sinc((tau*t)/(2*pi));
```



임펄스열 구현을위한 함수의 시간 축 결과를 sinc() 함수의 시간축에 그대로 대입했습니다.

- 실습 신호 $x(t)$의 Fourier Transform을 (손으로) 계산하라.

- **실습** 신호 $x(t)$의 크기 스펙트럼을 (MATLAB으로) 구하여 그래프에 표시하고 손으로 계산한 Fourier Transform과 비교하라.



-매트랩으로 구현한 Fourier Transform

(별다른 알고리즘 없이 앞에서 구현하였던 함수에 파라메터를 대입, 결과를 `myfun_SA()` 함수와 `abs()`한수를 사용하여 구현하였습니다.)

- **실습** 이 신호를 구성하는 가장 높은 주파수 B Hz는 얼마인가?

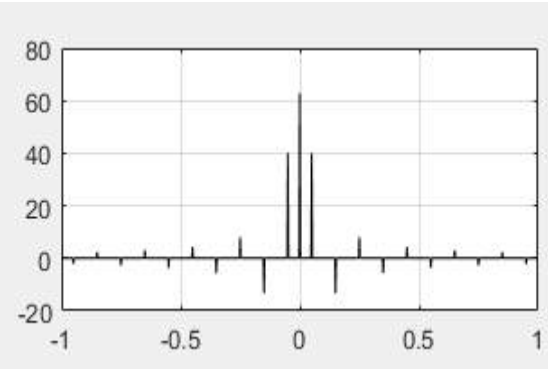
±5Hz 구간동안 유효한 신호가 있으므로 가장 높은 주파수는 5Hz입니다.

3.3 표본화

- 실습 앞서 발생한 임펄스열을 이용해 sinc함수를 표본화하고 표본화된 신호를 그래프에 표시하라. 표본화된 신호 $y(t)$는 다음과 같다.

$$y(t) = x(t)p(t)$$

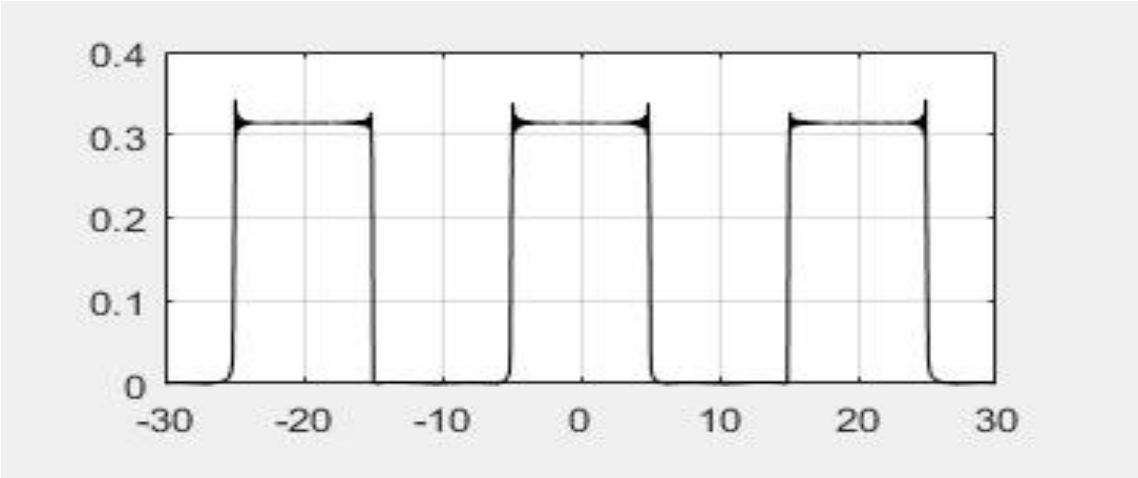
```
t1 = -5;
t2 = 5;
N = 4096;
f = 20;


t = t2 - t1;
[t , p] = t3_pulse_train(t1, t2, N, f);

tau = 20*pi;
y = tau*sinc((tau*t)/(2*pi));
Y = p.*y;
```



-시간축 샘플링 과정이기 때문에 앞에서 구현한 임펄스열에 싱크함수를 곱해주었습니다.

- 실습 표본화된 신호 $y(t)$의 Fourier Transform을 (손으로) 계산하라.

• 실습 표본화된 신호 $y(t)$의 크기 스펙트럼을 (MATLAB으로) 구하여 그래프에 표시하고 이와 같은 스펙트럼이 나오는 이유를 설명하라.




```
y = tau*sinc((tau*t)/(2*pi));      plot(f1,abs(y_1),'k','MarkerFaceColor','k')
Y = p.*y;                          axis([-30,30,0,0.4]);
[f1 , y_1] = myfun_SA(t,Y);        grid on;
```

-아래의 수식을 참고하여 설명을 하겠습니다.

$$= \boxed{\frac{1}{T_s}\sum_{k=-\infty}^{\infty} X(\omega - k\omega_s)}$$

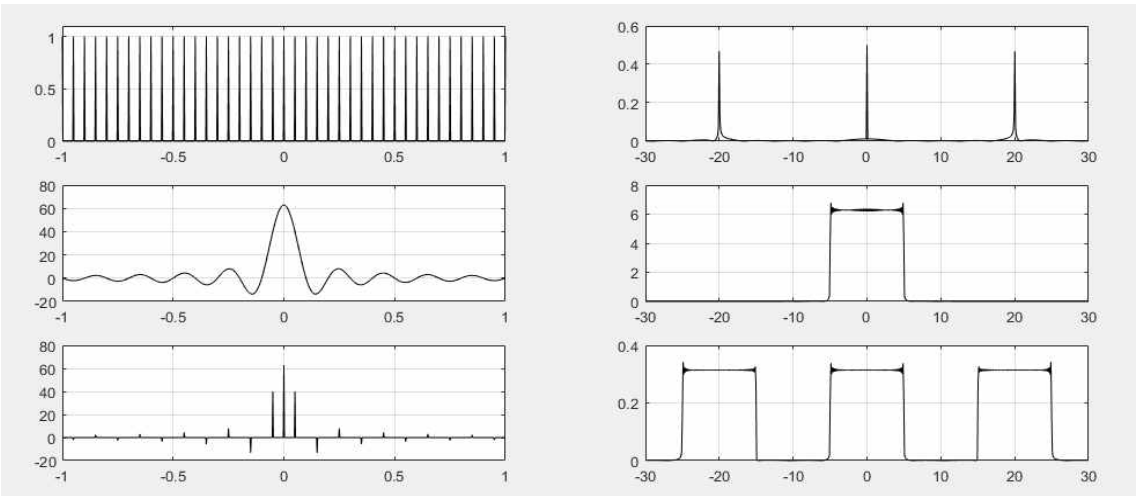
위 수식은 시간축에서 샘플링과정을 끝낸 신호 x(t) 의 푸리에변환 결과 식입니다. 수식을 해석해보면 -∞에서 ∞ 까지 Ws마다 X(w)가 반복됩니다. 표본화된 신호의 스펙트럼 결과를 보면, 3.2절에서 확인한 구형파의 파형이 그대로 반복되는 것을 보여주고 있으므로 결과가 정확하게 나왔다고 생각한다. 스펙트럼의 파형은 정확하게 맞았지만 크기는 수식과 다른 모습을 보여주는데, 매트랩에서 푸리에 변환을 구현하기위해 사용한 함수는 실제 푸리에 변환이 아닌 DFT 알고리즘을 사용하여 구현하였기 때문입니다.

3.4 시간영역과 주파수영역에서의 표본화 비교

- 실습 DEMO 실습 3.1 ~ 3.3의 그래프를 한 화면에 표시하고 시간영역과 주파수영역에서의 표본화 과정을 비교하라. (그림 7 참조)

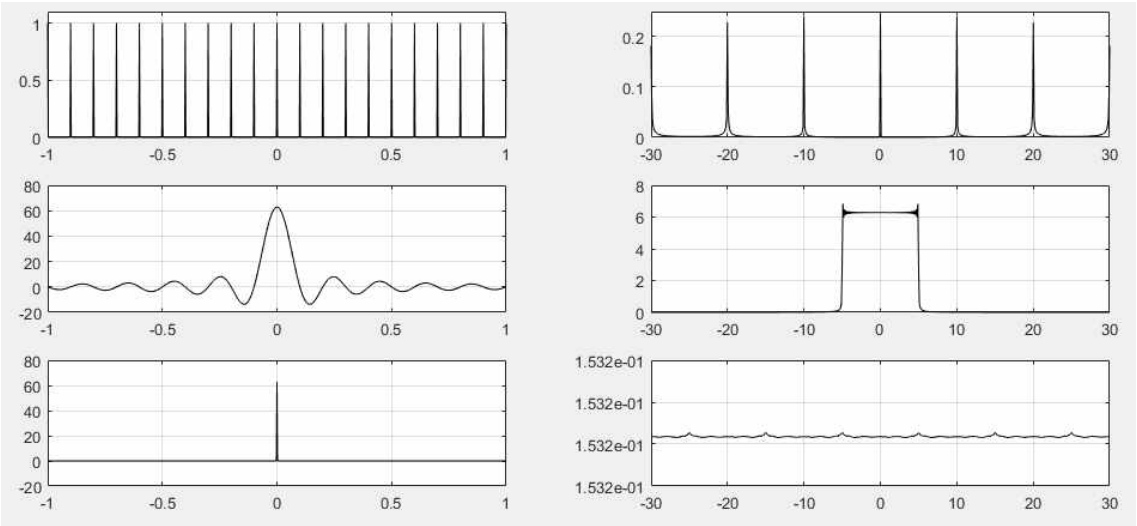
```
t1 = -5;
t2 = 5;
N = 4096;
f = 20;

[t , p] = t3_pulse_train(t1, t2, N, f);
%주어진 조건의 임펄스열 발생

tau = 20*pi;
y = tau*sinc((tau*t)/(2*pi)); %시간축에서 sinc함수 구현
Y = p.*y;  %시간축에서 sinc함수의 샘플링과정

[f_1 , p_1] = myfun_SA(t,p); %임펄스열의 스펙트럼
[f0 , y_0] = myfun_SA(t,y); %sinc함수의 스펙트럼
[f1 , y_1] = myfun_SA(t,Y); %시간축 함수의 샘플링된 신호의 스펙트럼

figure(1)
subplot(3,2,1)
plot(t,p,'k','MarkerFaceColor','k')
axis([-1,1,0,1.1]);
grid on;
```

subplot(3,2,1)아래부분은 모두 그려주는 부분이라 생략하였습니다.
다음은 결과 그래프입니다. 샘플링 후 반복되는 스펙트럼 결과물에서 스펙트럼 간 간격이 10 Hz로 충분한 것을 볼 수 있습니다.

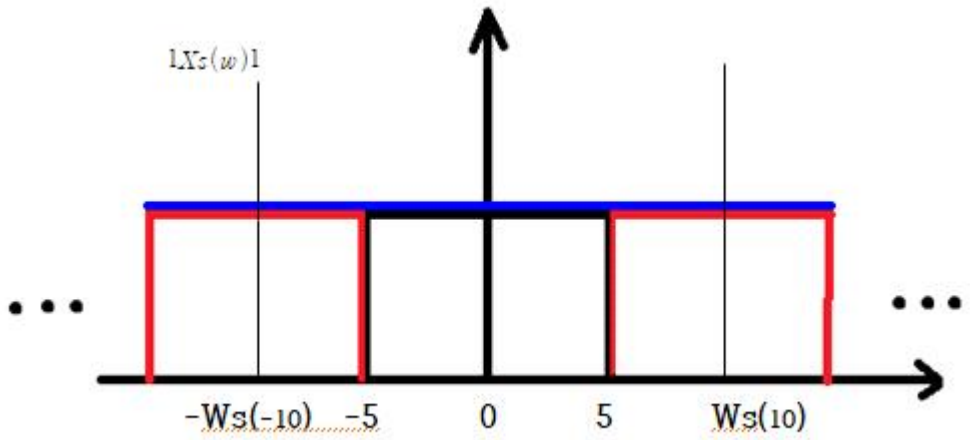


- 실습 DEMO 표본화 주파수를 다음과 같이 바꿔가며 실습 3.1 ~ 3.3를 반복하며 Nyquist Sampling Rate에 대해 설명하라. (그림 8, 9 참조)

$$f_s = \{10\ \text{Hz},\ 15\ \text{Hz}\}$$



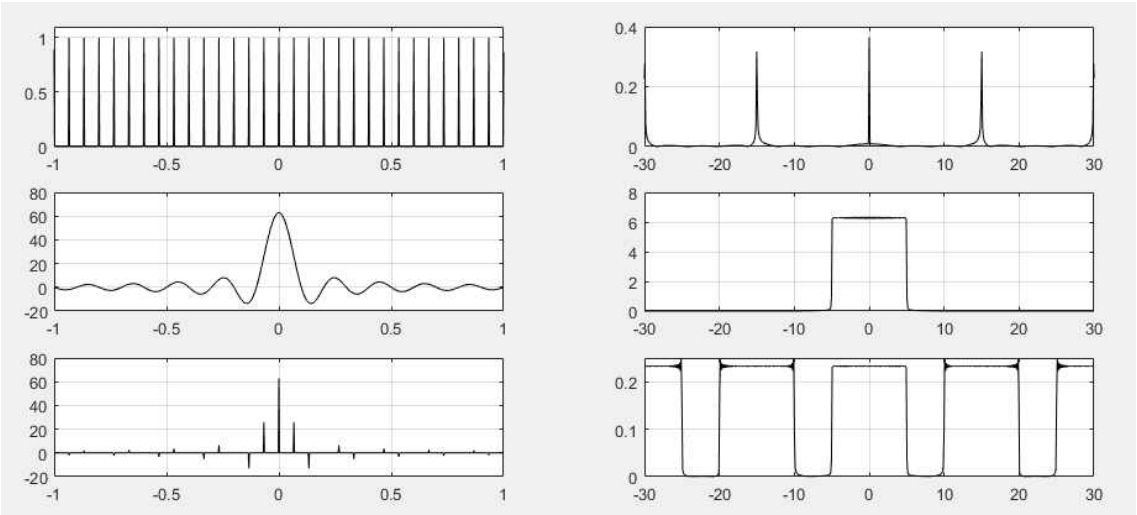
10Hz에서의 결과입니다. 알리아싱현상을 목격할 수 있습니다. Nyquist 주파수는 앞에서 정의했던 것처럼 Ws > 2B 인데, 현제 샘플링주파수가 10Hz 는 정확히 두배가되는 주파수여서 원신호의 sinc파형의 스펙트럼이 아래 그림처럼 겹쳐지게됩니다.



위의 그림을 보면 빨간색 부분의 중심부분이 정확히 10Hz가되어 폭이 5Hz인 구형파가 서로 겹치는 모습을 볼 수 있습니다. 결과적으로는 파란색 직선 모양의 파형이 나올 것 으로 예측되었고, 결과 그래프3행2열에 직선모양의 결과가 나오는 것을 확인했습니다. 따라서 알리아싱이 일어나지 않기 위해서는 원신호의 대역폭의 2배를 넘는 주파수로 샘플링하여 Nyquist 주파수를 지켜주는 이유를 알 수 있었습니다.

10Hz 결과에서 특이한점이 한 가지 더 있는데, 3행1열의 결과 그래프가 하나의 임펄스

신호와 같이 나온 것입니다. 이유는 실습에 사용할 sinc함수의 0이되는 주기가 0.1초인데 임펄스열의 주기도 0.1초이기 때문에 n = 0 일때의 점을 제외하면 sinc함수의 값이 0이 될 때만 샘플링이 되기 때문입니다. 이어서 15Hz로 샘플링을 진행하였습니다.



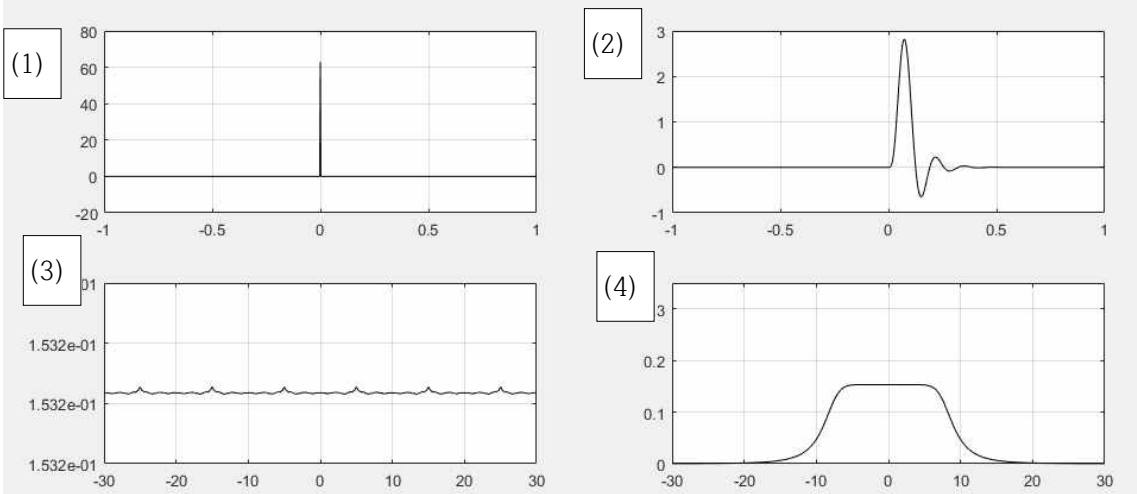
15Hz에서 결과입니다. 스펙트럼 간 간격이 5Hz로 늘어나고 20Hz 샘플링 결과처럼 깔끔하게 결과가 나온 것을 볼 수 있습니다.

- 실습 DEMO 실습 3.4에서 표본화한 각 신호, 즉 표본화 주파수가 10, 15, 20 Hz인 경우의 표본화 신호를 cutoff 주파수 8 Hz인 5차 버터워스 LPF를 이용해 복원하고 그 결과를 시간영역과 주파수영역에서 관찰하라. (그림 10, 11, 12 참조)

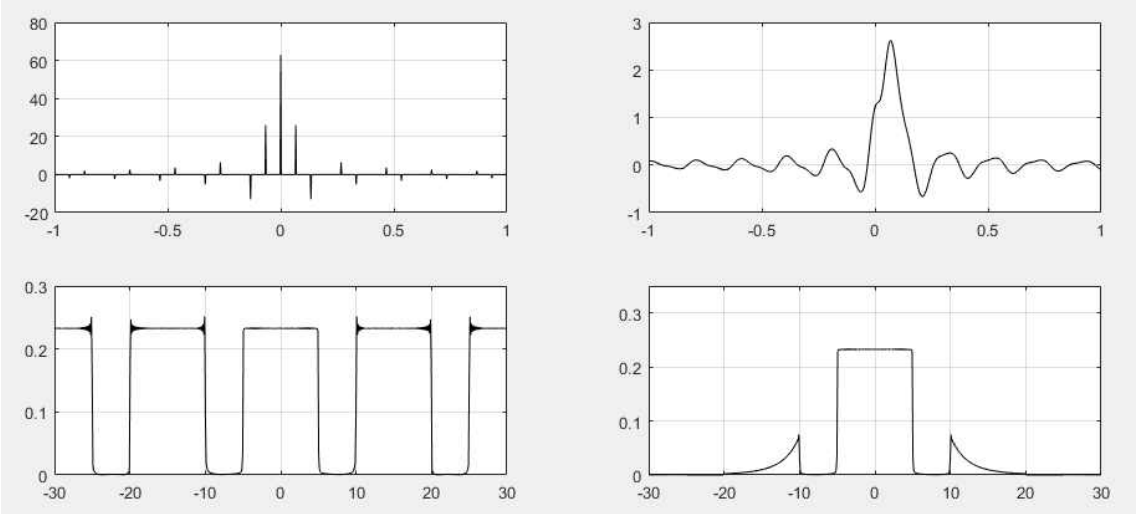
```
t1 = -5;
t2 = 5;
N = 4096;
f = 10;
tau = 20*pi;

[t , p1] = t3_pulse_train(t1, t2, N, f);
y = tau*sinc((tau*t)/(2*pi));

fc=8 ; %8차 버터워스필터 사용
[z, p, k] = buttap(5) ;
[num, den] = zp2tf(z,p,k) ;
[num, den] = lp2lp(num, den, 2*pi*fc) ;
[num_d, den_d] = bilinear( num, den, 1/abs(t(2)-t(1)) ) ;

Y = p1.*y; %시간축 샘플링
[f1 , y_1] = myfun_SA(t,Y); %시간축 샘플링 신호의 스펙트럼
y_out = filter(num_d, den_d,Y); %시간축 샘플링신호의 복원
[f_1, y1_out] = myfun_SA(t,y_out);%샘플링 완료된 신호의 스펙트럼
```

(이후 figure부분은 생략)

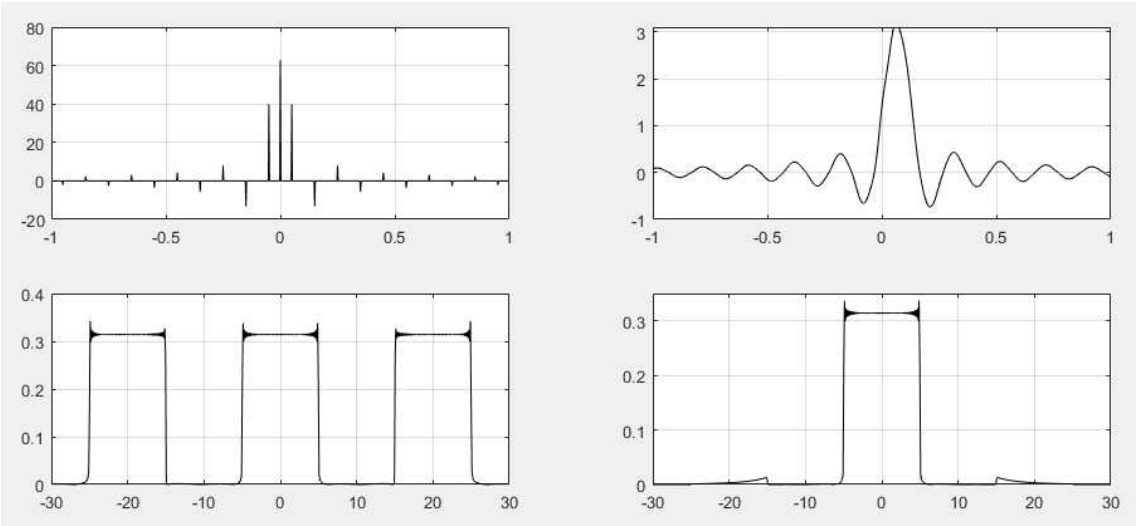
실습에 사용하도록 주어진 8차 버터워스필터를 사용하였습니다. 이어서 결과입니다.



(1) 시간축 샘플링신호, (2) 시간축 샘플링신호의 복원, (3) 시간축 샘플링신호의 스펙트럼, (4) 복원된 신호의 스펙트럼 그래프입니다. 앞에서 설명한 것처럼 알리아싱이 발생했기 때문에 복원이 제대로 되지 않은 결과를 볼 수 있습니다.



위 코드에서 f부분을 바꾸면서 결과를 보았습니다. 위 결과는 Fs가 15일때의 결과입니다. 알리아싱은 일어나지 않았지만 이상적인 필터가 아니여서 옆 스펙트럼의 신호의 간섭이 있는 결과를 볼 수 있습니다.



마지막 Fs가 20일때의 결과입니다. 충분한 샘플링 주파수를 사용했기 때문에 원래의 신호와 상당히 비슷하게 복원이 된 것을 볼 수 있습니다.

4. 실습 소감