설계 프로젝트 결과 보고서

20160408 강민수, 20160450 김용준, 신요안 교수님

1. 프로젝트 소개

이 프로젝트에서는 다양한 Amplitude Modulation과 Angle Modulation 방식을 구현해 본다. 코드로는 DSB, AM, Wideband FM 방식을 구현하였고, Simulink로는 AM, Wideband FM, Narrowband FM을 구현하였다. 또한 다수의 방식에 대해, Noise 채널이나 Multiplexing을 구현하여 보고, 동작을 확인한다.

프로젝트는 크게 Matlab Coding과 Simulink simulation Tool로 나누어 분담하였으며, 결과물을 이론과 검증 한 뒤, 서로의 결과물을 비교하며 검증 및 분석하는 방식으로 진행하였다.

- 실행 방법(동일 작동 검증시 실행 방법)

pre) audio.flac파일을 제외한 나머지 wav파일을 제거한다.

1) AM.slx를 실행한 후, 입력(1개, audio.flac)과 출력물(2개) 모두에 동일 디렉토리의 다음 이름에 맞게 설정한다. (AM_modulated.wav와 AM_demodulated.wav)

2) narrow_FM.xls를 실행한 후, 입력(1개, audio.flac)과 출력물(2개) 모두에 동일 디렉토리의 다음 이름에 맞게 설정한다. (narrow_FM_modulated.wav와 narrow_FM_demodulated.wav)

3) wide_FM.xls를 실행한 후, 입력(1개, audio.flac)과 출력물(2개) 모두에 동일 디렉토리의 다음 이름에 맞게 설정한다. (wide_FM_modulated.wav와 wide_FM_demodulated.wav)

4) Communication_System_Project_code.m 파일을 실행한다.(workspace는 파일이 현재 있는 디렉토리로 설정한다)

5) 그래프 두 개를 확인하며, demodulated.wav 파일들을 재생하여 동작을 확인한다.

6) received**.wav파일들을 통해 code로 인해 생성된 파일 들의 복조 형태를 확인한다.

--> modulated file은 스펙트럼 확인용입니다.

2. 역할분담

1) 20160408 강민수
- Simulink에서 AM, Narrowband FM, Wideband FM 구현
- 최종보고서: 이론 설명, AM, FM의 블록도 설명 작성 및 발표
- Matlab code와 simulink 결과물과의 비교 그래프 code 작성
- UCC ppt 제작 및 녹화

2) 20160450 김용준
- Matlab DSB, AM, FM 과 스펙트럼, Wave plot code 작성
- Matlab과 Simulink 비교 그래프 UCC 설명
- FM demodulation에서 Hilbert transform 사용방법 구현
- 최종 보고서 : 프로젝트 소개, 코드 설명, 결과물 설명
- UCC 영상 편집

3. simulink

3.1 이론 설명

- AM(Amplitude Modulation)

AM은 Amplitude Modulation의 약자로서 신호의 크기를 변조하여 Message를 전달하는 신호이다. AM 변조 방식은 두 가지로 DC bias를 더하고 carrier signal을 더하는 방식과 DSB 변조 후 Carrier signal을 더하는 방식이 존재한다.

첫 번째 방식은 DC bias를 더하고, carrier signal로 DSB 변조해서 전송하는 방식이다.

$$x_c(t) = (A + m(t))A_c' \cos w_c t$$

두 번째 방식은 message signal의 min value를 통해 generalized하고 modulation index를 곱하고 나서 carrier signal로 DSB 변조와 더하는 방식이다.

$$x_c(t) = A_c(1 + am_n(t))\cos w_c t, m_n(t) = m(t)/|\min m(t)|$$

첫 번째 방식은 DC bias를 통해 modulation index를 직접 결정하기 어렵기 때문에 두 번째 방식을 사용하여 modulation index를 parameter로서 결정하였다. 0<a<1값은 가장 power가 클 때, 출력으로 나올 signal의 소리 크기가 적절하므로 a=1로 설정하여 결정하였다. normalized message signal을 사용하여 DSB 변조를 통해 AM Modualtion을 결정하기로 하였다.

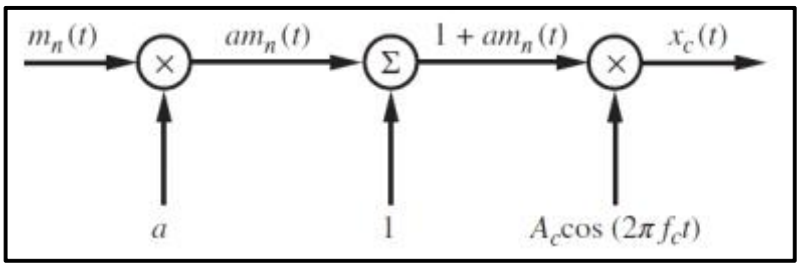


그림 1 두 번째 AM modulation 방식

Demodulation 또한 크게 두 가지 종류가 존재한다. AM에서는 coherent demodulation과 envelop detection방식이 존재한다. Coherent demodulation은 Carrier signal을 곱하여 삼각함수 배각 공식을 활용한 복조방식이며, envelop detection방식은 주파수의 샘플링 주기와 peak값의 Capacitor의 전압값 형태를 활용한 복조방식이다. 프로젝트에서는 phase가 동일하다는 가정 하에 coherent demodulation을 기반으로 DSB변조하여 low pass filter를 사용하였다. 하지만 low pass filter이후에 DC term을 제거하여야 온전한 message signal이 반환되므로 이러한 기능 또한 추가하였다.

- FM(Frequency Modulation)

FM은 Frequency Modulation의 약자로서 신호의 위상을 변조하여 Message를 전달하는 신호이다. FM 변조 방식은 두 가지로 narrow band와 wide band 변조방식이 존재한다.

첫 번째 방식은 narrow band 변조방식으로 교과서에 수록된 Power Series expansion에 의한 Block diagram을 기반으로 제작하였다. 기본적으로 Frequency Modulation이므로 적분을 통해 phase로 만들어주고 frequency deviation constant를 곱한 후, Carrier신호를 곱하고 더하여 아래 그림2의 모양과 동일하게 구성하였다.

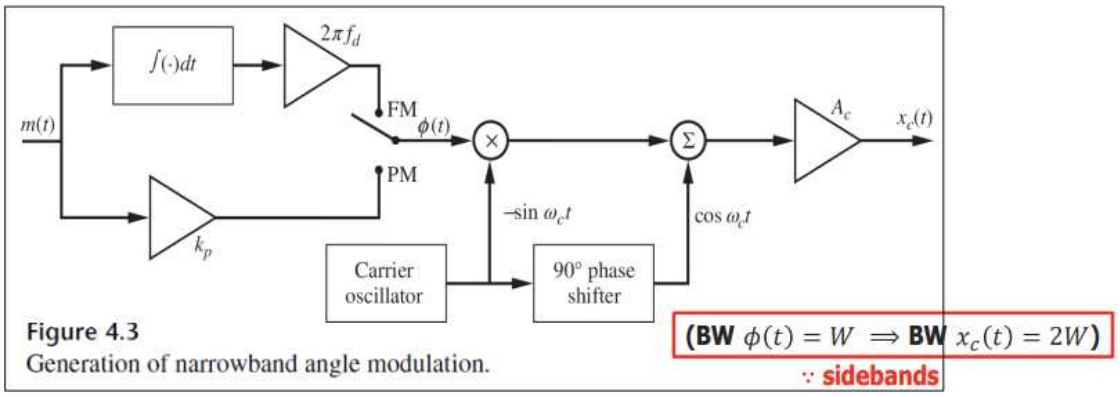


그림 2 FM Narrow band Modulation

Demodulation은 FM에서 두 가지 종류가 존재한다. Discriminator 기반의 유추 복조와 PLL(Phase Locked Loop)기반의 복조방식이 존재한다.

Discriminator 방식의 복조는 잡음을 limiter와 bandpass filter로 처리하며, differentiator로 phase내의 message signal을 amplitude화 한 후, 이를 envelop detector가 잡아내어 message signal로 반환하는 방식이다.

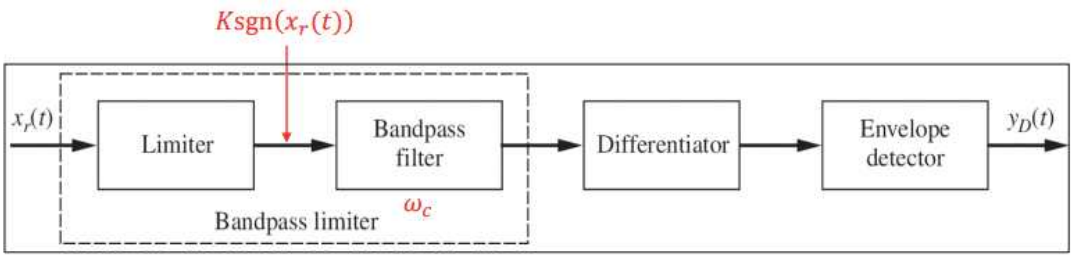


그림 3 Discriminator 방식의 FM demodulation

Phase Locked Loop 방식의 복조는 잡음과 위상차를 Phase detector와 VCO를 통해 극복하며, 반환된 신호를 differentiate를 통해 message signal을 반환하는 방식이다.

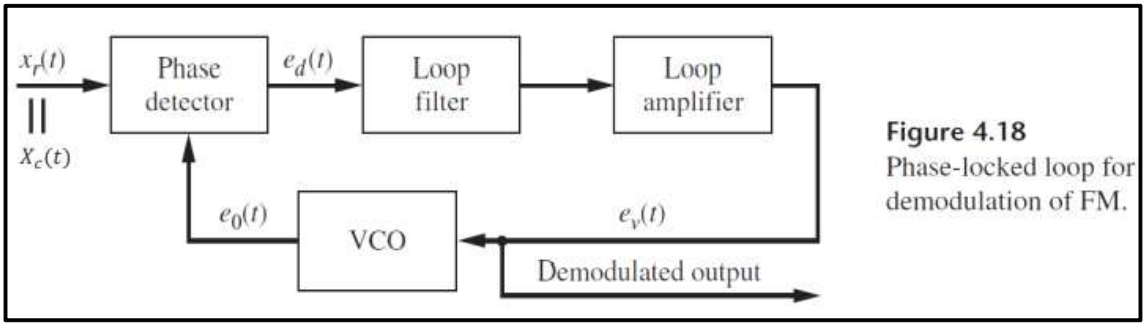


Figure 4.18
Phase-locked loop for demodulation of FM.

그림 4 PLL의 Block Diagram

프로젝트에서는 narrow band FM 변조와 wide band FM변조 방식을 모두 사용하여 비교하였으며, demodulation에서는 discriminator방식을 사용하였으며, Envelop Detector에 low pass filter를 사용하여 유사하게 기능하도록 조절하였다.

Narrow band에서는 그림 2와 같이 동일하게 구성하였으며, wide band는 다음의 식처럼 구성하였다.

$$x_c(t) = A_c \cos(w_c t + \phi(t)),\ \phi(t) = \beta \int^t m(a)da$$

3.2 AM

본 프로젝트에서 구현한 전체 블록도는 다음과 같다.

Subsystem 기능을 활용하여 Modulation과 Demodulation으로 크게 분리하여 진행하였다.

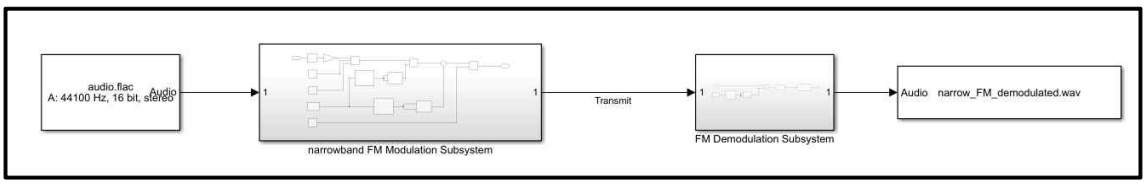


그림 5 AM modulation 과 demodulation 블록도

Modulation Subsystem

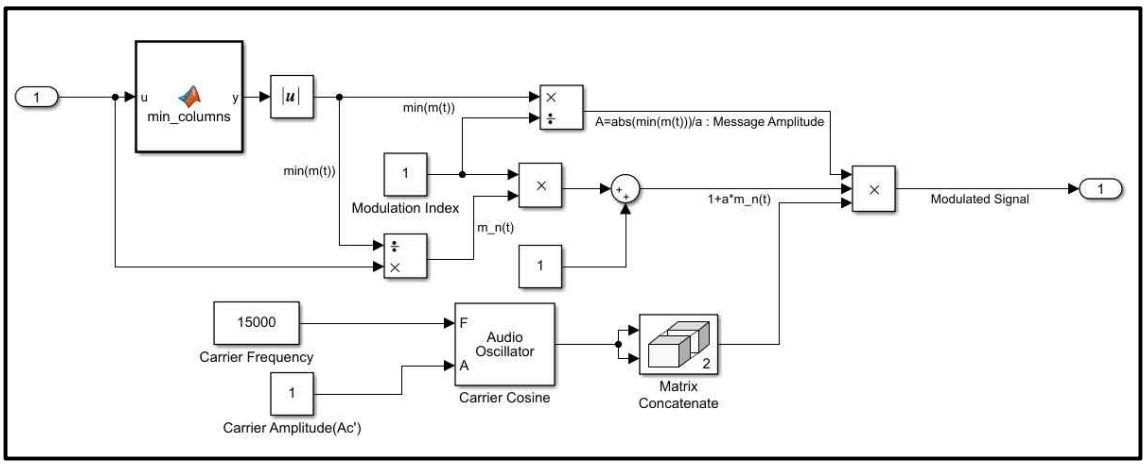


그림 6 AM modulation Subsystem 블록도

min_colums함수는 받은 signal에서 각 column의 min값을 출력하는 함수로서 min_n(t)를 구하기 위해 사용되었다. 절댓값을 통해 |min(m(t))|를 결정하였다. 블록도에서 Matrix Concatenate함수를 사용한 이유는 사용한 파일이 음원이기에 Stereo형식으로 구성되어 있어 이를 사용하기 위해 동일한 carrier signal을 복사하여 left side signal과 right side signal을 동시에 modulate하여 AM signal로 전송하였다. 사용한 Carrier frequency는 15kHz로 주로 사용하는 AM보다는 낮은 주파수 대역을 사용하였고, AM modulation에서 power를 온전하게 전달하여 복조될 수 있도록 modulation index는 1을 사용하였다.

Demodulation Subsystem

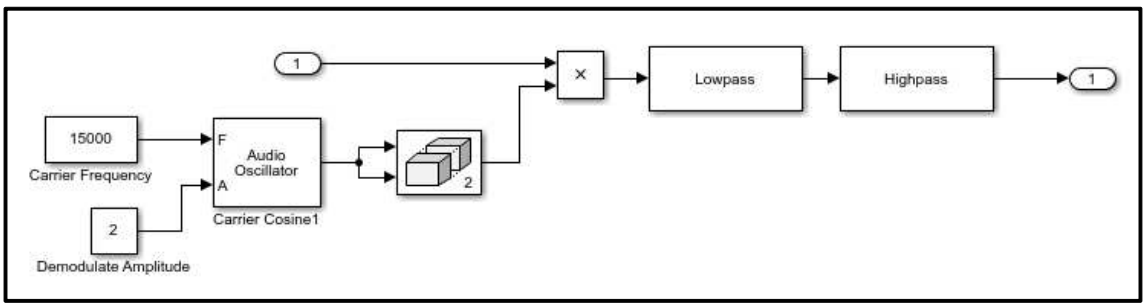


그림 7 AM demodulation Subsystem 블록도

AM의 변조에서 Noise환경을 고려하지 않았으므로, Costas phase locked loop나 Square+band pass filter는 사용하지 않고, Carrier를 곱하여 DSB 복조처럼 해결하였다. Low pass를 통해 Message+DC bias 형태로 분리해낸다. low pass filter의 pass band frequency는 1500Hz와 stop band frequency는 1600Hz를 사용하였다. 그 이유는 음원이므로 사람의 고음대역인 1500Hz이하 부분만 사용할 것이므로 이렇게 구현하였다. 이후, High Pass filter를 사용한 이유는 DC term만을 제거하는 상황은 수신자는 정확히 DC value를 모르기 때문에 pass band frequency 20Hz와 stop band frequency 50Hz를 적용하여 0Hz인 DC Tern이 원만히 제거될 수 있도록 하였다. 저역 대에서 제거되는 신호가 존재하여 이후 20Hz와 30Hz로 변환하였다.

3.3 Narrow band FM

본 프로젝트에서 구현한 전체 블록도는 다음과 같다.
Subsystem 기능을 활용하여 Modulation과 Demodulation으로 크게 분리하여 진행하였다.

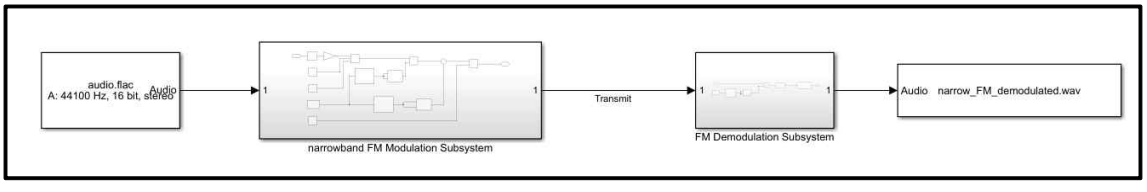


그림 8 FM Narrow band modulation과 demodulation 블록도

Narrow band modulation Subsystem

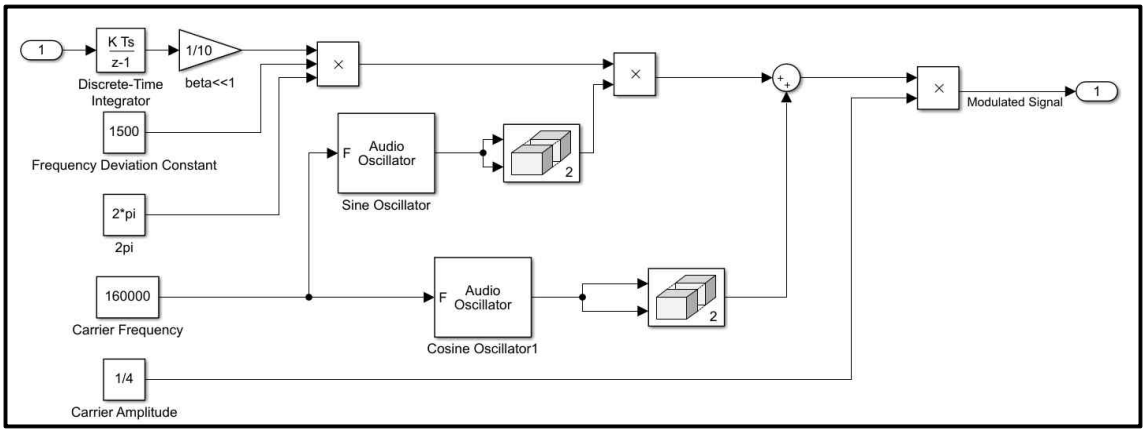


그림 9 FM Narrow band modulation Subsystem 블록도

Modulation에서 Narrow band인 상황을 가정하기 위해 modulation index인 beta 값을 1보다 작게 만들기 위해 음원의 평균 크기인 1~1.5보다 1/10으로 줄여서 power series가 성립하는 상황을 가정하였다. 이후 K_f값인 f_d값과 2*pi값을 곱하여 message signal의 pi(t)형태로 변환 시킨 후, sine 신호와 곱한 후, Carrier insertion을 통해 FM modulation처럼 변환하였다.

Narrow band demodulation Subsystem

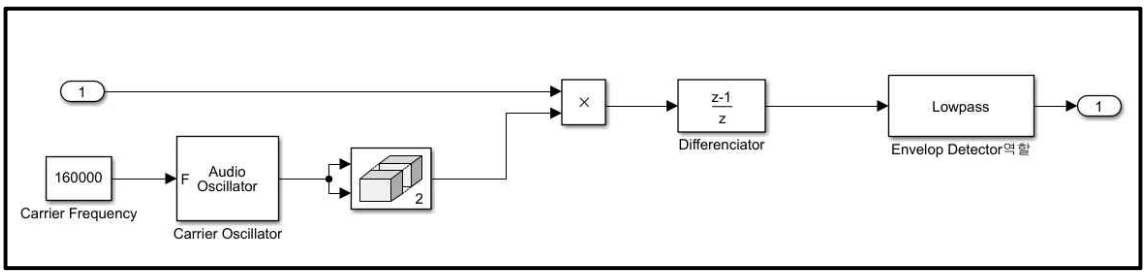


그림 10 FM Narrow band demodulation Subsystem 블록도

Demodulation에서는 discriminator방식을 사용하여 differentiator와 envelop detection기능을 구현하였다. Carrier신호를 통해 sine함수를 구현하여 곱한 후, differentiator를 통해서 amplitude에 message가 나타나도록 구현하였다. differentiator는 제공된 강의 자료의 approximation of differentiator #1부분을 활용하여 구현하였으며, Envelop detection은 AM처럼 low pass filter를 활용하여 구현하였다.

3.4 wideband FM

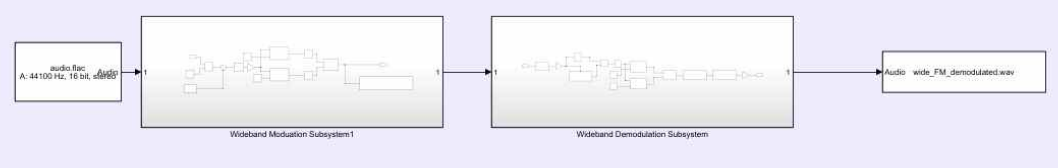
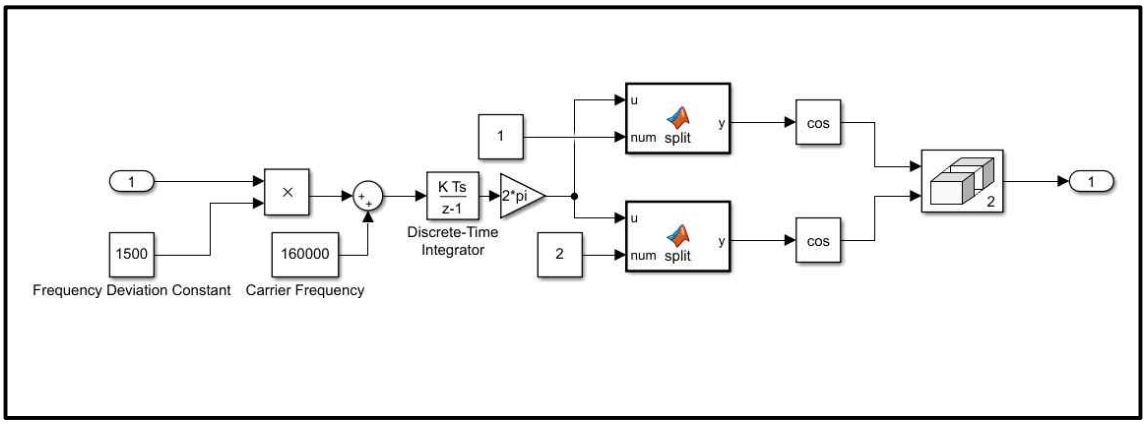


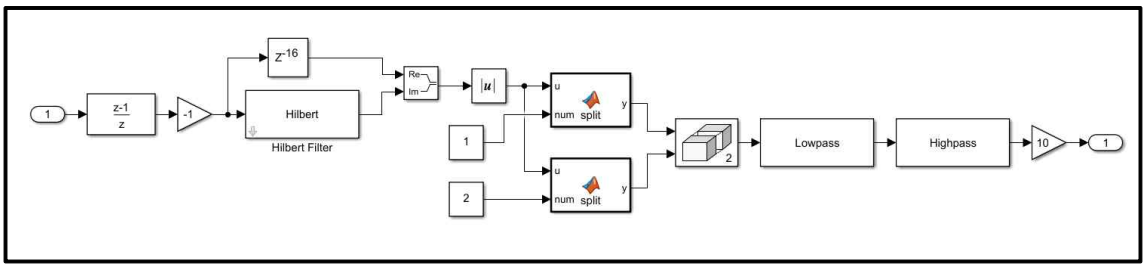
그림 11 FM Wideband 전체 블록도

Wide band modulation Subsystem



Modulation에서 Wideband 상황을 가정하여 진행하기 위해 기존의 누적합을 이용한 cosine carrier에 담아 전송하는 형태의 신호를 만들기 위해 그림 11과 같이 구성하였다. wideband 에서는 신호가 narrow band와 달리 carrier신호에 직접 곱하지 않고 신호를 실어서 보낼 수 있다. Power series로 approximation하지 않고 보내므로 위와 같이 좀 더 단순한 형태로 구성 가능하다. Carrier frequency를 누적 합에 넣은 이유는 적분시에 t이 곱해지는 형태를 유지하기 때문에 넣고 동시에 적분하였다. 여기서 split 함수는 stereo신호를 연산에 용이하게 나누는 함수로서 input값을 index값을 받아서 사용한다.

Wide band demodulation Subsystem



Demodulation에서는 Discriminator상황을 부여하여 진행하였다. Matlab에서는 envelop detection에서 두 가지 방식으로 구현이 가능하다. 성능이 낮지만 구현하기 쉬운 1번 방식과 성능이 우수하지만 개념이 어려운 Hilbert Transform 방식이 존재한다. 이에 후자 방식인 Hillbert Transform을 사용하였다. 이 방식은 Hilbert Transform을 통해 32차로 complex 항을 제시하고 Hilbert Transform에서 delay되는 항만큼 real항을 16time을 delay시켰다. 이를 phasor항으로 변환하여 연산하는 방식으로 과정이 복잡하지만, 성능이 우수하다는 특징이 있다.

4. Code

※주의사항
Text 형식의 코드는 가독성이 매우 떨어집니다. 이미지로 첨부하려 했으나 원활한 설명을 위해 보고서에는 텍스트로 기재하며, 실제 코드 파일을 별도로 첨부하니 원리 설명 용도로만 확인해 주세요. 설명을 삽입하였기 때문에, 복사하여 구동할 수 없습니다.

%% Load audio file

```
[OriginalAudio,Fs] = audioread('audio.flac');
```

Audioread 함수를 이용해 음성 파일을 행렬 Fs로 옮긴다

```
% Sampling Frequency와 Bit Resolution을 찾기 위한 audioinfo 함수
info = audioinfo('audio.flac');
```

음성 파일의 정보를 제공해주는 함수

```
% Write the original audio file in assigned format before Transmission
audiowrite('beforeTransmissionFile.wav',OriginalAudio,44100);
```

원 신호를 음성 파일로 저장한다.

```
%% Parameters            //각종 parameter
```

%Audioinfo 함수를 통해 얻은 값

```
SamplingFrequency = 44100;
BitResolution = 16;
%Modulation / Demodulation Parameter
CarrierFrequency = 10000;
CarrierAmplitude = 0;
CarrierAmplitudeDemod = 0;
InitialPhaseForModulation = 0;
FrequencyDeviation = 5000;

%% 모노와 스테레오 구분(큰 의미 없음)
wid = size(OriginalAudio,1);
if(wid ==1)
    OriginalAudio = OriginalAudio(:);
end
```

Audio가 Stereo일 시 행렬은 2열이 되며, Mono일 시 1열이다.

```
%% 시간축 생성
t = (0:1/SamplingFrequency:((size(OriginalAudio, 1)-1)/SamplingFrequency))';
t = t(:, ones(1, size(OriginalAudio, 2)));
```

Audio의 길이와 같은 시간만큼 시간행렬 T를 생성한다.

%% DSB

%Modulation

ModulatedDSBSignal = (OriginalAudio) .* cos(2 * pi * CarrierFrequency * t + InitialPhaseForModulation);

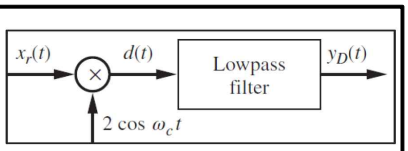
$$x_c(t) = A(t)\cos\omega_c t \qquad (\phi(t) \equiv 0)$$
$$= A_c m(t)\cos\omega_c t$$
$$\triangleq \text{message} \times \text{carrier}$$

% Demodulation (Carrier를 곱하여 (위상이 같다고 가정) 복조한다.)

DemodulatedDSBSignal = ModulatedDSBSignal .* cos(2*pi * CarrierFrequency * t + InitialPhaseForModulation);

Demodulation

$$x_c(t) \times 2\cos\omega_c t = A_c m(t) + A_c m(t)\cos 2\omega_c t \Rightarrow \text{LPF}$$
$$\triangleq d(t)$$



- Problem: Demodulation needs exact value of ω_c and phase coherency
  (≡ "coherent" demodulation, 동기 복조)

Ex. $d(t) = x_c(t) \times 2\cos(\omega_c t + \theta(t))$ → 위상 오차

실제로는 위상오차가 발생하여 Phase detector를 통해 바로잡아야 하지만, Code에서는 이론적인 부분을 확인하고, Simulink를 통해 AM과 FM을 중심적으로 다룬다.

% Low-pass Filter (Carrier을 곱할 경우 두개의 Sideband가 생기므로 Baseband의 반대쪽을 없앤다. 아래 그림 참조)

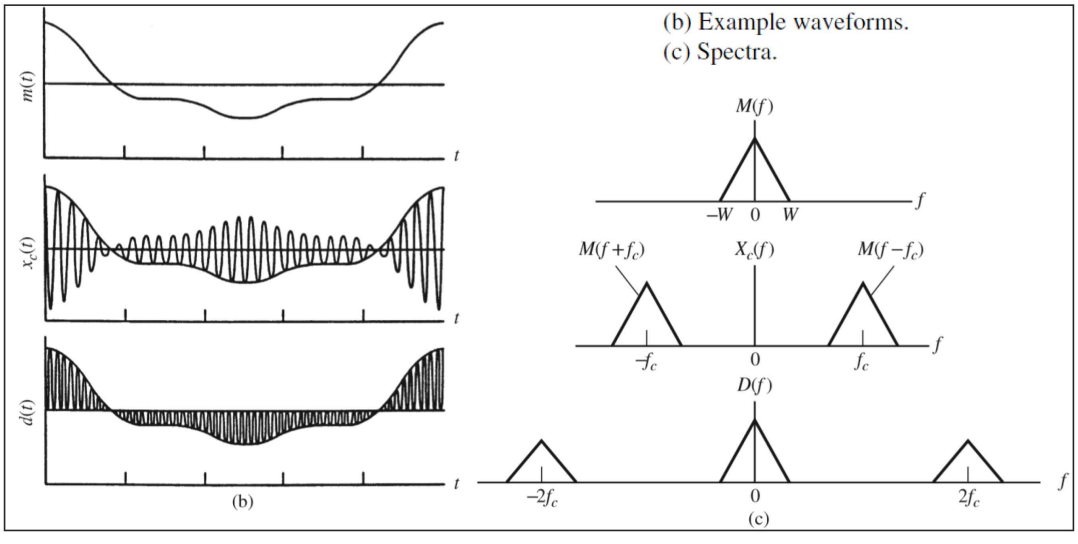
DemodulatedDSBSignal = lowpass(DemodulatedDSBSignal,10000,SamplingFrequency);

%신호 정규화 Audio 저장을 위함

DemodulatedDSBSignalNorm = DemodulatedDSBSignal./(max(abs(DemodulatedDSBSignal)));

% Audio 출력

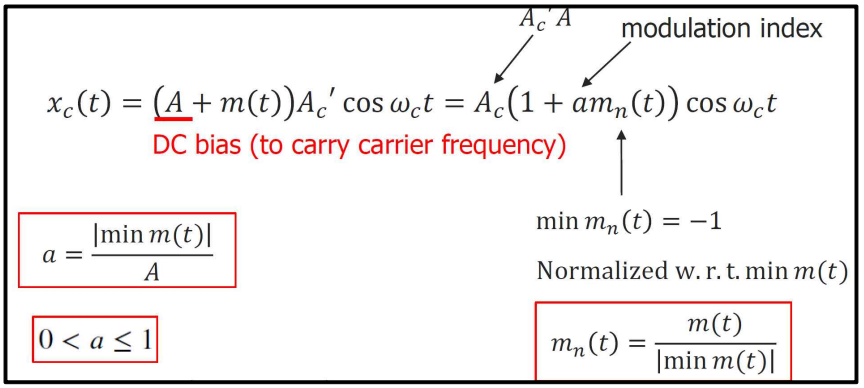
audiowrite('receviedDSBSoundFile.wav',DemodulatedDSBSignalNorm,SamplingFrequency);



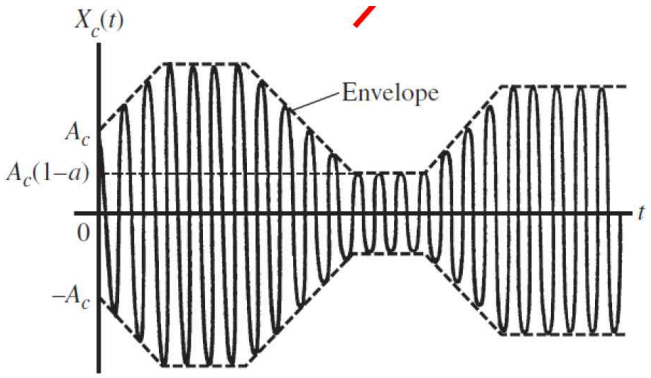
%% AM

%정규화를 위한 DC Bias 값을 찾는다.
AMBias = abs(min(OriginalAudio)); **그림 하단의 수식**

%Modulation
ModulatedAMSignal = (OriginalAudio + AMBias) .* cos(2 * pi * CarrierFrequency * t + InitialPhaseForModulation);



%Demodulation (Envelope Detector with Hilbert Transform)
DemodulatedAMSignal = abs(hilbert(ModulatedAMSignal).*exp(-1i*2*pi*CarrierFrequency*t));



%Demodulation 신호 정규화 Audio 출력을 위함
DemodulatedAMSignalNorm = DemodulatedAMSignal./(max(abs(DemodulatedAMSignal(50000:500000,1:2))));

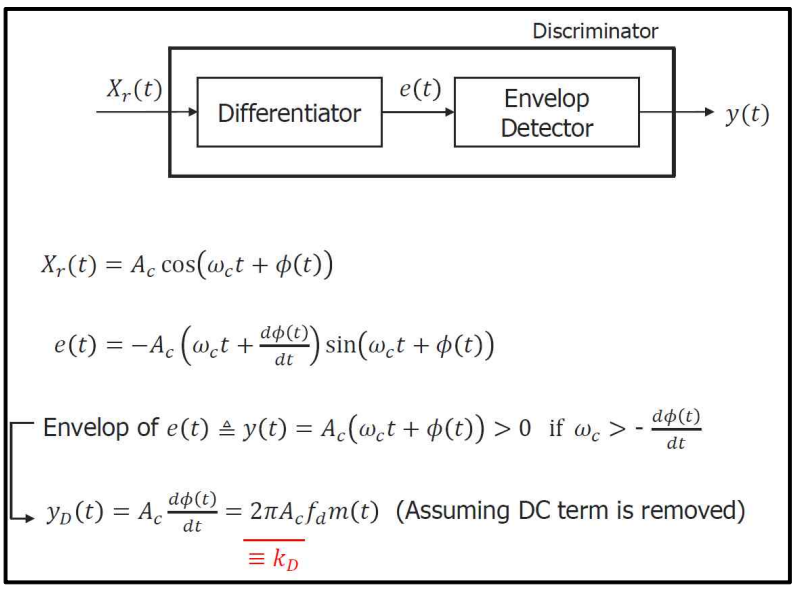
% Audio 출력
audiowrite('receviedAmSoundFile.wav',DemodulatedAMSignalNorm,SamplingFrequency);

%% FM (Wideband)

%modulation
int_x = cumsum(OriginalAudio)/SamplingFrequency;
%% m(t)를 적분하여 위상 함수를 만든다.
ModulatedFMSignal = cos(2*pi*CarrierFrequency*t + 2*pi*FrequencyDeviation*int_x + InitialPhaseForModulation); %%FM signal의 정의

- FM: $x_c(t) = A_c \cos\left(\omega_c t + \underbrace{2\pi f_d \int^t m(\alpha)d\alpha}_{\phi(t)}\right)$

% demodulation (Hilbert 변환과 미분을 이용한 Demodulation)
yq = hilbert(ModulatedFMSignal).*exp(-1i*2*pi*CarrierFrequency*t);
DemodulatedFMSignal = (1/(2*pi*FrequencyDeviation))*[zeros(1,size(yq,2));
diff(unwrap(angle(yq)))*SamplingFrequency];



%%신호 정규화

DemodulatedFMSignalNorm =
DemodulatedFMSignal./(max(abs(DemodulatedFMSignal(50000:500000,1:2))));
50000:500000값의 이유는, Signal 행렬의 양 끝 2개의 샘플은 그 절댓값이 매우 크다. 이는 오디오 신호가 Analog이지만, 컴퓨터에서 연산 시 Sample된 값으로 미분하므로, 연산을 위한 연속적인 정보가 충분히 확보되지 않기 때문이다. 따라서 이 값들을 제외하기 위해 샘플의 일부를 택하였다.

```
%Audio out
audiowrite('receviedFmSoundFile.wav',DemodulatedFMSignalNorm,SamplingFrequency);


%% Noise channel for AM and FM
```
Channel에 Noise를 추가하여 비교한다. 설명한 부분을 제외하고는 위의 AM과 FM 코드와 동일하다.

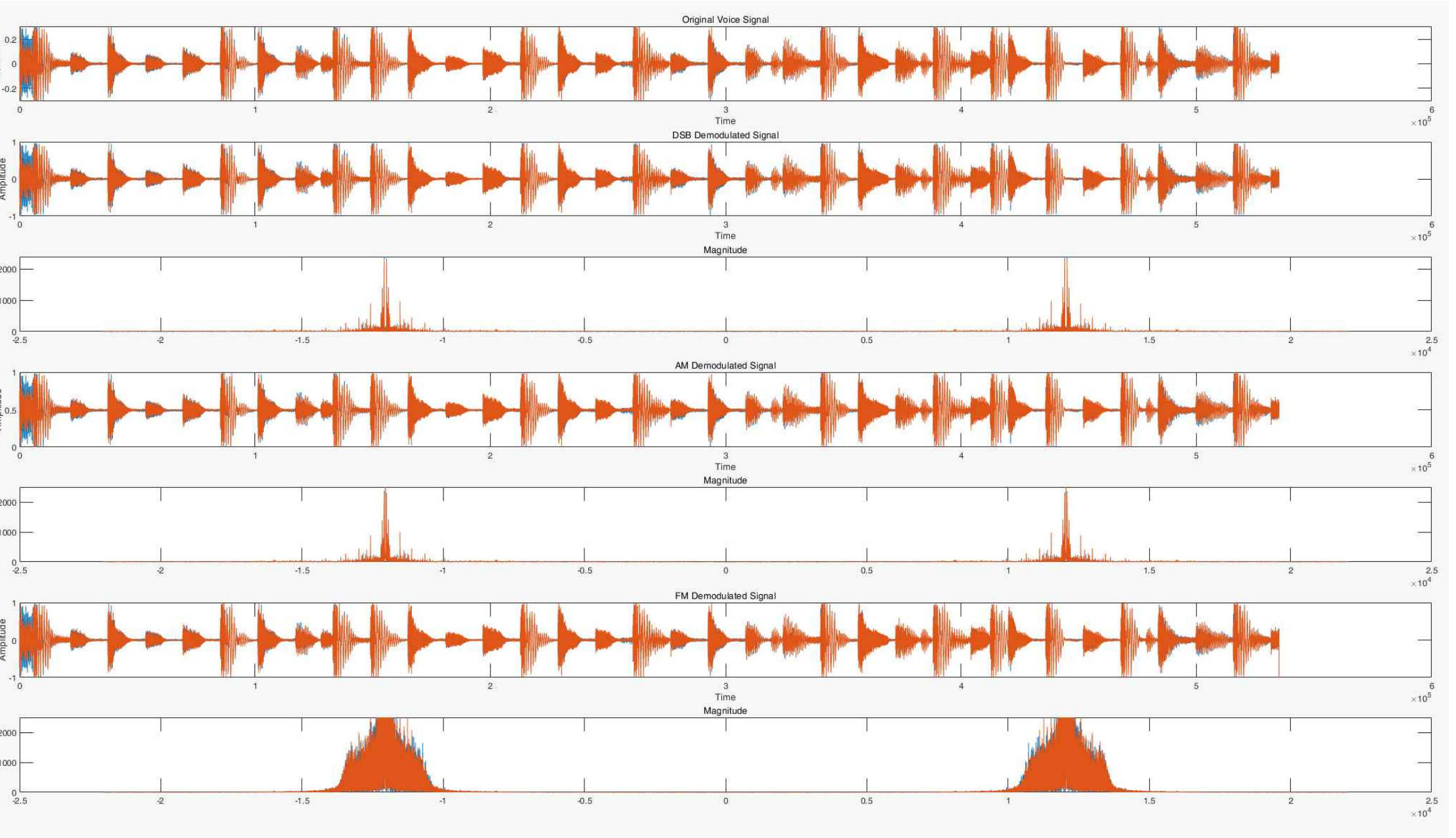
```
%Modulation
ModulatedAMSignalNoise = awgn(ModulatedAMSignal,40);
```
Noise를 추가하여 값을 반환하는 AWGN 함수를 이용하였다. 40의 값은 SNR로, dB단위이다.
```
%Demodulation (Envelope Detector with Hilbert Transform)
DemodulatedAMSignalNoise =
abs(hilbert(ModulatedAMSignalNoise).*exp(-1i*2*pi*CarrierFrequency*t));
%Demodulation 신호 정규화
DemodulatedAMSignalNormNoise =
lowpass(DemodulatedAMSignalNoise./(max(abs(DemodulatedAMSignalNoise(50000:500000,1
:2)))),5000,SamplingFrequency);
% Audio 출력
audiowrite('receviedAmSoundFileNoise.wav',DemodulatedAMSignalNormNoise,SamplingFreq
uency);


%modulation
ModulatedFMSignalNoise = awgn(ModulatedFMSignal,40);
% demodulation (Hilbert 변환을 이용한 Demodulation)
yqNoise = hilbert(ModulatedFMSignalNoise).*exp(-1i*2*pi*CarrierFrequency*t);
DemodulatedFMSignalNoise =
lowpass((1/(2*pi*FrequencyDeviation))*[zeros(1,size(yqNoise,2));
diff(unwrap(angle(yqNoise)))*SamplingFrequency],5000,SamplingFrequency);
%%신호 정규화
DemodulatedFMSignalNormNoise =
DemodulatedFMSignalNoise./(max(abs(DemodulatedFMSignalNoise(50000:500000,1:2))));
%Audio out
audiowrite('receviedFmSoundFileNoise.wav',DemodulatedFMSignalNormNoise,SamplingFrequ
ency);
```

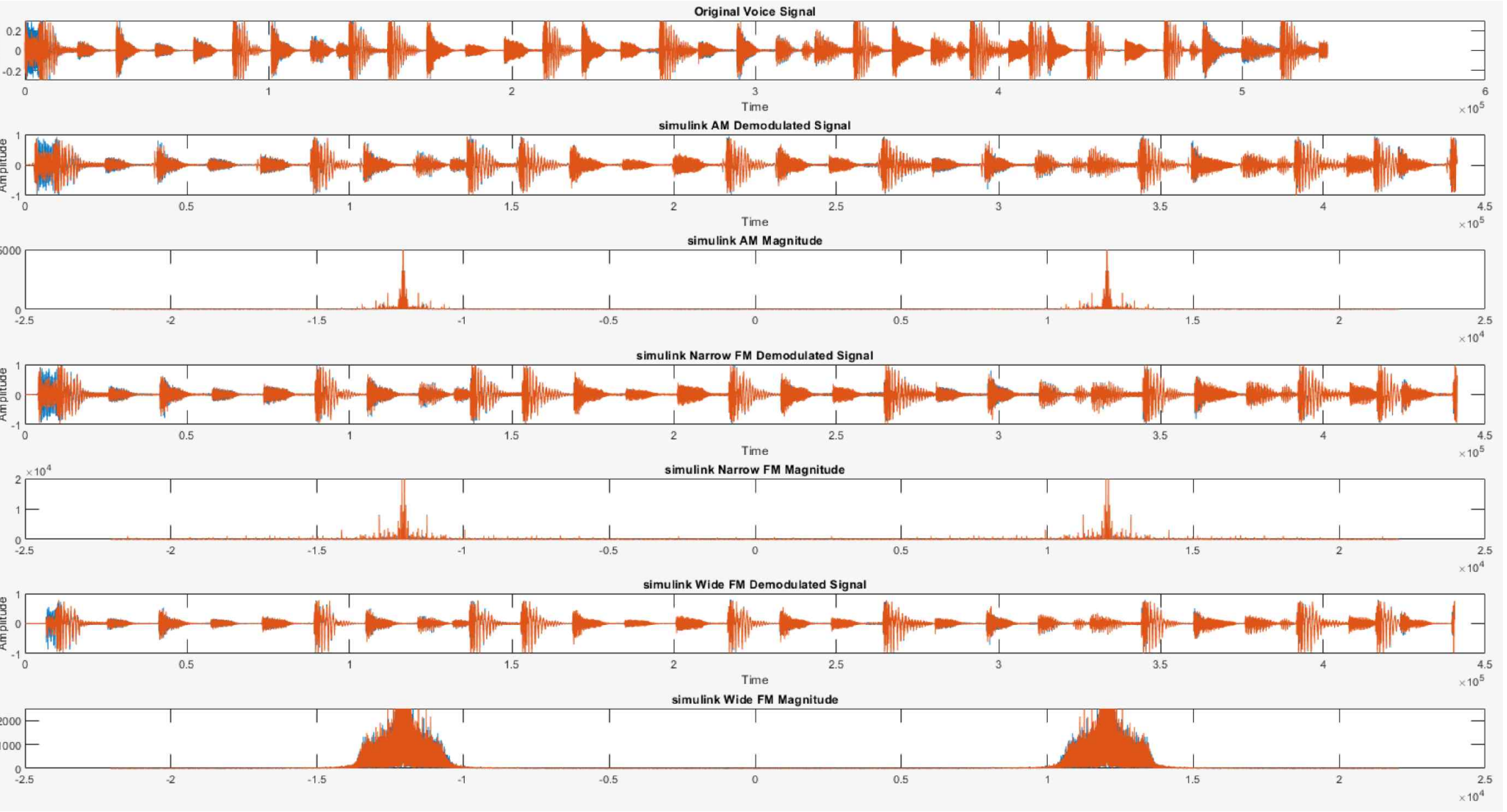
%% 뒤의 plot 파트는 결과보고서에서 생략하였다.

5. 결과

5.1 Wave(Code)



5.2 Wave(Simulink)



5.3 설명

5.1의 Wave 파형을 살펴보자. 총 7줄이며, 맨 위 줄부터 시작하여 각각

1. 원 신호의 파형

2. 원 신호 - DSB modulation - demodulation을 거친 파형
3. DSB modulated Signal의 Spectrum

4. 원 신호 - AM modulation - demodulation을 거친 파형
5. AM modulated Signal의 Spectrum

6. 원 신호 - wideband FM modulation - demodulation을 거친 파형
7. wideband FM modulated Signal의 Spectrum

이다. 5.2의 Wave 파형을 살펴보자. 맨 위 줄부터 시작하여

1. 원 신호의 파형

2. AM system을 거친 파형
3. AM modulated Signal의 Spectrum

4. narrowband FM system을 거친 파형
5. narrowband FM modulated Signal의 Spectrum

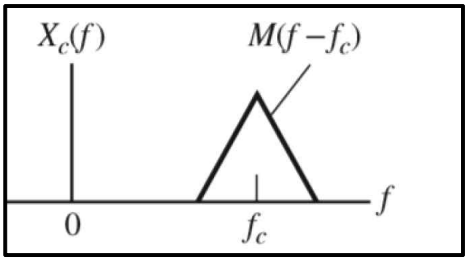
6. wideband FM system을 거친 파형
7. wideband FM modulated Signal의 Spectrum

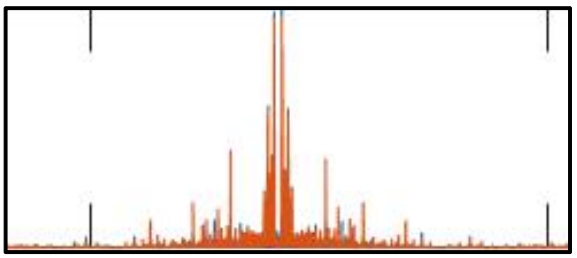
이다.

전체적으로 파형은 잘 보존되었으며, 결과물인 음성신호 또한 UCC에서 볼 수 있다시피 원래의 신호 (음성 파일)과 수신 후 복조된 음성 파일이 충분히 유사하다.

파형에 대한 설명은 생략하고, 이제 Spectrum을 분석하며 이론과 비교해보자.

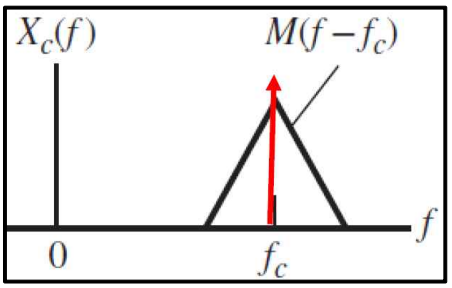
DSB Spectrum (Code)

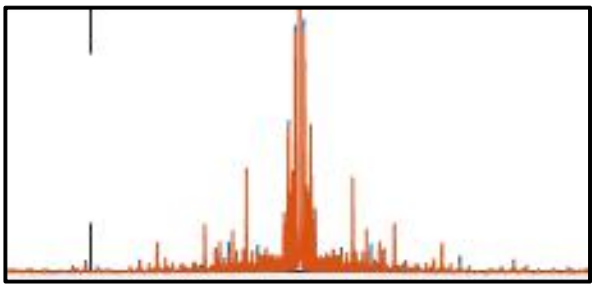




Baseband Spectrum이 주파주축 상에서 Fc(Carrier Frequency)만큼 옮겨진 것이 확인된다.

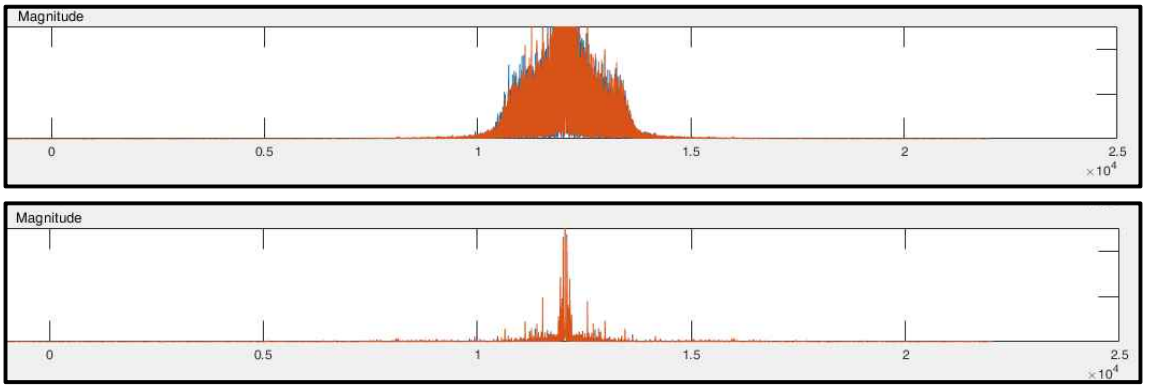
AM Spectrum (Code)





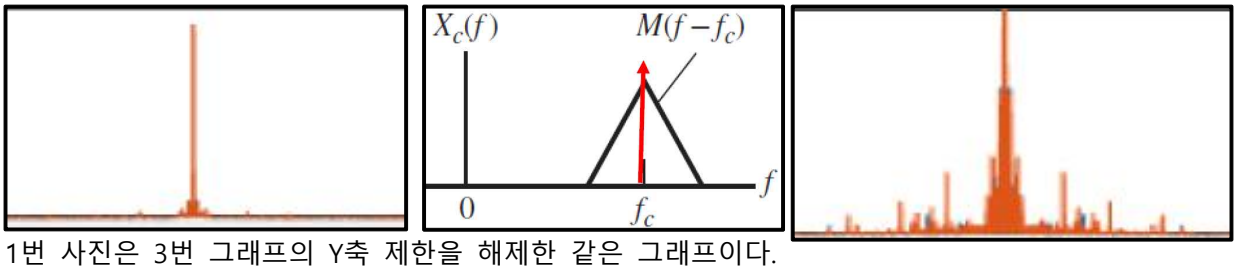
DSB와 유사하지만, AM에서는 Carrier Frequency가 더해지므로, 주파수축의 Fc상에 Impulse가 발견된다. 현재 그림은 Magnitude 축을 잘라내어 눈에 띄지 않지만, Impulse가 발견되며, 뒤의 Simulink 파형에서 명확히 보여 줄 예정이다.

Wideband FM Spectum (Code)



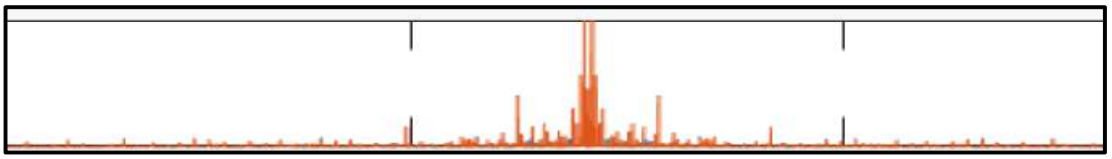
아래 그림(Amplitude Modulation)에 비해 Wideband FM의 Bandwidth가 훨씬 큰 것을 확인할 수 있었다.

AM Spectrum (Simulink)



1번 사진은 3번 그래프의 Y축 제한을 해제한 같은 그래프이다.
이 실험의 핵심은 2번 그림(강의자료)의 Carrier Frequency 가 Spectrum에 나타나는 것이다. Code의 Spectrum과 같다.

Narrowband FM Spectum (Simulink)



Narrowband FM Spectrum은 유독 잔가지가 많았다. 그 이유를 추측하기로는, Narrowband FM Modulation의 정의에 있다. 정의에 따르면,

Narrowband angle modulation

$$x_c(t) = A_c \cos\big(\omega_c t + \phi(t)\big) = \mathrm{Re}\big(A_c e^{j\omega_c t} e^{j\phi(t)}\big)$$

power series expansion →

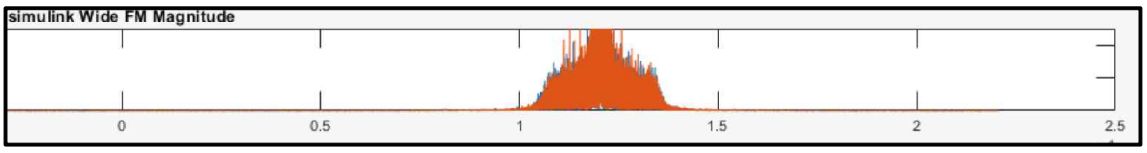
$$\approx \mathrm{Re}\left\{A_c e^{j\omega_c t}\left[1 + j\phi(t) - \frac{\phi^2(t)}{2!} + \cdots\right]\right\} \approx \mathrm{Re}\big(A_c e^{j\omega_c t} + A_c \phi(t) j e^{j\omega_c t}\big)$$

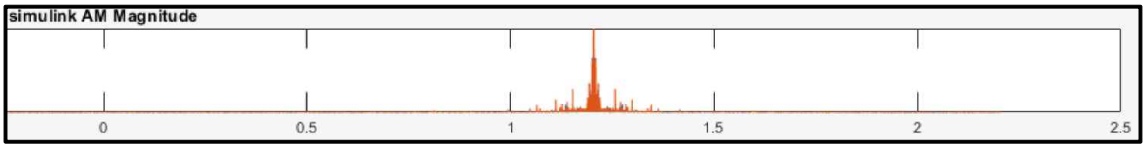
$|\phi(t)| \ll 1$

$$x_c(t) \approx A_c \cos\omega_c t - A_c \phi(t) \sin\omega_c t$$

Modulation에서 가정된 Power Series expansion에서 $|\Phi(t)|$가 1보다 아주 작음을 가정하여 2차항 이후를 생략하는데, 여기서 생긴 Distortion 현상으로 추측된다.
전체적인 Spectrum은 이론과 비슷하게 출력되었다.

Wideband FM Spectum (Code)





아래 그림(Amplitude Modulation)에 비해 Wideband FM의 Bandwidth가 훨씬 큰 것을 확인할 수 있었다.

6. 결론

우리 프로젝트만의 강조하고 싶은 부분은 다음과 같다.

1. 오디오 신호를 Source로 사용하였다.
정현파로 구현하는 일반적인 프로젝트와 달리, 우리는 실제 정보의 특성을 지닌 오디오 신호를 이용하였다. 음성 신호와 같은 실제 정보는 불확실성을 가지고 있으며, 다양한 주파수 성분이 있다. Matlab 내의 Audio Toolbox를 추가로 설치하여 이를 구현하였다. 이는 Spectrum 분석 시 삼각형 형태(실제 통신시스템과 유사한 Baseband Signal 주파수 대역)로 확인되며, 이는 정현파를 사용한 경우와는 달리, AM과 같이 Carrier 주파수에 Impulse가 존재하는 경우를 명확히 확인 할 수 있다.

2. Simulink를 사용하여 코드와 비교 검증 하였다.
강의에서 학습한 이론을 토대로 직접 Block을 설계 해보며, Matlab 시뮬레이션 역량을 키웠다. Subsystem을 활용하였는데, Subsystem은 가독성을 높힐 뿐 아니라 앞으로 Simulink를 현장에서 사용할 경우 팀원/팀 간 소통을 원활하게 하는 기본적인 테크닉이라고 생각된다. Simulink를 사용하여 탐구하는 시간이 프로젝트에서 큰 비중을 차지했으며, 좋은 학습이 되었고 결과물 또한 만족스러웠다.

3. 다양한 Modulation/Demodulation 방식을 구현
이번 학기의 전반적인 내용을 모두 다루며, Noise Channel과 Discriminator(Differentiator, Envelop detector), Filter(DC Value제거) 등 강의에서 볼 수 있었던 이론들을 다양한 Coding Style / Simulink Block으로 구현하였다.

반대로, 아쉬운 점들은 다음과 같다.

1. 완성하지 못한 PLL Block
제안서에서 정한 주제를 최대한 완성해보려 했으나, VCO 블록에 막혀 끝내 완성하지 못했다.

2. AM과 FM 비교에서 가설을 검증하지 못함
프로젝트를 시작하기 전 FM이 AM에 비해 노이즈에 강하다는 정보를 찾았고, 그에 대한 검증을 하는 것을 프로젝트의 목표로 삼았다. 제안서에서 계획했던 주제는 PLL을 이용한 FM System을 주로 다루는 것이었다. 하지만 Phase Locked loop를 구현하지 못했다. PLL에 필요한 핵심 Block인 VCO를 구현하지 못했기 때문이며, 따라서 계획을 변경하였다. 때문에 동일 위상 demodulation을 가정할 수밖에 없었으며 Noise 채널 비교가 큰 의미를 갖지 못하게 되었다.

7. 작업일지

~10월 26일 : 1차 회의 후 제안서 제출

~11월 4일: AM Simulink Block Diagram 제작 완료, AM과 DSB Matlab Code 구현 완료

~11월 6일 Filter, Stereo Divider 등 프로젝트에 기반이 되는 다양한 기능들을 Simulink에서 Module로 구현, Audio 입력 신호 Load 및 Store 하는 Function 작동 확인

~11월 9일: 신호의 단순 입출력 여부 확인 및 GUI(Simulink에서 지원하는 유저 인터페이스) 구현 여부 결정 (부결됨)

~11월 11일: Debugging Tool 제작 Spectrum 기반 modulation Debugging 확인

~11월 17일: FM 알고리즘 중간고사 이후 제작 결정(가결)

~11월 20일: 중간 결과 보고서 작성 완료

~11월 23일 중간 결과 보고서 제출 일자(알고리즘 위주 구현 내용)
-> Feedback: Simulink 지속 사용 및 함수 세분화

~11월 24일: FM narrowband signal 기반 변조 및 복조 Simulink 구현 완료

~11월 25일: FM wideband signal 기반 변조 및 복조 Matlab 구현 완료

~11월 28일: FM wideband signal 기반 변조 및 복조 Simulink 구현 완료
-> Feedback: 동일위상 구현은 성공했으나, 위상이 다를 경우 PLL을 이용해야 하는데 구현에 실패. 주제를 변경하였으며, Noise채널을 구현할지 여부를 결정해야 함.

~11월 30일: 노이즈 제작 여부 결정(가결, Matlab code한정)

~11월 31일: Matlab에서 Noise가 포함된 FM, AM 변조 복조 구현 완료

~12월 2일: FM과 AM 스펙트럼 비교 그래프 제작 완료

~12월 20일: UCC제작 및 결과보고서 작성 완료

~12월 21일 : 최종 결과 보고서, 소스코드 공개 및 작품 동작 설명하는 UCC 제출