MACHİNE LEARNİNG FOR SİGNAL PROCESSİNG – EHB-328E

COMPUTER HOMEWORK-3
SİGNAL SEPERATİON WİTH NMF

Prof. Dr. Ender Mete Ekşioğlu

Osman Kaan Kurtça

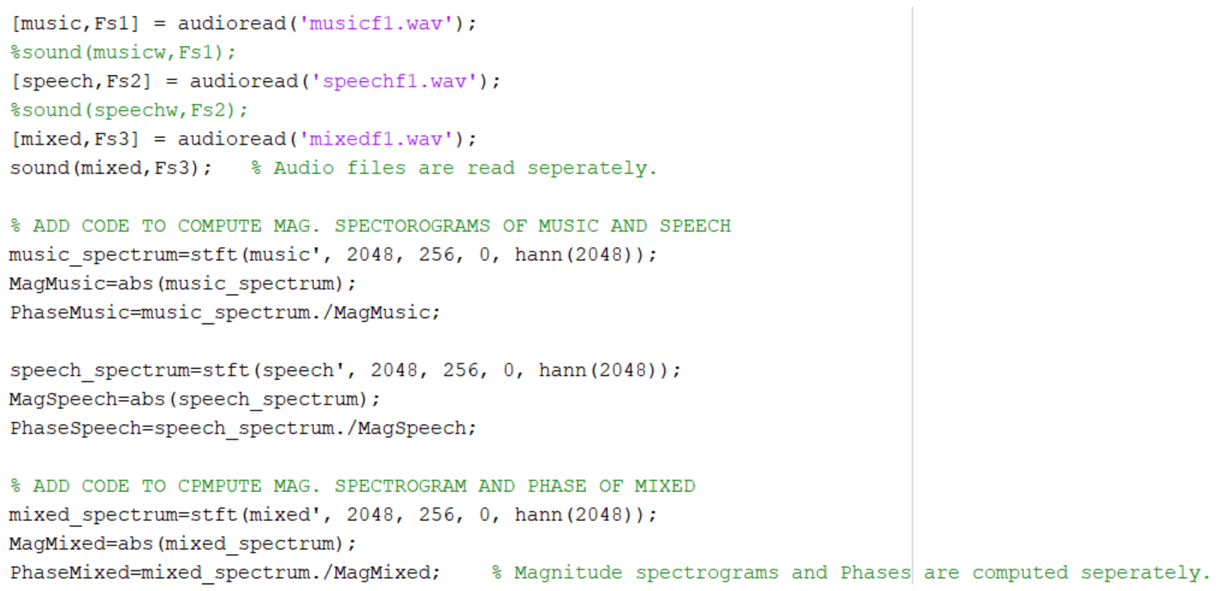
040160090

Aziz Gündoğdu

040160112

In this homework, our goal is to separate music and speech sounds from a signal consisting of both using Non-Negative Matrix Factorization(NMF).

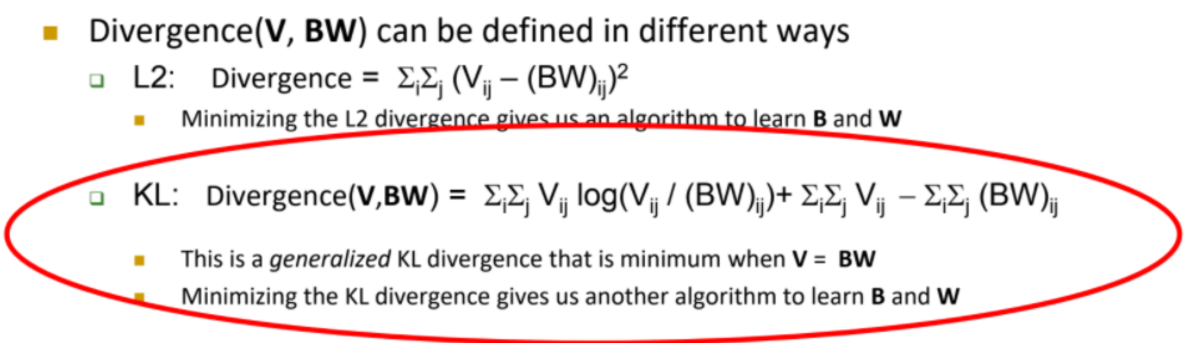
Firstly, we read all audio files separately and take their STFT (Short-time Fourier transform) to work in the frequency domain. After that, Magnitude and phase components are computed.



```
[music,Fs1] = audioread('musicf1.wav');
%sound(musicw,Fs1);
[speech,Fs2] = audioread('speechf1.wav');
%sound(speechw,Fs2);
[mixed,Fs3] = audioread('mixedf1.wav');
sound(mixed,Fs3);    % Audio files are read seperately.

% ADD CODE TO COMPUTE MAG. SPECTOROGRAMS OF MUSIC AND SPEECH
music_spectrum=stft(music', 2048, 256, 0, hann(2048));
MagMusic=abs(music_spectrum);
PhaseMusic=music_spectrum./MagMusic;

speech_spectrum=stft(speech', 2048, 256, 0, hann(2048));
MagSpeech=abs(speech_spectrum);
PhaseSpeech=speech_spectrum./MagSpeech;

% ADD CODE TO CPMPUTE MAG. SPECTROGRAM AND PHASE OF MIXED
mixed_spectrum=stft(mixed', 2048, 256, 0, hann(2048));
MagMixed=abs(mixed_spectrum);
PhaseMixed=mixed_spectrum./MagMixed;    % Magnitude spectrograms and Phases are computed seperately.
```

Figure 1: Audio files are read and computed their spectrogram.

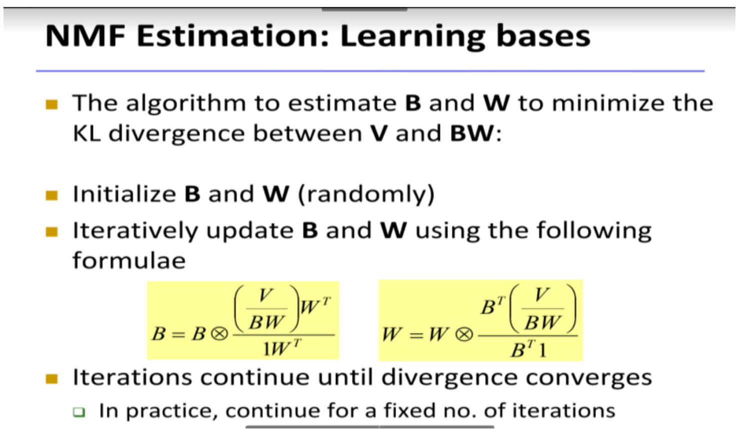
Secondly, we must learn bases for speech and music using NMF algorithm with the least error.



- Divergence(**V**, **BW**) can be defined in different ways
  - L2: Divergence = $\Sigma_i\Sigma_j (V_{ij} - (BW)_{ij})^2$
    - Minimizing the L2 divergence gives us an algorithm to learn **B** and **W**
  - KL: Divergence(**V**,**BW**) = $\Sigma_i\Sigma_j V_{ij} \log(V_{ij} / (BW)_{ij}) + \Sigma_i\Sigma_j V_{ij} - \Sigma_i\Sigma_j (BW)_{ij}$
    - This is a *generalized* KL divergence that is minimum when **V** = **BW**
    - Minimizing the KL divergence gives us another algorithm to learn **B** and **W**

Figure 2: Error minimizing optimization problem.

To minimize error, we can use L2 or KL divergence. In this problem, we choose KL divergence because NMF based representation work best when we minimize the KL divergence for many signals.



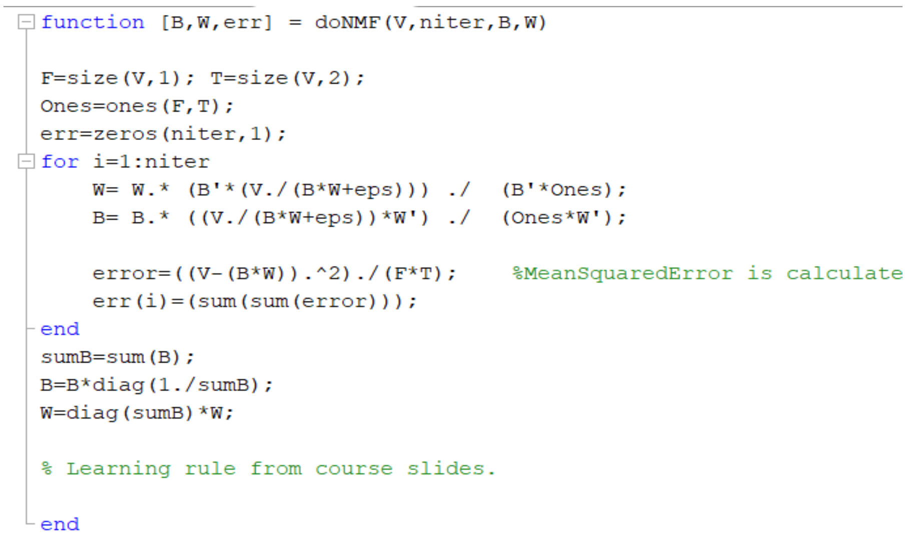
NMF Estimation: Learning bases

- The algorithm to estimate **B** and **W** to minimize the KL divergence between **V** and **BW**:
- Initialize **B** and **W** (randomly)
- Iteratively update **B** and **W** using the following formulae

$$B = B \otimes \frac{\left(\frac{V}{BW}\right)W^T}{1W^T} \qquad W = W \otimes \frac{B^T\left(\frac{V}{BW}\right)}{B^T 1}$$

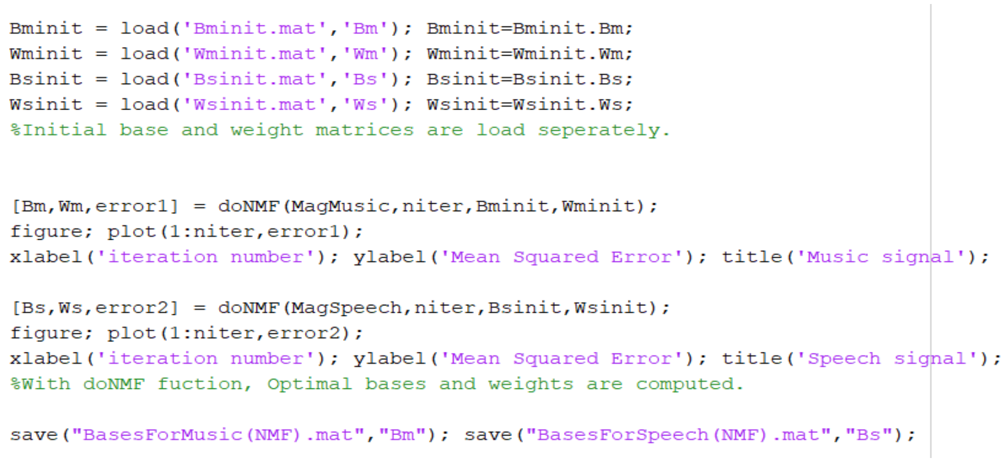
- Iterations continue until divergence converges
  - In practice, continue for a fixed no. of iterations

Figure 3: Learning rule for basis and weights.



```
function [B,W,err] = doNMF(V,niter,B,W)

F=size(V,1); T=size(V,2);
Ones=ones(F,T);
err=zeros(niter,1);
for i=1:niter
    W= W.* (B'*(V./(B*W+eps))) ./  (B'*Ones);
    B= B.* ((V./(B*W+eps))*W') ./  (Ones*W');

    error=((V-(B*W)).^2)./(F*T);    %MeanSquaredError is calculate
    err(i)=(sum(sum(error)));
end
sumB=sum(B);
B=B*diag(1./sumB);
W=diag(sumB)*W;

% Learning rule from course slides.

end
```

Figure 4: NMF Function.



```
Bminit = load('Bminit.mat','Bm'); Bminit=Bminit.Bm;
Wminit = load('Wminit.mat','Wm'); Wminit=Wminit.Wm;
Bsinit = load('Bsinit.mat','Bs'); Bsinit=Bsinit.Bs;
Wsinit = load('Wsinit.mat','Ws'); Wsinit=Wsinit.Ws;
%Initial base and weight matrices are load seperately.


[Bm,Wm,error1] = doNMF(MagMusic,niter,Bminit,Wminit);
figure; plot(1:niter,error1);
xlabel('iteration number'); ylabel('Mean Squared Error'); title('Music signal');

[Bs,Ws,error2] = doNMF(MagSpeech,niter,Bsinit,Wsinit);
figure; plot(1:niter,error2);
xlabel('iteration number'); ylabel('Mean Squared Error'); title('Speech signal');
%With doNMF fuction, Optimal bases and weights are computed.

save("BasesForMusic(NMF).mat","Bm"); save("BasesForSpeech(NMF).mat","Bs");
```

Figure 5: Initial matrices are loaded and optimal bases and weights computed.

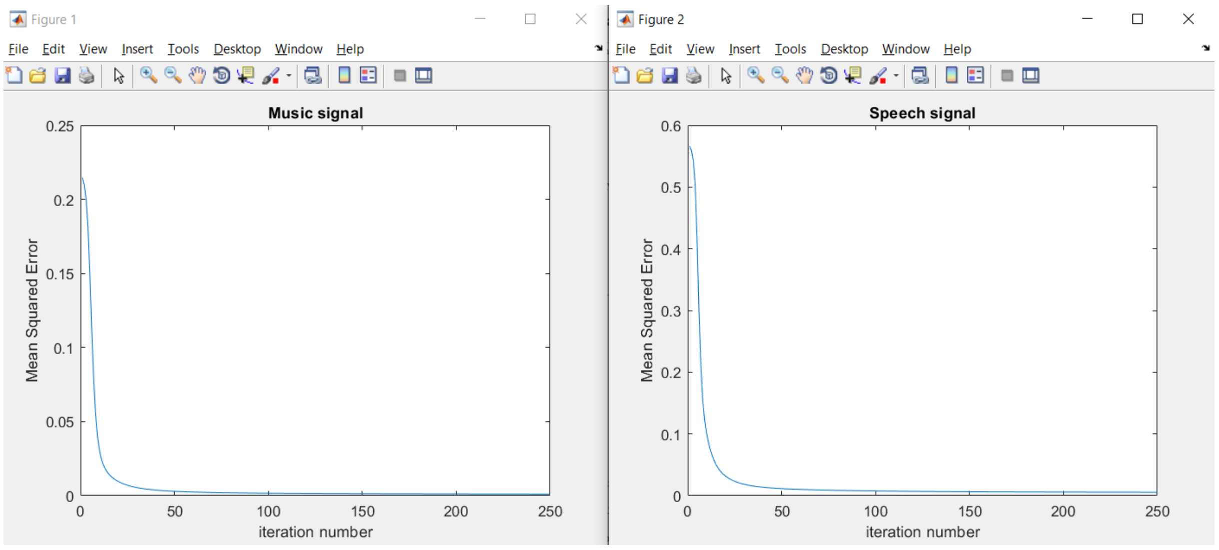


Figure 6: Graphs showing the error according to the number of iterations. $\Sigma_i\Sigma_j (V_{ij} - (BW)_{ij})^2$

After calculating the optimum base matrices, with the signal separation function, music and speech sounds are obtained from the mixed signal.

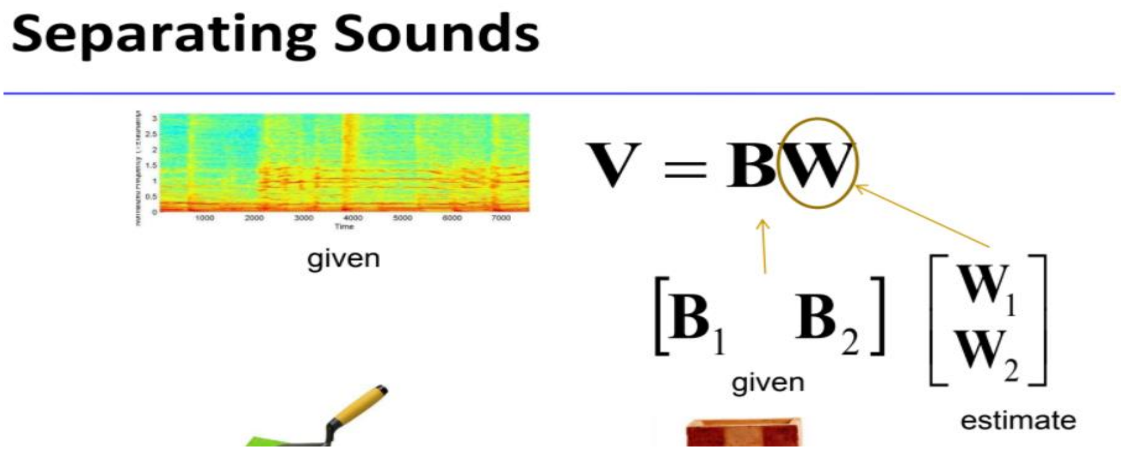
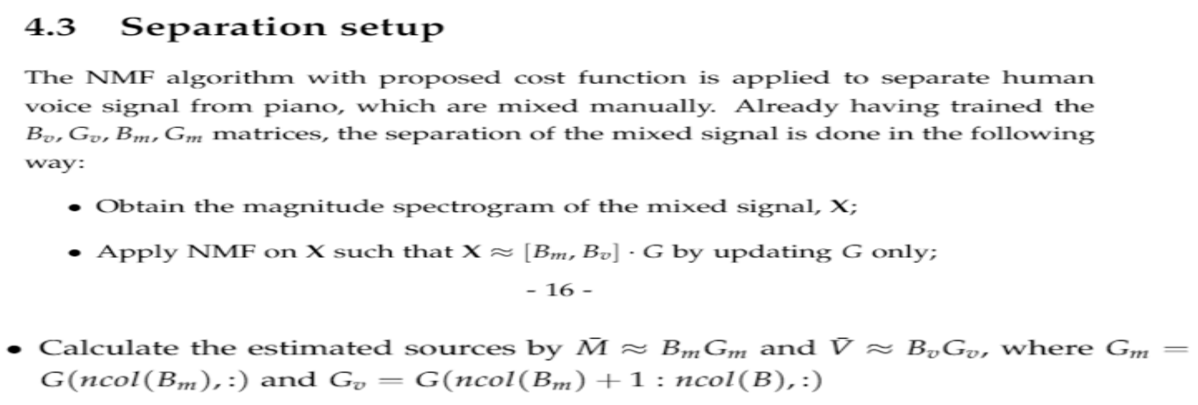


Figure 7: Signal Seperation



4.3 Separation setup

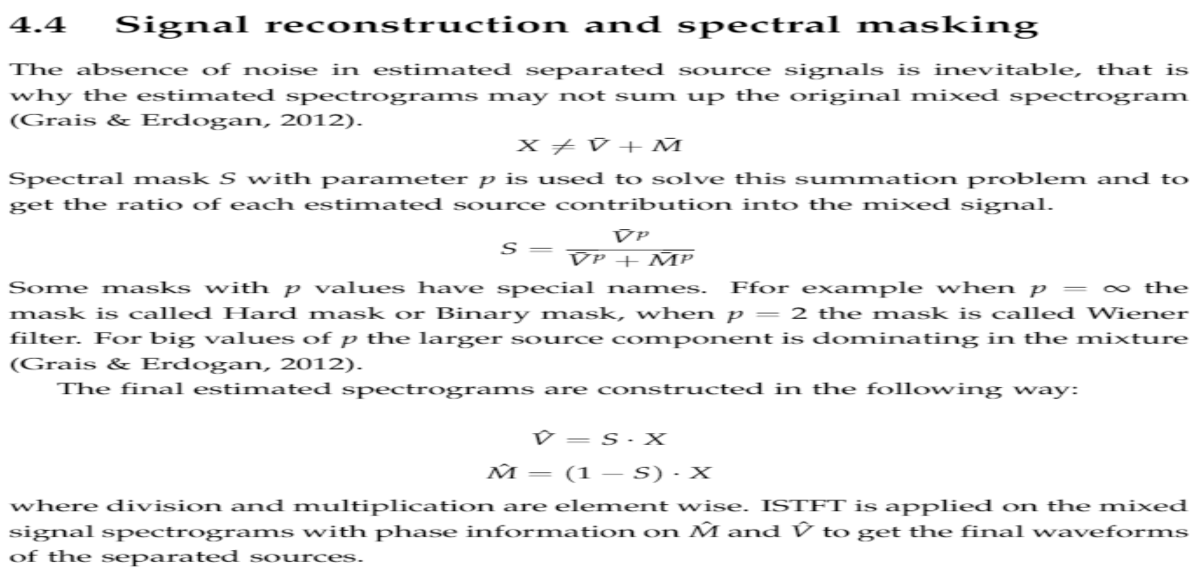
The NMF algorithm with proposed cost function is applied to separate human voice signal from piano, which are mixed manually. Already having trained the B_v, G_v, B_m, G_m matrices, the separation of the mixed signal is done in the following way:

- Obtain the magnitude spectrogram of the mixed signal, $\mathbf{X}$;
- Apply NMF on $\mathbf{X}$ such that $\mathbf{X} \approx [B_m, B_v] \cdot G$ by updating G only;

- 16 -

- Calculate the estimated sources by $\bar{M} \approx B_m G_m$ and $\bar{V} \approx B_v G_v$, where $G_m = G(ncol(B_m), :)$ and $G_v = G(ncol(B_m) + 1 : ncol(B), :)$

Figure 8: Seperation algorithm from: https://www.researchgate.net/publication/326914780_Supervised_sound_source_separation_using_Nonnegative_Matrix_Factorization Page 15-16



4.4 Signal reconstruction and spectral masking

The absence of noise in estimated separated source signals is inevitable, that is why the estimated spectrograms may not sum up the original mixed spectrogram (Grais & Erdogan, 2012).

$$X \neq \hat{V} + \hat{M}$$

Spectral mask S with parameter p is used to solve this summation problem and to get the ratio of each estimated source contribution into the mixed signal.

$$S = \frac{\hat{V}^p}{\hat{V}^p + \hat{M}^p}$$

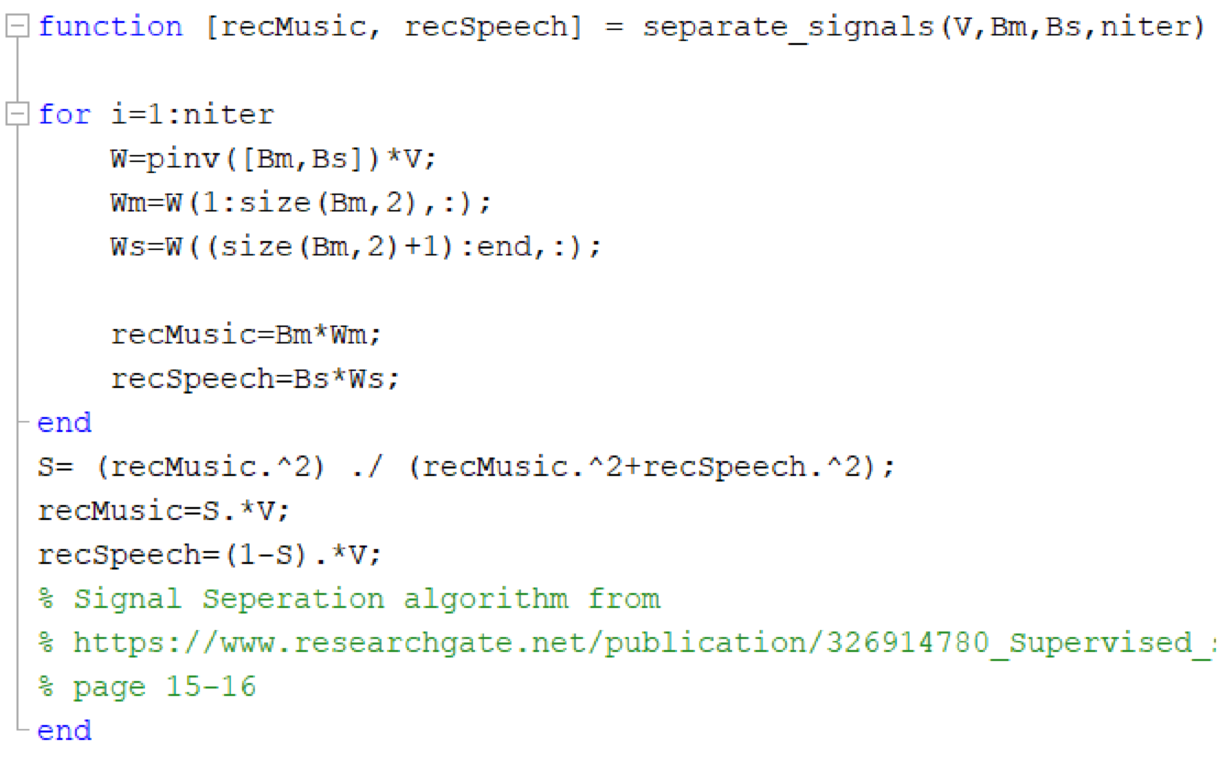
Some masks with p values have special names. Ffor example when $p = \infty$ the mask is called Hard mask or Binary mask, when $p = 2$ the mask is called Wiener filter. For big values of p the larger source component is dominating in the mixture (Grais & Erdogan, 2012).

The final estimated spectrograms are constructed in the following way:

$$\hat{V} = S \cdot X$$
$$\hat{M} = (1 - S) \cdot X$$

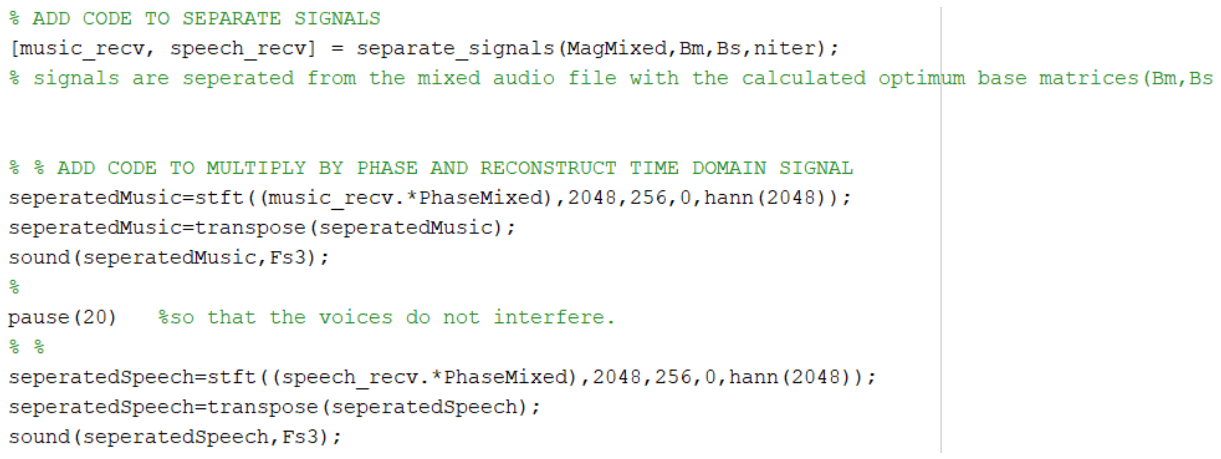
where division and multiplication are element wise. ISTFT is applied on the mixed signal spectrograms with phase information on $\hat{M}$ and $\hat{V}$ to get the final waveforms of the separated sources.

Figure 9: Signal reconstruction (Notations are different. V and M correspond to spectrograms of music and speech signals.)

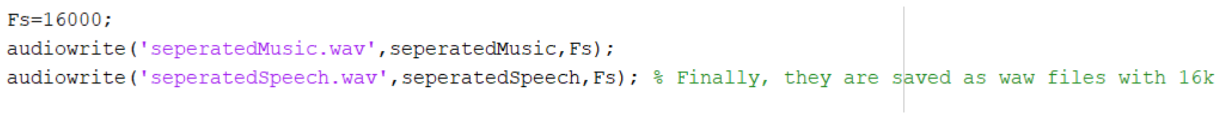


```
function [recMusic, recSpeech] = separate_signals(V,Bm,Bs,niter)

for i=1:niter
     W=pinv([Bm,Bs])*V;
     Wm=W(1:size(Bm,2),:);
     Ws=W((size(Bm,2)+1):end,:);

     recMusic=Bm*Wm;
     recSpeech=Bs*Ws;
end
S= (recMusic.^2) ./ (recMusic.^2+recSpeech.^2);
recMusic=S.*V;
recSpeech=(1-S).*V;
% Signal Seperation algorithm from
% https://www.researchgate.net/publication/326914780_Supervised_
% page 15-16
end
```

Figure 10: Signal_separation function.



```
% ADD CODE TO SEPARATE SIGNALS
[music_recv, speech_recv] = separate_signals(MagMixed,Bm,Bs,niter);
% signals are seperated from the mixed audio file with the calculated optimum base matrices(Bm,Bs


% % ADD CODE TO MULTIPLY BY PHASE AND RECONSTRUCT TIME DOMAIN SIGNAL
seperatedMusic=stft((music_recv.*PhaseMixed),2048,256,0,hann(2048));
seperatedMusic=transpose(seperatedMusic);
sound(seperatedMusic,Fs3);
%
pause(20)   %so that the voices do not interfere.
% %
seperatedSpeech=stft((speech_recv.*PhaseMixed),2048,256,0,hann(2048));
seperatedSpeech=transpose(seperatedSpeech);
sound(seperatedSpeech,Fs3);
```



```
Fs=16000;
audiowrite('seperatedMusic.wav',seperatedMusic,Fs);
audiowrite('seperatedSpeech.wav',seperatedSpeech,Fs); % Finally, they are saved as waw files with 16k
```

Figure 11: Separated Music and Speech signals are obtained.

Obtained “music_recv” and “speech_recv” are actually spectrogram of time domain signals. So that, we multiply them with phase matrices and take inverse short-time fourier transform(istft) to get time domain audio signals.

We play these time domain audio signals to observe if we can separate signals well. Finally, separated speech and music files are saved using audiowrite function.

According to our observations; We have successfully separated the signals, but there is some noise and the amplitude of the signals is slightly less than the original. You can run the code or save it and listen that way. They are also available directly in the homework file we send.