Table of Contents

examples of recording & generating sound

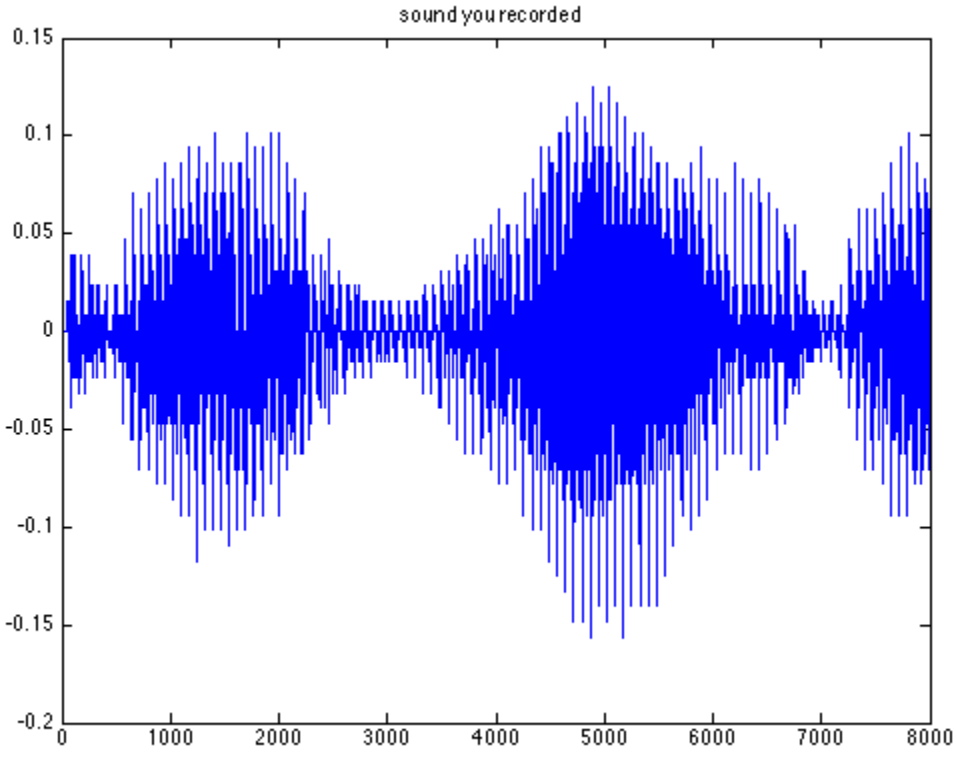
record and play back sound

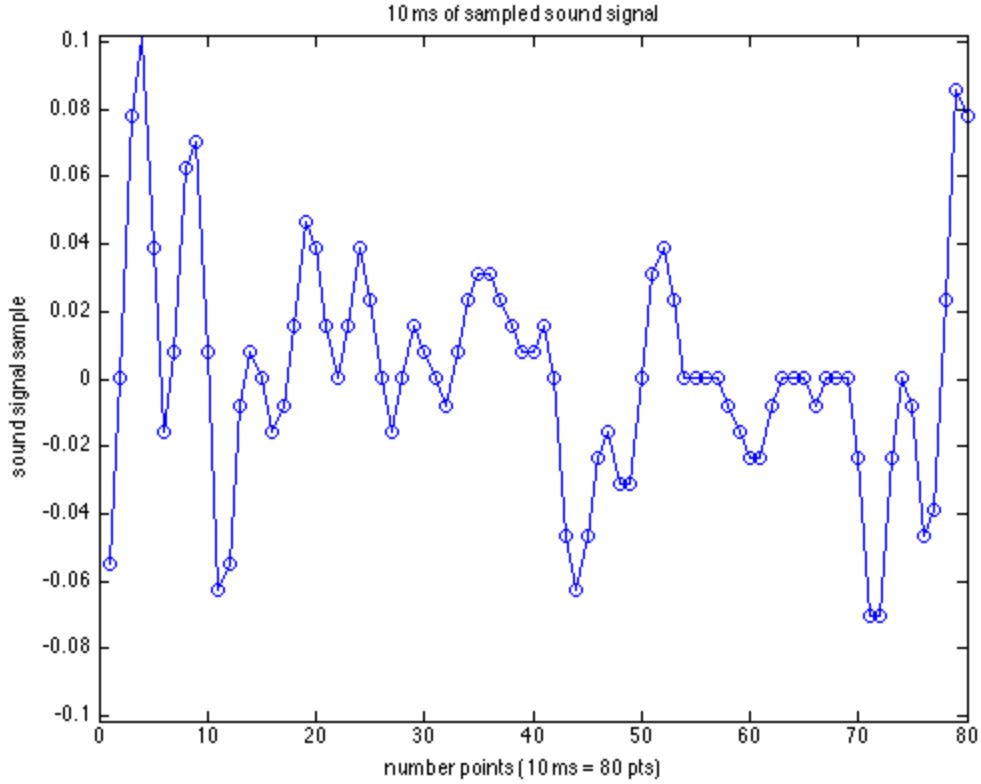
```
clear all
close all

% http://www.mathworks.com/help/matlab/import_export/record-and-play-audio.html
% default audiorecorder creates an 8000 Hz, 8-bit, 1-channel ... object
% audiorecorder can have options such as different sample rate

ttime = 1; % seconds to record sound
fprintf('Start speaking for %3.1f seconds \n',ttime)
recObj = audiorecorder; % create audiorecorder object, default options
recordblocking(recObj, ttime); % last arg is time in seconds
disp('End of Recording');

% get and plot entire sound sample
s = getaudiodata(recObj);
plot(s)
title('sound you recorded')

% plot a section of the sound sample
figure(2)
n = 80; % 80 pts = 10 ms
x = s(2000:n+2000);
maxx = max(x);
if maxx == 0 % in case select silent section
    maxx = 0.01;
end
plot(x,'-o')
axis([0 n -maxx maxx])
title('10 ms of sampled sound signal')
ylabel('sound signal sample')
xlabel('number points (10 ms = 80 pts)')

% two ways to play back sound
%    play() plays back audiorecorder object
%    sound() plays back array returned by getaudiodata()
% play(recObj)
sound(s,8000)
```

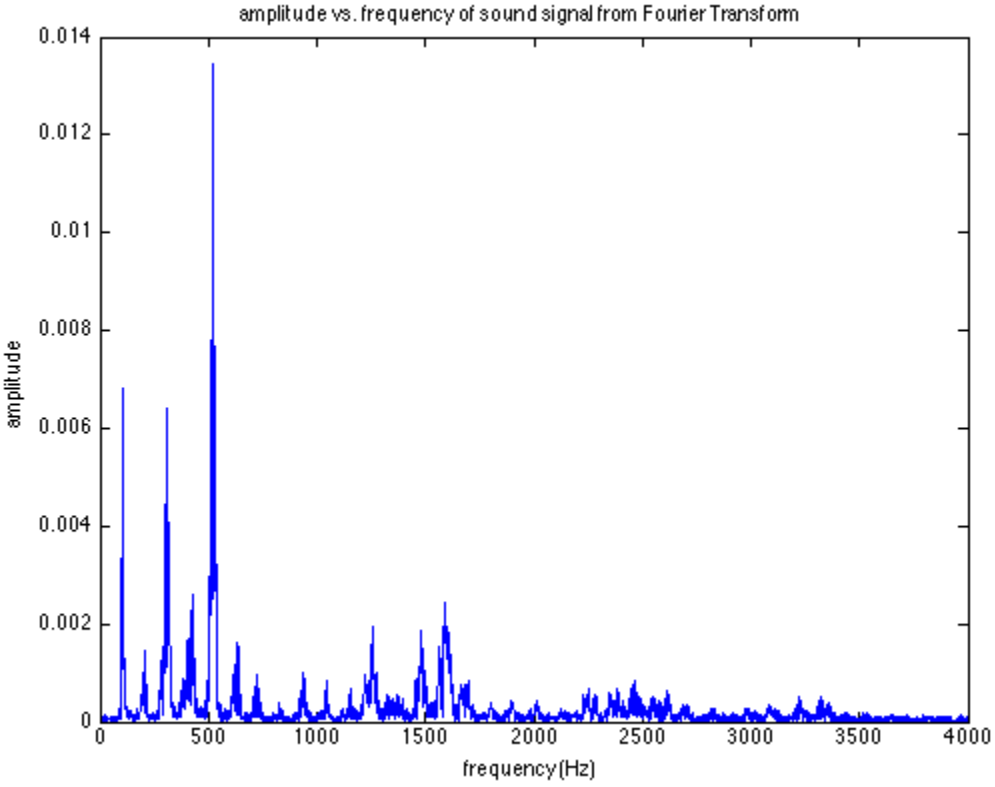
```
% use Fourier Transform to get frequency components
% code below from web example for fast Fourier transform
% from help nextpow2
%   nextpow2(N) returns the first P such that 2.^P >= abs(N)
tPtPerSec = 8000; % 8000 is default sample rate of audiorecorder obj
nfft = 2^nextpow2(tPtPerSec);
Y = fft(s,nfft)/tPtPerSec;
f = tPtPerSec/2*linspace(0,1,nfft/2+1);
figure(3)
plot(f,2*abs(Y(1:nfft/2+1)))
title('amplitude vs. frequency of sound signal from Fourier Transform')
ylabel('amplitude')
xlabel('frequency (Hz)')
```

Start speaking for 1.0 seconds

End of Recording

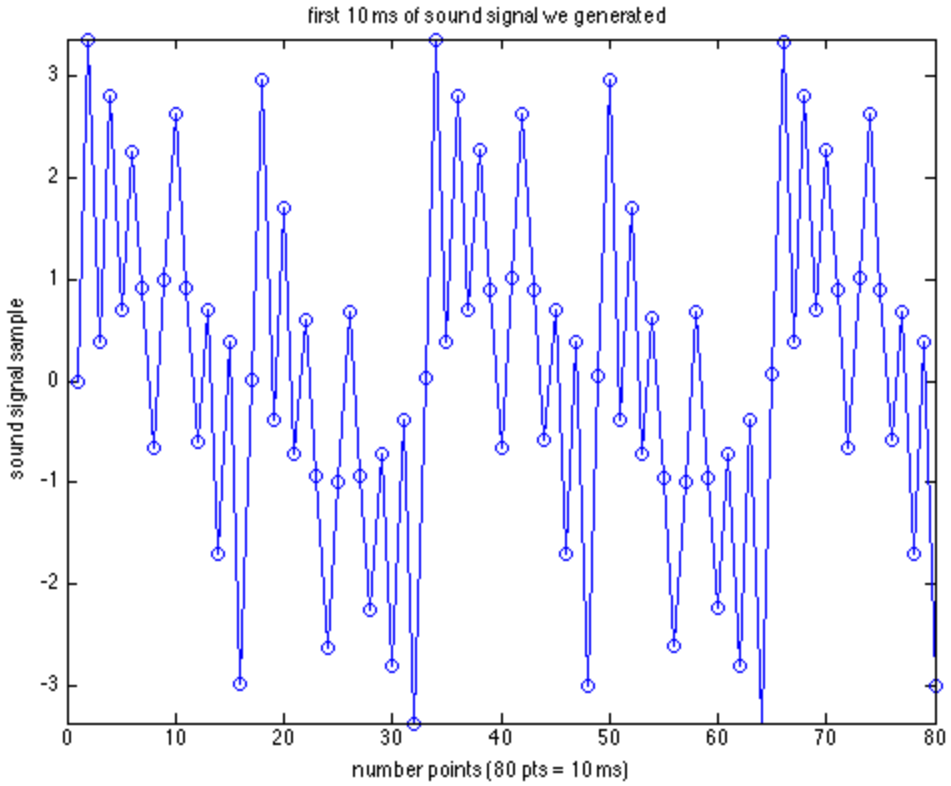


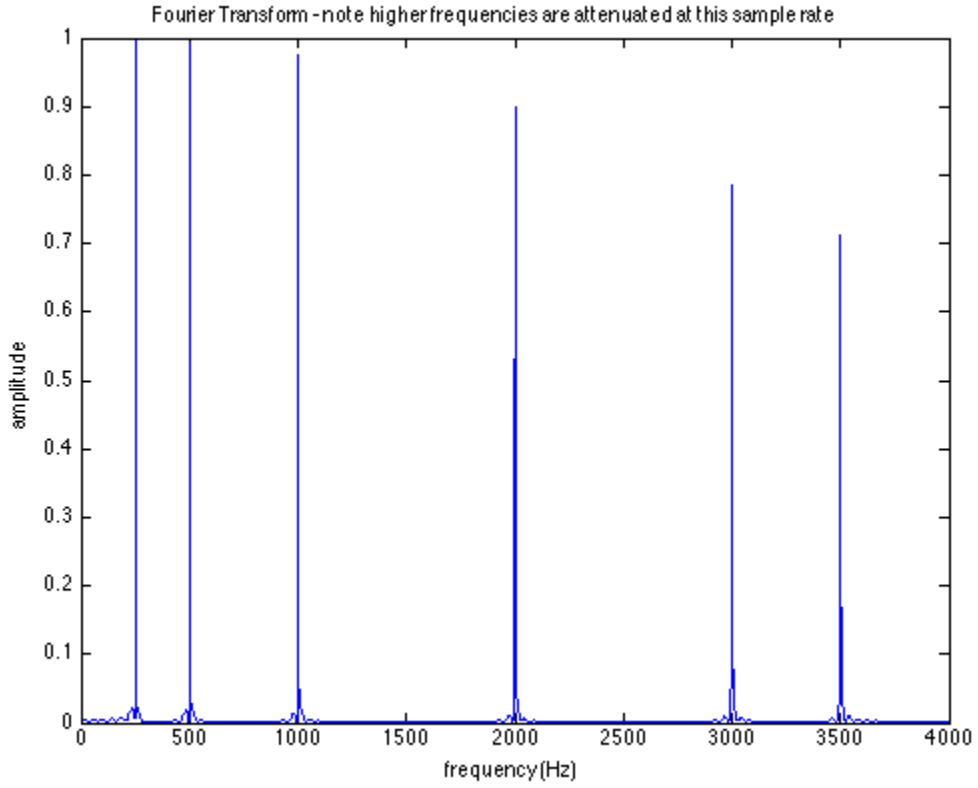

10 ms of sampled sound signal
sound signal sample
number points (10 ms = 80 pts)
0.1
0.08
0.06
0.04
0.02
0
-0.02
-0.04
-0.06
-0.08
-0.1
0
10
20
30
40
50
60
70
80


amplitude vs. frequency of sound signal from Fourier Transform
amplitude
frequency (Hz)
0.014
0.012
0.01
0.008
0.006
0.004
0.002
0
0
500
1000
1500
2000
2500
3000
3500
4000

generate sound from Fourier series

```
clear all
close all

% from help sound
%    sound() plays the sound at the default sample rate of 8192 Hz
% here we will specify a sample rate of 8000, which is that
% of default audiorecorder
tPtPerSec = 8000;
x = linspace(0,1,tPtPerSec); % generate time series for 1 second

% for 8 kHz sampling frequency, max freq that can be recorded
% is < 4 kHz per Nyquist-Shannon sampling theorem
% here pick small number of arbitrary frequencies in this range as example
Ff = [0.25 0.5 1 2 3 3.5] * 1000; % kHz, Fourier series frequencies
Fc = [1 1 1 1 1 1]; % Fourier series coefficient
Fp = 2 * pi * [0 0 0 0 0 0]; % Fourier series phase shift
% sum frequency components in Fourier series
s = 0*x; % initialize sound signal s using x from above
for k = 1:length(Ff)
    s = s + Fc(k)*sin(2*pi*Ff(k)*x + Fp(k));
end
% note that Matlab processes entire arrays s and x in one line
% eliminating an inner repeat required by other languages

% play the sound we generated
sound(s,tPtPerSec)

% plot a section of the sound we generated
n = 80; % 80 pts = 10 ms at tPtPerSec = 8000 Hz sampling freq
x = s(1:n);
maxx = max(x);
plot(x,'-o')
axis([0 n -maxx maxx])
title('first 10 ms of sound signal we generated')
ylabel('sound signal sample')
xlabel('number points (80 pts = 10 ms)')

% use Fourier Transform to get frequency components to check our input
% code below from a web example for fast Fourier transform
% from help nextpow2
%   nextpow2(N) returns the first P such that 2.^P >= abs(N)
nfft = 2^nextpow2(tPtPerSec);
Y = fft(s,nfft)/tPtPerSec;
f = tPtPerSec/2*linspace(0,1,nfft/2+1);
figure(2)
plot(f,2*abs(Y(1:nfft/2+1)))
title('Fourier Transform - note higher frequencies are attenuated at this sample r
ylabel('amplitude')
xlabel('frequency (Hz)')
```


first 10 ms of sound signal we generated
sound signal sample
number points (80 pts = 10 ms)
3
2
1
0
-1
-2
-3
0
10
20
30
40
50
60
70
80


Fourier Transform - note higher frequencies are attenuated at this sample rate
amplitude
frequency (Hz)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
500
1000
1500
2000
2500
3000
3500
4000