Joanna Masikowska

B9TB1710

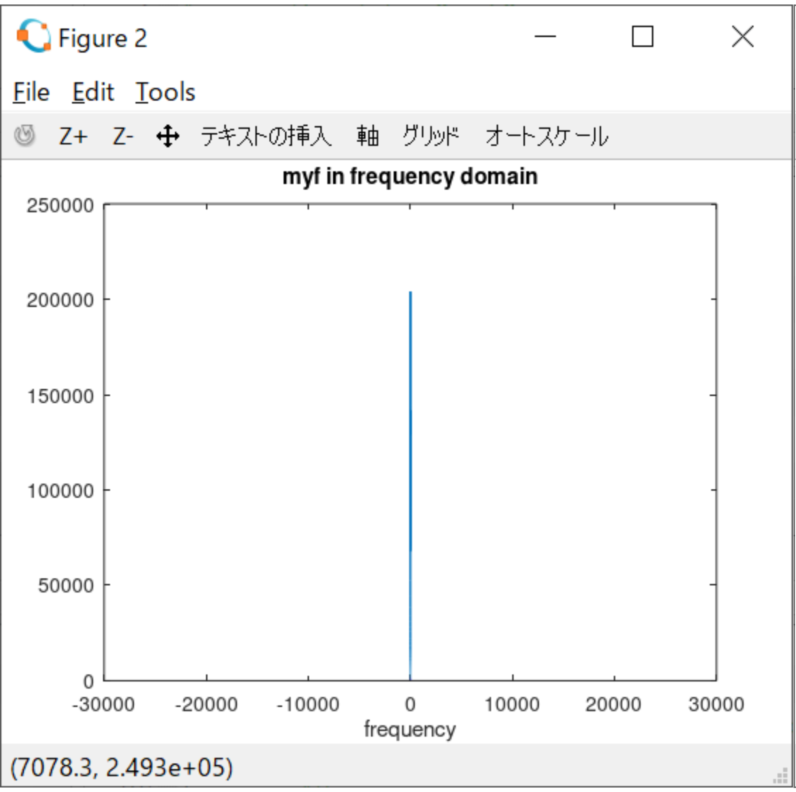
I start from loading audio file “test.wav” with function **audioread,** assigning audio data to matrix f and sampling rate to $freq$. All audio signals I will work with have the same sampling rate equal to 44100 points per second. I find the time length of test.wav and assign it to a variable $x1$. Based on $x1$ and number of audio data in f, I create the time domain x.

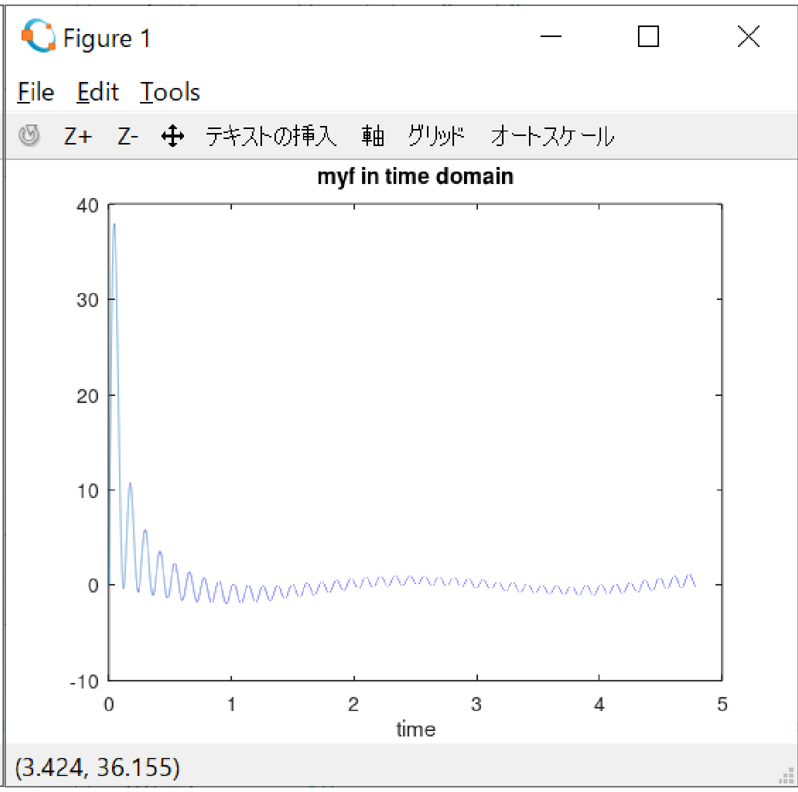
CAPS07_B9TB1710(1).m | CAPS07_B9TB1710(2).m | * <unnamed>

```
[f,freq]=audioread("test.wav");

x1=length(f)/freq; #time length of the audio
x=linspace(0,x1,length(f)); #time domain

#my audio data "myf"
myf=0;
i=1;
imax=50;
while i<=imax
  myf=myf+sin((i+2)*x);
  i+=1;
endwhile
myf=myf';

#figure 1
plot(x,myf); #myf in time domain
title("myf in time domain");
xlabel("time");
set(gca,"fontsize",14);

#Fourier transormation of myf
myF=fft(myf);
myFshift=fftshift(myF);#shifting myF

df=freq/length(myF); #resolution[Hz]
x2=-freq/2:df:freq/2-df; #frequency domain

figure #2
plot(x2,abs(myFshift)); #myf in frequency domain
title("myf in frequency domain");
xlabel("frequency");
```

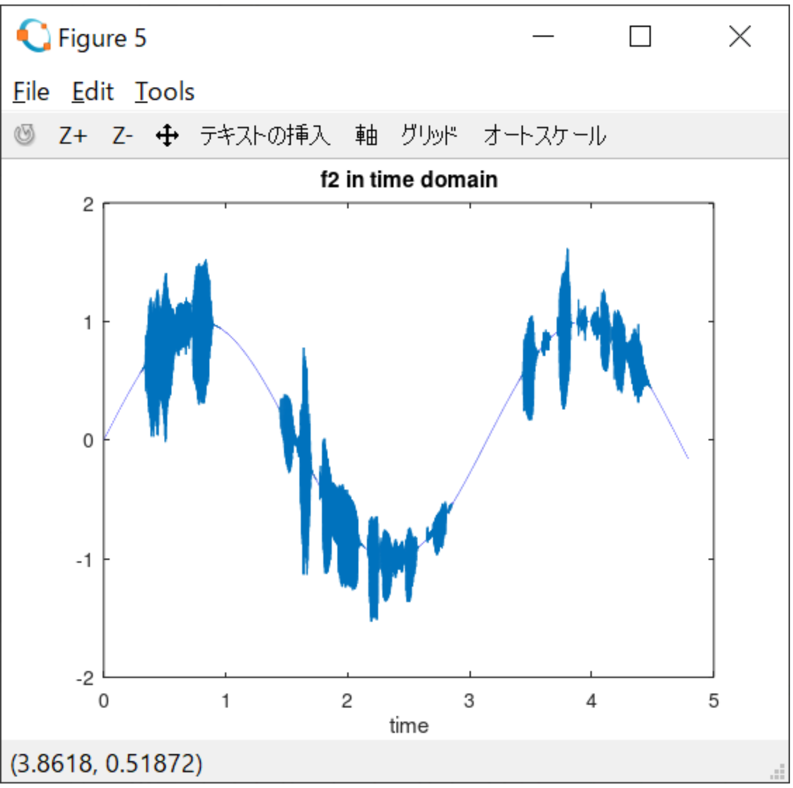
```
figure #2
plot(x2,abs(myFshift)); #myf in frequency domain
title("myf in frequency domain");
xlabel("frequency");
set(gca,"fontsize",14);

figure #3
plot(x,f); #f in time domain
title("f in time domain");
xlabel("time");
set(gca,"fontsize",14);

#Fourier transormation of test.wav
Fshift=fftshift(fft(f));#shifting F

figure #4
plot(x2,abs(Fshift)); #f in frequency domain
title("f in frequency domain");
xlabel("frequency");
set(gca,"fontsize",14);

#adding sine wave to test.wav
f2=sin(2*x)'.+f;

figure #5
plot(x,f2); #f2 in time domain
title("f2 in time domain");
xlabel("time");
set(gca,"fontsize",14);

F2shift=fftshift(fft(f2));
```

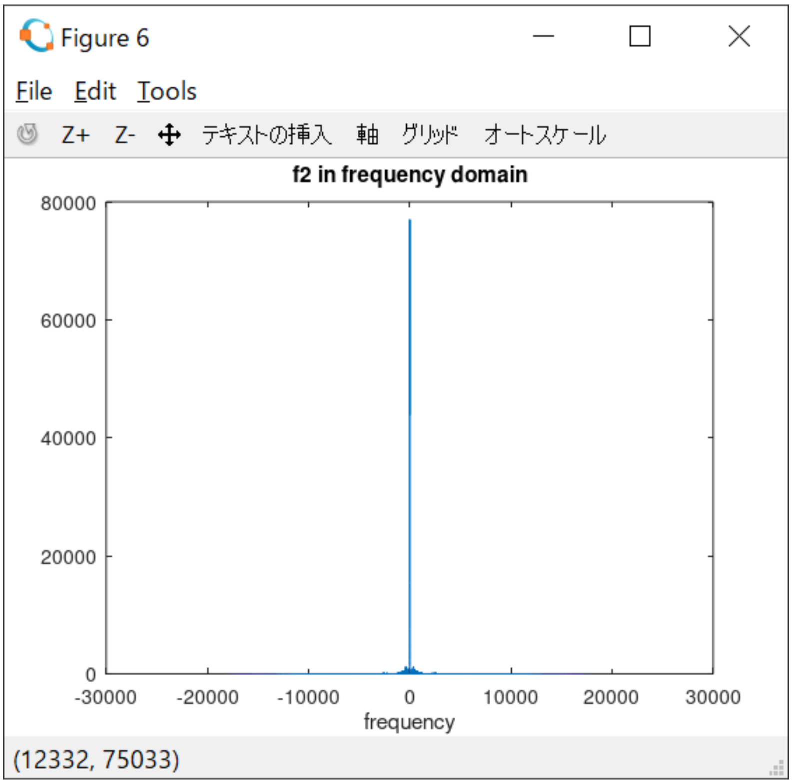
1. Using while loop, I create my audio data which consists of sum of different sine waves. I name it “myf”. The time length my audio signal is the same as in of “test.wav”. I plot my audio signal myf in the time domain x. After that, using function **fft** I perform Fourier transformation to change the domain of my signal to frequency domain $x2$. I shift the domain so that 0 frequency is in the middle and I plot my result. The graphs are as below. I play it using function **sound** in in the end of the script.





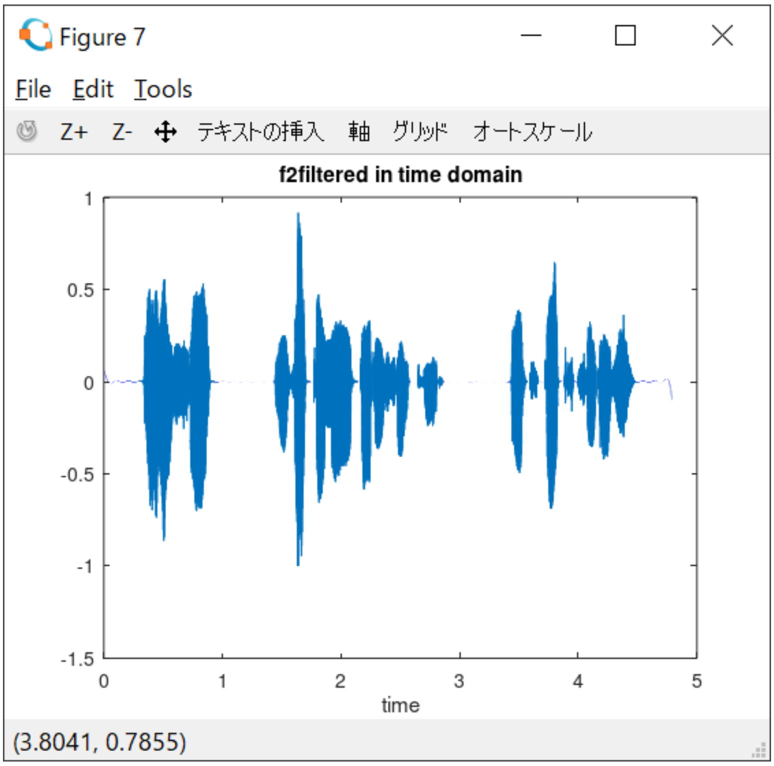
2. I add one sine wave signal $sin(2x)$ to f and name it $f2$. Using the same method as in (1), I plot $f2$ in time domain and frequency domain as below.

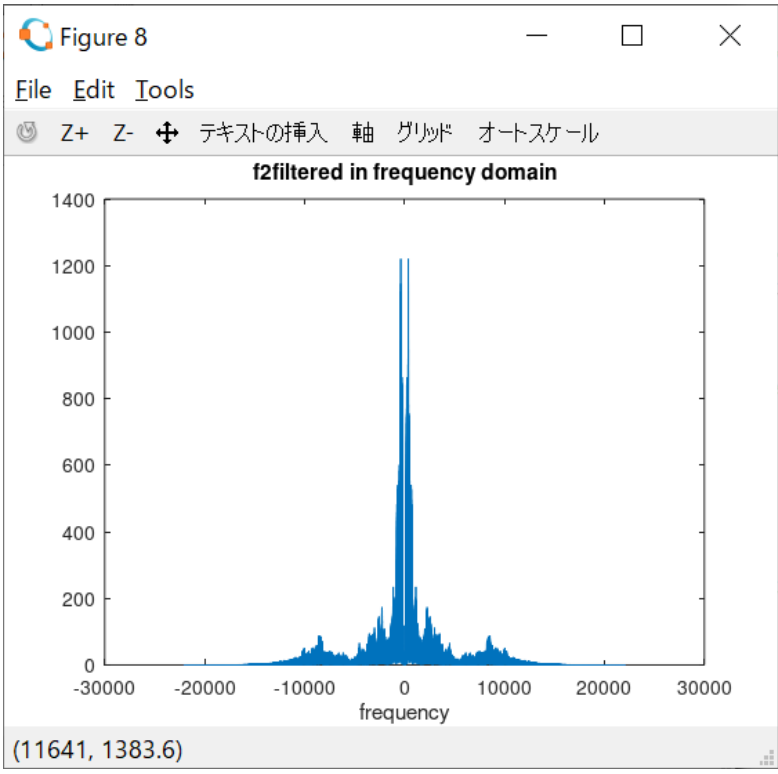




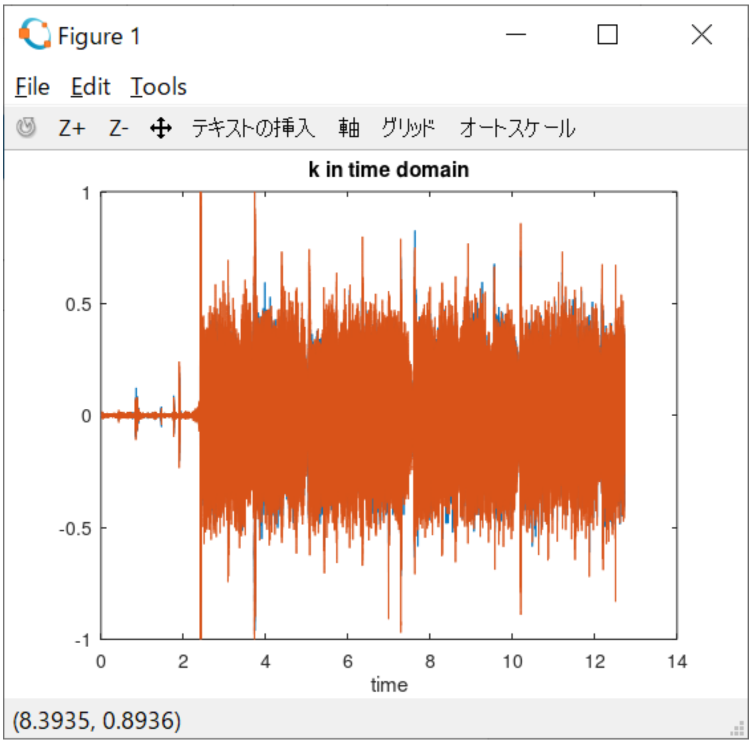
```
xlabel("time");
set(gca,"fontsize",14);

F2shift=fftshift(fft(f2));

figure #6
plot(x2,abs(F2shift)); #f2 in frequency domain
title("f2 in frequency domain");
xlabel("frequency");
set(gca,"fontsize",14);

#filtering noise
filter=abs(x2)>10; #filtering out freq>-10Hz&freq<10Hz
f2filtered=ifft(ifftshift(F2shift.*filter'));

figure #7
plot(x,f2filtered); #f2filtered in time domain
title("f2filtered in time domain");
xlabel("time");
set(gca,"fontsize",14);

figure #8
plot(x2,abs(F2shift.*filter')); #f2filtered in frequnecy domair
title("f2filtered in frequency domain");
xlabel("frequency");
set(gca,"fontsize",14);

#playing all audios
sound(myf,freq);
sound(f2,freq);
sound(f2filtered,freq);
```

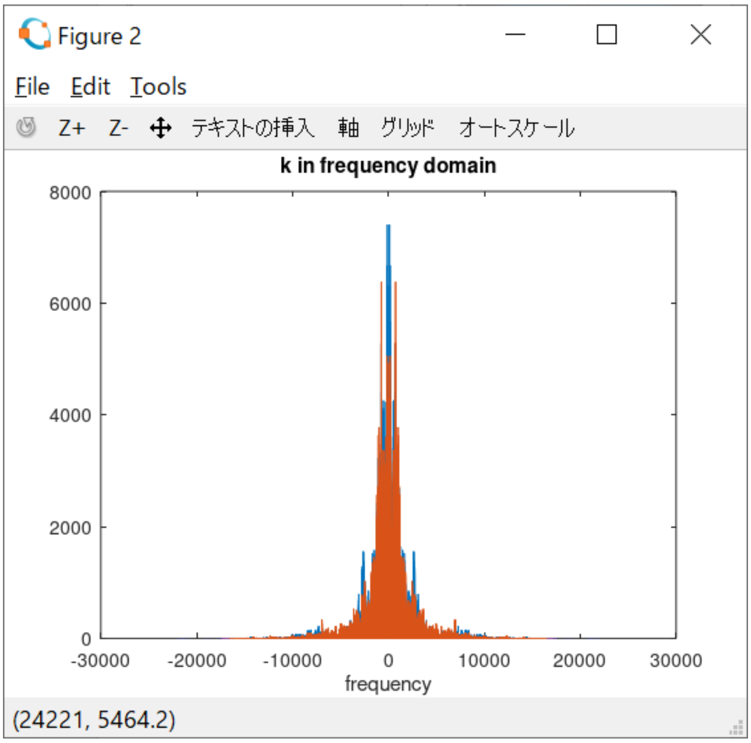
3. I compare graphs in frequency domain of f and $f2$. I can see that main difference is that $f2$ has much higher density of frequencies around zero. After zooming in, I notice that frequencies between $(-10,10)$ are the main cause of noise. Therefore, I create a variable $filter$ which takes value *True* for all frequencies other than within range $(-10,10)$. I multiply the filter by distribution of frequencies $F2shift$ which results in data that consists from frequencies for which filter takes value *True*. To present my result also in time domain, I reverse Fourier transformation on the filtered data with function **ifft**. Graphs of filtered audio signal are as below.

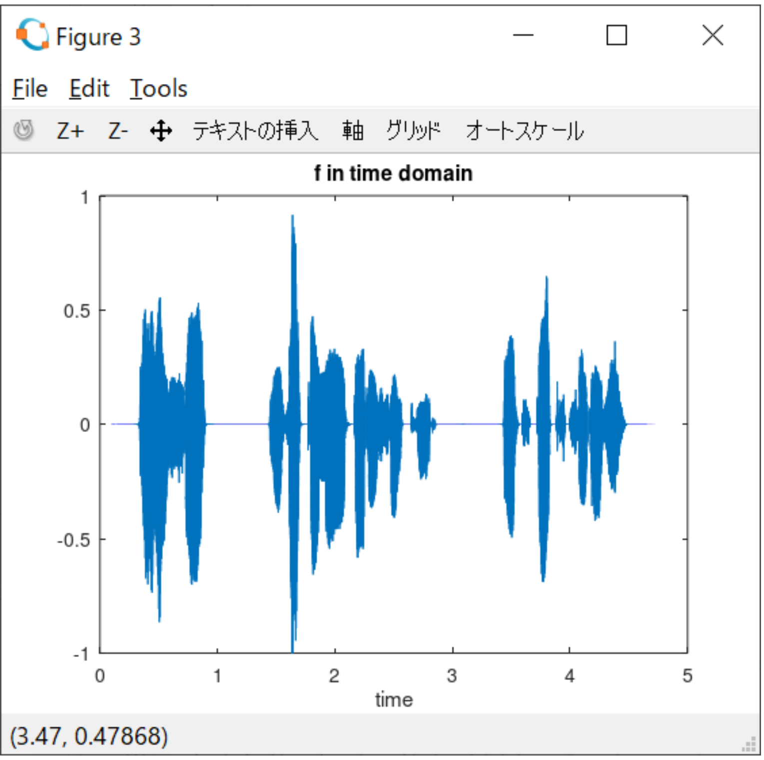


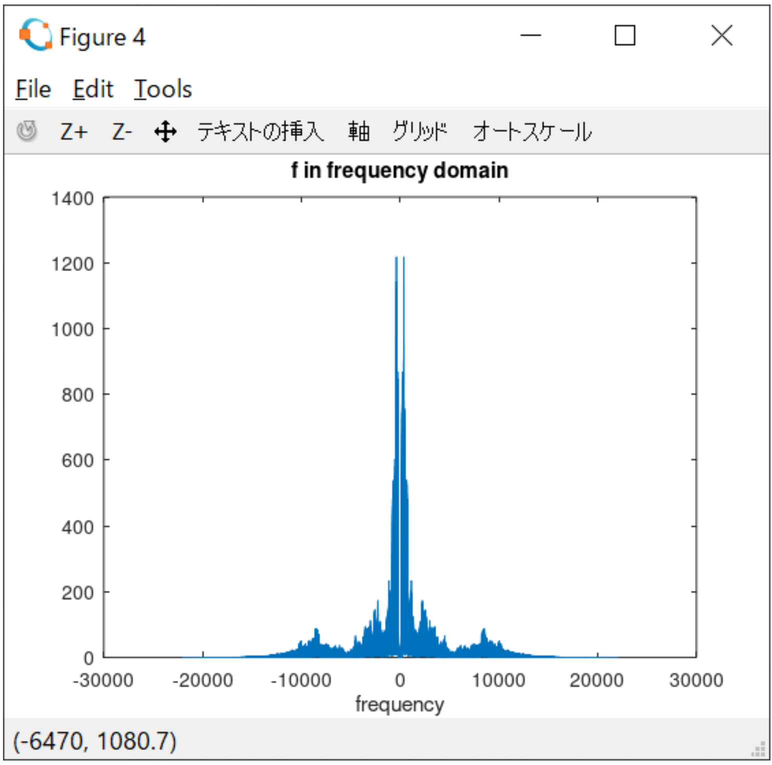


4. I load chosen audio file into Octave . Since it is a bit long, I cut it and assign shorter version to k. I graph it in time and frequency domains using method from (1) and compare it with graphs of audio data from test.wav.







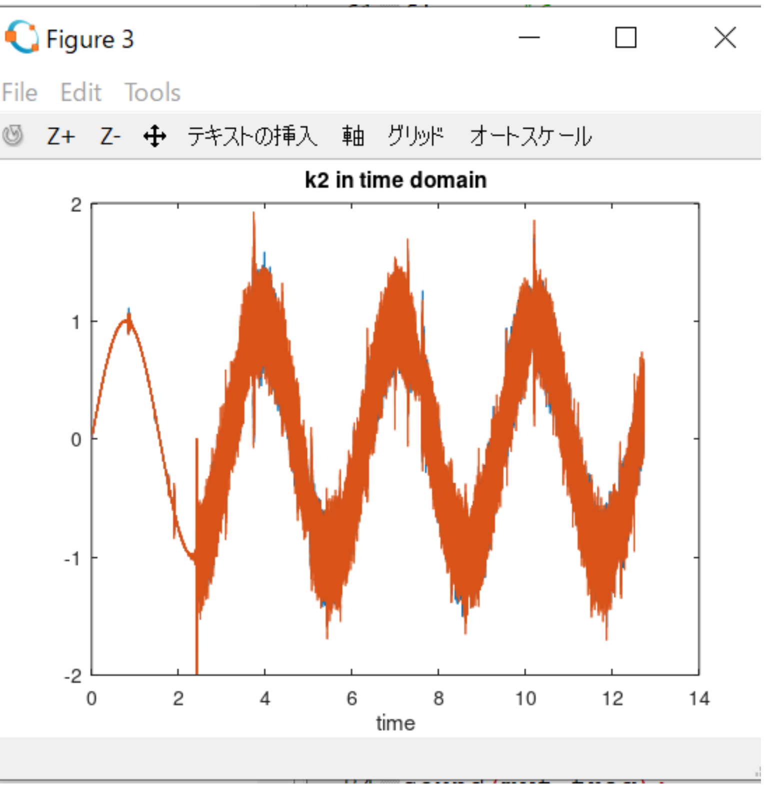


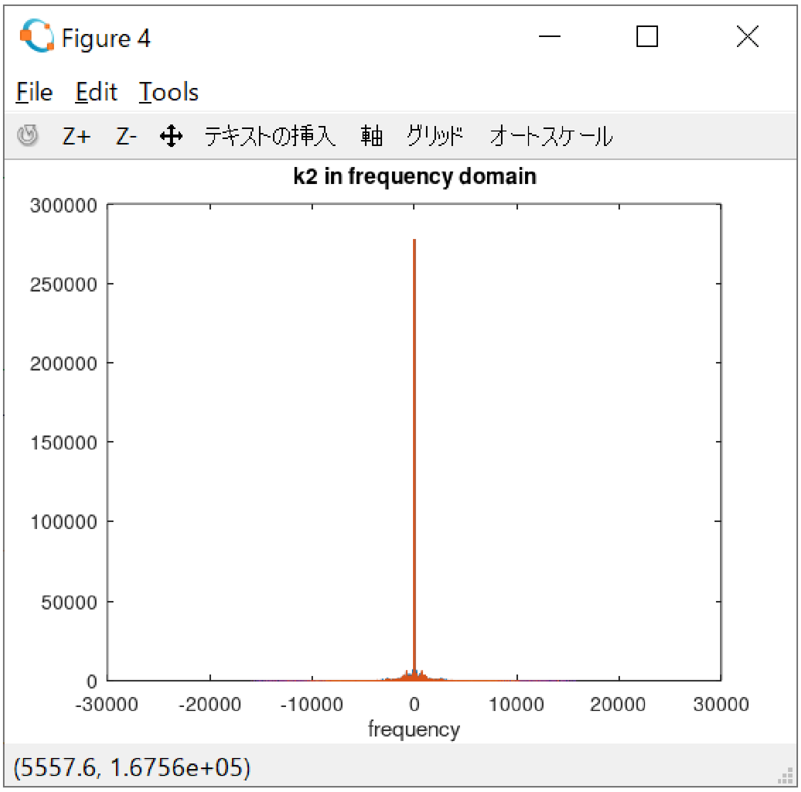

```
[g,freq]=audioread("CAPS07_B9TB1710.wav");
[f,freq]=audioread("test.wav");

k=g(1:561000,:);

xx1=length(k)/freq; #time length of the audio
xx = linspace(0,xx1,length(k)); #time domain

#Fourier transormation&shifting
K=fft(k);
Kshift=fftshift(K);

df=freq/length(K); #resolution[Hz]
xx2=-freq/2:df:freq/2-df; #frequency domain

#figure 1
plot(xx,k); #k in time domain
title("k in time domain");
xlabel("time");
set(gca,"fontsize",14);

figure #2
plot(xx2,abs(Kshift)); #k in frequency domain
title("k in frequency domain");
xlabel("frequency");
set(gca,"fontsize",14);

#adding sine wave to my audio
k2=sin(2*xx)'.+k;

#Fourier transormation&shifting
K2shift=fftshift(fft(k2));
```

```
title("k2 in time domain");
xlabel("time");
set(gca,"fontsize",14);

figure #4
plot(xx2,abs(K2shift)); #k2 in frequency domain
title("k2 in frequency domain");
xlabel("frequency");
set(gca,"fontsize",14);


#filtering noise
filter=abs(xx2)>10; #filtering out freq>-10Hz&freq<10Hz
k2filtered=ifft(ifftshift(K2shift.*filter'));

figure #5
plot(xx,k2filtered); #k2filtered in time domain
title("k2filtered in time domain");
xlabel("time");
set(gca,"fontsize",14);

figure #6
plot(xx2,abs(K2shift.*filter')); #k2filtered in freq domain
title("k2filtered in frequency domain");
xlabel("frequency");
set(gca,"fontsize",14);

#playing all audios
sound(k,freq);
sound(k2,freq);
sound(k2filtered,freq);
```

I can see that graphs of f and k differ noticeably. One difference is that f has one channel and so the there is only one plot of it. k, on the other hand, has two channels and which is reflected in orange and blue on its graph. Also, shapes of graphs in time domain is different. That is because f is the record of one person saying a sentence, while k is the record of a guitar. Graphs in frequency domain have similar shapes.

I repeat steps in (2) and (3). I call k summed with sine wave k2. Resulting graphs are as below.





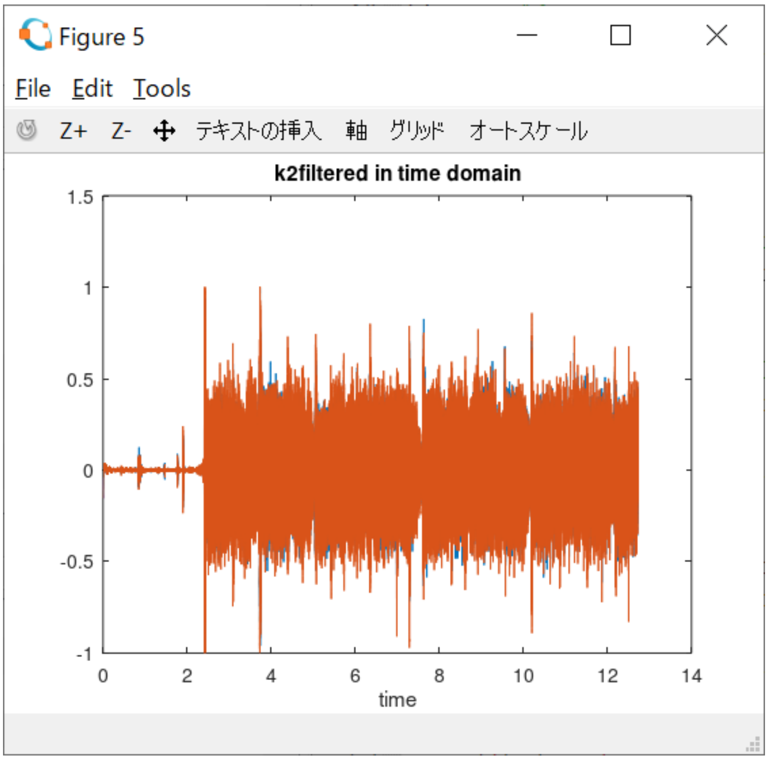


Figure 5
File Edit Tools
Z+ Z- テキストの挿入 軸 グリッド オートスケール
k2filtered in time domain
1.5
1
0.5
0
-0.5
-1
0 2 4 6 8 10 12 14
time

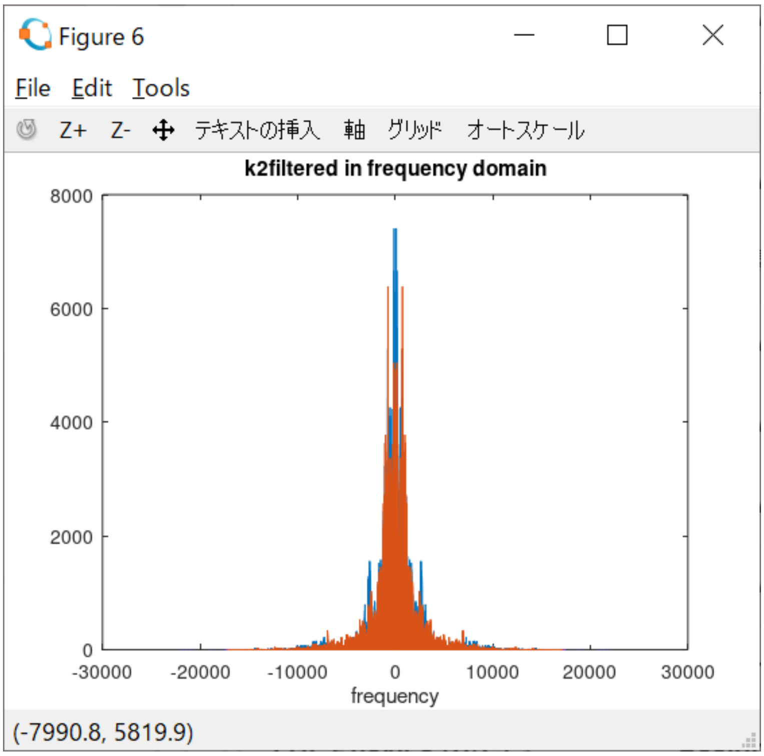


Figure 6
File Edit Tools
Z+ Z- テキストの挿入 軸 グリッド オートスケール
k2filtered in frequency domain
8000
6000
4000
2000
0
-30000 -20000 -10000 0 10000 20000 30000
frequency
(-7990.8, 5819.9)