**Middle East Technical University Electrical-Electronics Engineering Department**

**EE301 Term Project (2016-2017 Fall Semester)**

**PHASE 2**

Berk İSKENDER 2031920

Asena Melisa SARICI 2031284

**Synthesis of the sound signals**

**7)**

In this part, as required we synthesized the three sound signals using the provide code for a duration of 5 seconds. And using the code sound(sound\_filename,Fs) we played the synthesized sounds and observed that they were very similar to the original sounds. Among them the violin was the most similar, the flute follows it and then comes the singer.

We also plotted the time domain plots of the signals for 1.36<t<1.415 and found the following data shown in Figure 1, 2 and 3.

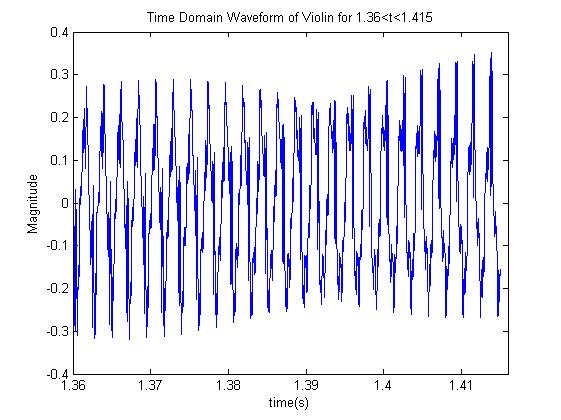


Figure 1. Time Domain Waveform for Synthesized Violin Sound

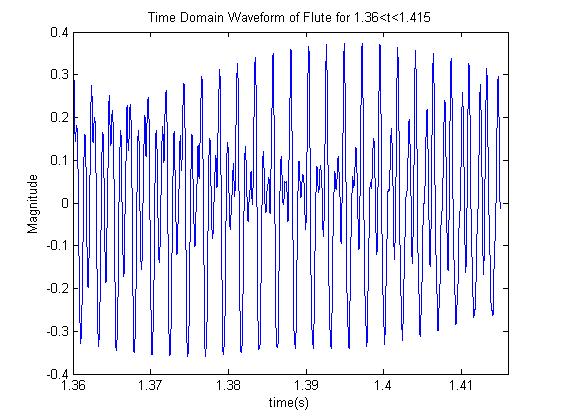


Figure 2. Time Domain Waveform for Synthesized Flute Sound

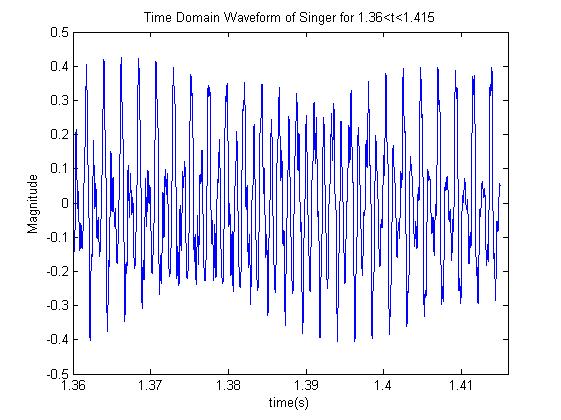


Figure 3. Time Domain Waveform for Synthesized Singer Sound

The code used for the reconstruction of the signals by making use of the FS coefficients

% For flute, the magnitude and the phase of the FS coefficients are  
flute\_mag=[0.7795 1.0000 0.5534 0.1867 0.0780 0.0273 0.0492 0.0262 0.0378 0.0223]; % for k=1,..,10.  
flute\_phase=[1.2746 5.8427 0.7077 4.3386 4.9902 1.9148 0.2783 3.6495 1.0951 5.7888]; % in radians for k=1,..,10.  
  
violin\_mag=[1 0 0.2098 0.1225 0.1745 0.03614 0.08944 0.1911 0.08307 0.2004 0.04268 0.08082 0.115 0.1097 0.0233 0.02627 0.0205 0.0101 0.006174]; % for k=1,..,19.  
violin\_phase=[0.4319 0 4.1878 3.4878 1.5787 4.5619 1.2929 5.1425 4.4743 0.6902 5.1557 0.0548 2.4212 3.0708 2.8690 1.8168 1.9318 2.9805 3.3459]; % in radians for k=1,..,19.  
  
singer\_mag=[0.4579 1 0.4448 0.05475 0 0 0.2663 0.1466 0]; % for k=1,..,9.  
singer\_phase=[3.6159 5.8819 5.9292 2.0619 0 0 5.3230 4.5716 0]; % in radians % for k=1,..,9.  
  
frequency\_violin=[441.4 882 1346 1793 2239 2686 3138 3585 4037 4484 4926 5378 5825 6277 6724 7176 7628 8075 8522];  
frequency\_singer=[463 923 1389 1849 2312 2786 3243 3709 4177];  
frequency\_flute=[436 866.7 1314 1750 2180 2638 3063 3515 3935 4355];  
  
Fs=44100; %Hz  
sound\_violin\_freq=zeros(1,19);  
sound\_violin=zeros(1,5\*44100);  
  
for t=1:5\*Fs  
for k=1:19  
sound\_violin(t)=sound\_violin(t)+(violin\_mag(k))\*cos(2\*pi\*t.\*(frequency\_violin(k))./Fs);  
end  
end  
  
M1=max(sound\_violin);  
sound\_violin=sound\_violin./M1;  
sound\_violin=sound\_violin.\*0.5;  
  
  
sound\_singer\_freq=zeros(1,19);  
sound\_singer=zeros(1,5\*44100);  
  
for t=1:5\*Fs  
for k=1:9  
sound\_singer(t)=sound\_singer(t)+(singer\_mag(k))\*cos(2\*pi\*t.\*(frequency\_violin(k))./Fs);  
end  
end  
  
M2=max(sound\_singer);  
sound\_singer=sound\_singer./M2;  
sound\_singer=sound\_singer.\*(0.5);  
  
sound\_flute\_freq=zeros(1,19);  
sound\_flute=zeros(1,5\*44100);  
  
for t=1:5\*Fs  
for k=1:9  
sound\_flute(t)=sound\_flute(t)+(flute\_mag(k))\*cos(2\*pi\*t.\*(frequency\_flute(k))./Fs);  
end  
end  
  
M3=max(sound\_flute);  
sound\_flute=sound\_flute./M3;  
sound\_flute=sound\_flute.\*(0.5);  
  
%For time domain signal plots:  
T=59976:62401;  
figure;  
plot(T/44100,sound\_violin(T));  
xlabel('time(s)');  
ylabel('Magnitude');  
xlim([1.36 1.416])  
title('Time Domain Waveform of Violin for 1.36<t<1.415');  
T=59976:62401;  
figure;  
plot(T/44100,sound\_flute(T));  
xlabel('time(s)');  
ylabel('Magnitude');  
title('Time Domain Waveform of Flute for 1.36<t<1.415');  
xlim([1.36 1.416])  
T=59976:62401;  
figure;  
plot(T/44100,sound\_singer(T));  
xlabel('time(s)');  
ylabel('Magnitude');  
title('Time Domain Waveform of Singer for 1.36<t<1.415');  
xlim([1.36 1.416])

[*Published with MATLAB® R2013a*](http://www.mathworks.com/products/matlab)

**Recognition of sound signal type**

**8)**

The MSE values for singer, violin and flute is found for each sound signals. These values are shown is Table 1.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Violin | Flute | Singer |
| Another\_Violin\_A4.wav | ***0.0163109*** | 0.0735715 | 0.0803173 |
| Another\_Flute\_A4.wav | 0.0523944 | ***0.0166126*** | 0.0514202 |
| Another\_Singer\_A4.wav | 0.0769939 | 0.0309234 | ***0.0099179*** |

Table 1. MSE values of Violin, Flute and Singer for each sound signal

As observed from the table, the MSE values for each instrument has the least value for their own sound signal sample. These values are indicated in bold and italic numbers in the table.

This is owing to the fact that what characterizes an instrument is the coefficient values for each kf0. And since violin-violin; flute-flute; singer-singer has the most similar coefficient values, the errors for each coefficient are the smallest, hence MSE for these cases (as an average error estimation) are the smallest.

The code used for the recognition of the source of different sound signals

function out=recognize\_sound\_Group\_149(audio\_signal\_file\_name)  
  
%ak's for initial sound files  
  
flute\_mag=[0.7795 1.0000 0.5534 0.1867 0.0780 0.0273 0.0492 0.0262 0.0378 0.0223]; % for k=1,..,10.  
flute\_phase=[1.2746 5.8427 0.7077 4.3386 4.9902 1.9148 0.2783 3.6495 1.0951 5.7888]; % in radians for k=1,..,10.  
  
violin\_mag=[1 0 0.2098 0.1225 0.1745 0.03614 0.08944 0.1911 0.08307 0.2004 0.04268 0.08082 0.115 0.1097 0.0233 0.02627 0.0205 0.0101 0.006174]; % for k=1,..,19.  
violin\_phase=[0.4319 0 4.1878 3.4878 1.5787 4.5619 1.2929 5.1425 4.4743 0.6902 5.1557 0.0548 2.4212 3.0708 2.8690 1.8168 1.9318 2.9805 3.3459]; % in radians for k=1,..,19.  
  
singer\_mag=[0.4579 1 0.4448 0.05475 0 0 0.2663 0.1466 0]; % for k=1,..,9.  
singer\_phase=[3.6159 5.8819 5.9292 2.0619 0 0 5.3230 4.5716 0]; % in radians % for k=1,..,9.  
  
Fs=44100;  
  
[audio\_signal, Fs]=audioread(audio\_signal\_file\_name);  
  
 time\_index=audio\_signal(44100:52920);  
  
 N=2^12;  
 CTFT\_sampled = fft((time\_index),N)/N;  
 CTFT\_sampled\_abs = abs(CTFT\_sampled(2:N/2));  
 freq = (1:N/2-1)\*Fs/N;  
 CTFT\_sampled\_abs=CTFT\_sampled\_abs/max(CTFT\_sampled\_abs);  
  
 for k=1:20  
 ind\_440(k)=ceil(N/Fs\*440\*(k));  
 error\_interval=ceil(N/Fs\*40);  
  
 b(k)=max(CTFT\_sampled\_abs(ind\_440(k)-error\_interval:ind\_440(k)+error\_interval));  
 end  
 % finding the maximum value.  
  
E\_violin=zeros(1,19);  
E\_singer=zeros(1,9);  
E\_flute=zeros(1,10);  
  
for k=1:19  
E\_violin(k)=(abs(violin\_mag(k))-abs(b(k)));  
end  
  
for k=1:9  
E\_singer(k)=(abs(singer\_mag(k))-abs(b(k)));  
end  
  
for k=1:10  
E\_flute(k)=(abs(flute\_mag(k))-abs(b(k)));  
end  
  
 MSE\_violin=0;  
 MSE\_singer=0;  
 MSE\_flute=0;  
  
 for k=1:19  
 MSE\_violin=MSE\_violin+(2/39).\*(E\_violin(k).^2);  
 end  
  
 fprintf('MSE value for violin is %g.\n', MSE\_violin);  
  
  
 for k=1:10  
 MSE\_flute=MSE\_flute+(2/21).\*(E\_flute(k).^2);  
 end  
  
 fprintf('MSE value for flute is %g.\n', MSE\_flute);  
  
  
 for k=1:9  
 MSE\_singer=MSE\_singer+(2/19).\*(E\_singer(k).^2);  
 end  
  
 fprintf('MSE value for singer is %g.\n', MSE\_singer);  
  
if MSE\_singer==min(MSE\_singer,MSE\_flute) && MSE\_singer==min(MSE\_violin,MSE\_singer)  
 fprintf('The sound signal is a singer signal');  
 out=3;  
end  
  
if MSE\_flute==min(MSE\_violin,MSE\_flute) && MSE\_flute==min(MSE\_flute,MSE\_singer)  
 fprintf('The sound signal is a flute signal');  
 out=2;  
end  
  
 if MSE\_violin==min(MSE\_violin,MSE\_flute) && MSE\_violin==min(MSE\_violin,MSE\_singer)  
 fprintf('The sound signal is a violin signal');  
 out=1;  
 end

[*Published with MATLAB® R2013a*](http://www.mathworks.com/products/matlab)

**Sampling rate change**

**10)**

When the sampling rate changed by a ratio of 10, we observed that for singer case, reconstructed sound signal had more resemblance to the original one with some defections. On the contrary, for the flute and violin cases, reconstructed signals were nearly unrecognizable compared to the original ones. This is expected since flute and violin signals have wider bandwidth compared to the singer signal. (10 kHz and 4 kHz respectively) All of these reconstructred signals are undersampled. However, flute and violin cases are relatively more undersampled according to their bandlimits and this causes higher distortions from the original signals during reconstruction. (Our Fs’=4410 Hz and the Nyquist condition is Fs>2Fm)

**11)** When we repeated the same steps for the singer case with Fs’=Fs/2, we obtained almost the same signal back. In this step, Fs’ was larger than Nyquist condition according to our assumption of singer signal is bandlimited to 4 kHz. Owing to this fact, we were able to reconstruct the original signal nearly without any information loss.