Matt Bachmeier

Lab 05 Part 1 ECE 203 3/5/2017

Contents

Section 3.1

Consider the continuous time sinusoid x(t)=A*cos(2*pi*f0*t+phi) Sampling x(t) at a rate of fs (=1/Ts) results in a discrete time signal x[n]=A*cos(2*pi*f0*Ts*n+phi) If we sample the sinusoid over the time period 0 <= t <= T seconds, then we will acquire N = T*fs samples. The following Matlab script generates and plots a sampled signal:

```
clear all; close all;
A = 10;

f0 = 100;
phi = pi/4;
fs = 400;
Ts = 1/fs;
T = 1;
N = T*fs;
tn = (0:N-1)/fs;
x = A*cos(2*pi*f0*Ts*(0:N-1)+phi);

figure(1)
plot(tn,x)
% what is the frequency the sinusoid that was sampled?
fprintf('The frequency of the sinusoids that was sampled is %g Hz\n', f0)
% Was it adequately sampled at or above the Nyquist rate?
fprintf('because %g < fs/2 ( %g ), the sampling frequency is above Nyquist rate\n',
...
    f0, fs/2)
fprintf('Number of samples acquired per cycle = %g\n', fs/f0)

% The DFT of this signal an be coputed by simply implementing the
% mathematical formula X[k]=sum_n=0^N-1 x[n]*exp(-j*2*pi*k*n/N).
% Note that in Matlab, index starts at 1 rather than 0.

type mydft.m
% Create another function my_idft that takes X and synthesizes according to
% x[n]=(1/N)*sum_(k=0)^(N-1) X[k]exp(j*2*pi*k/N*n), n = 0, 1, ..., N-1.

type my_idft.m

% Compare the speed of your functions to the built-in Matlab functiosn
```

```
% using the script:
tic;

myX = mydft(x);
mytime = toc;
tic;
matX = fft(x);

mattime = toc;
fprintf('mytime = %g and mattime = %g\n',mytime, mattime)
% How does mytime compare to mattime? Which one is faster?
if mytime < mattime,
    fprintf('mydft runs faster\n')
else
    fprintf('fft runs faster\n')
end
```

The frequency of the sinusoids that was sampled is 100 Hz

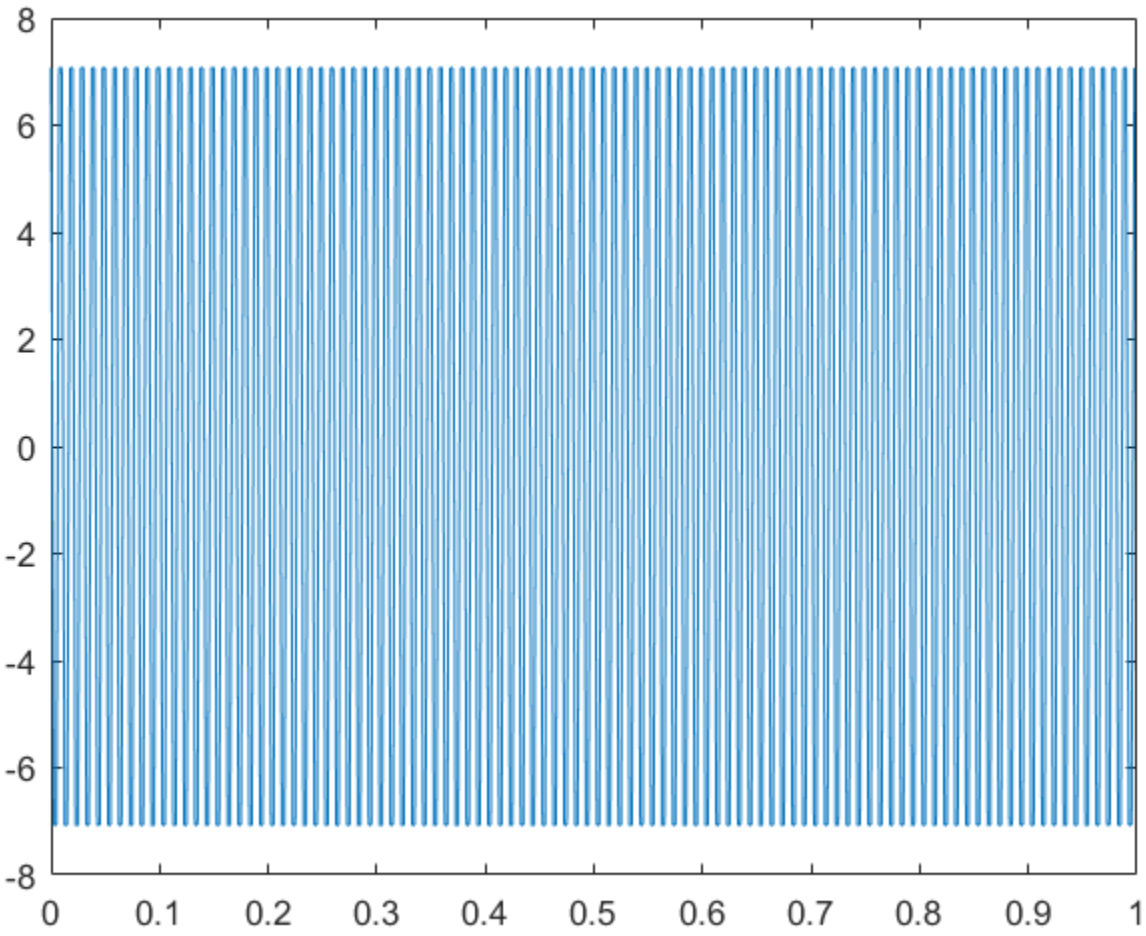
because 100 < fs/2 (200), the sampling frequency is above Nyquist rate

Number of samples acquired per cycle = 4

```
function X = mydft(x)

N=length(x);

for k=1:N

    X(k) = 0;

    for n = 1:N

        X(k) = X(k) + x(n)*exp(-j*2*pi*(k-1)/N*(n-1));

    end

end
```

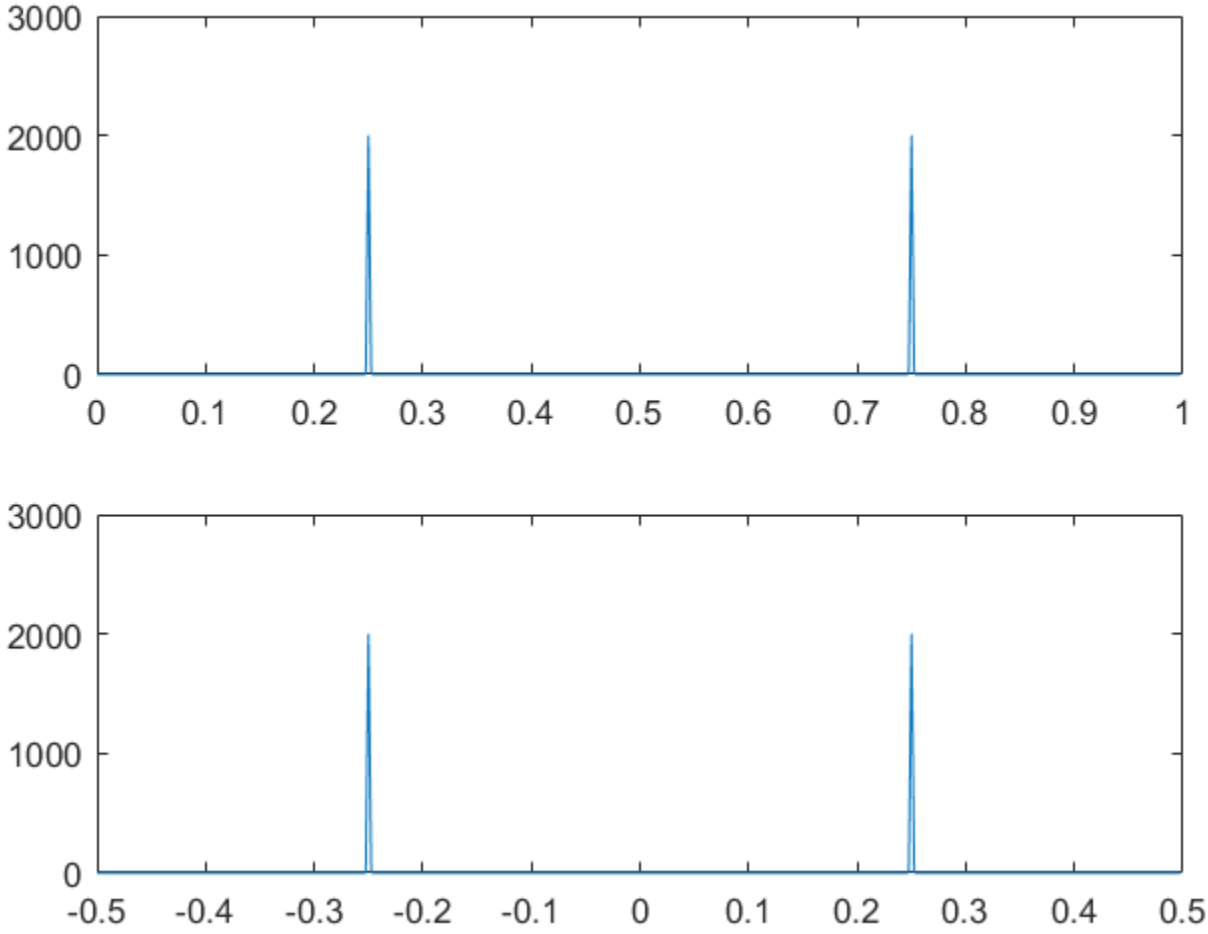
```
function X = my_idft(x)

N = length(x);

for n = 1:N
```

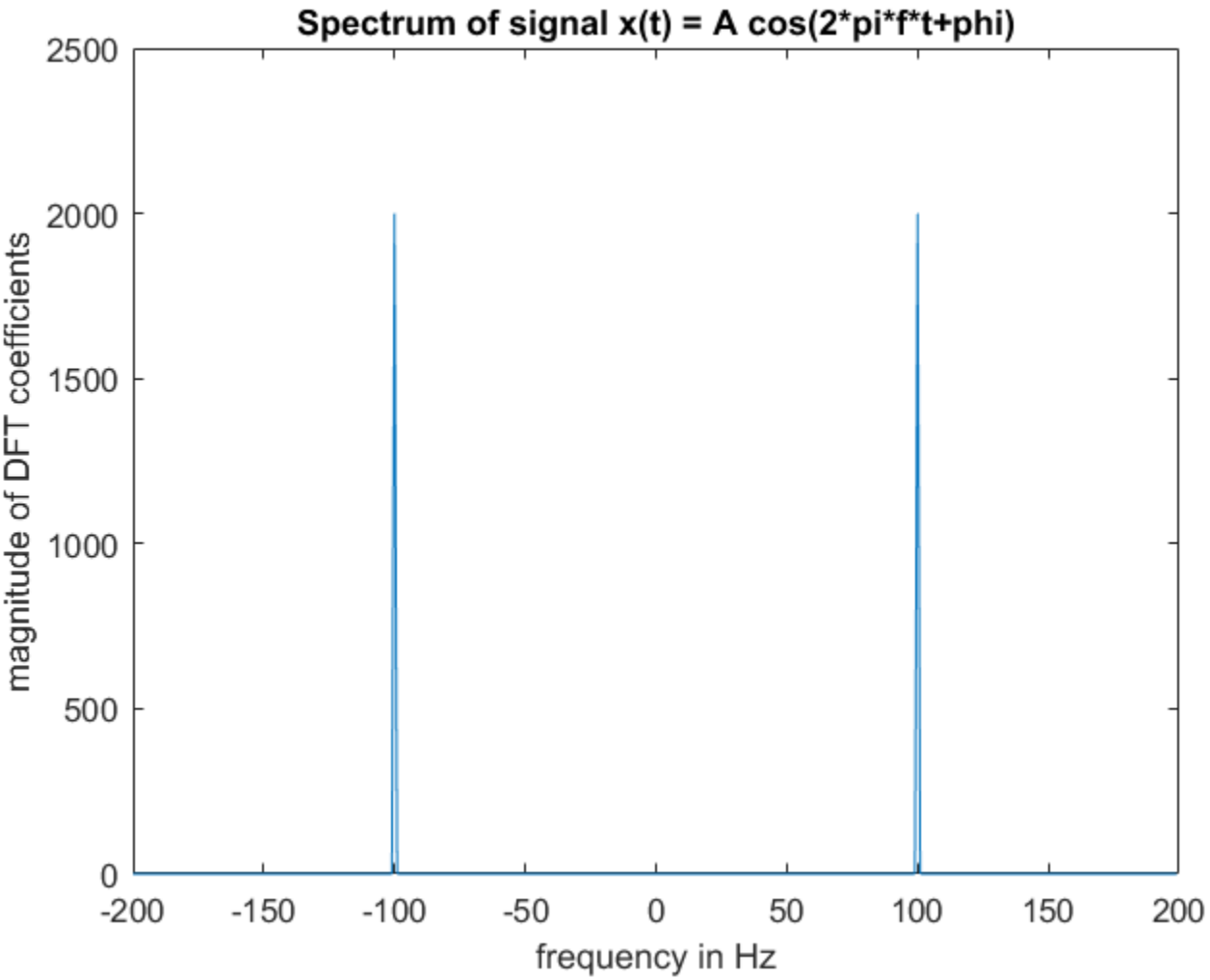
```
    X(n) = 0;
    for k = 1:N
        X(n) = X(n) + x(k)*exp(j*2*pi*(k-1)/N*(n-1));
    end
    X(n) = (1/N)*X(n);
end
mytime = 0.122353 and mattime = 0.000314909
fft runs faster
```

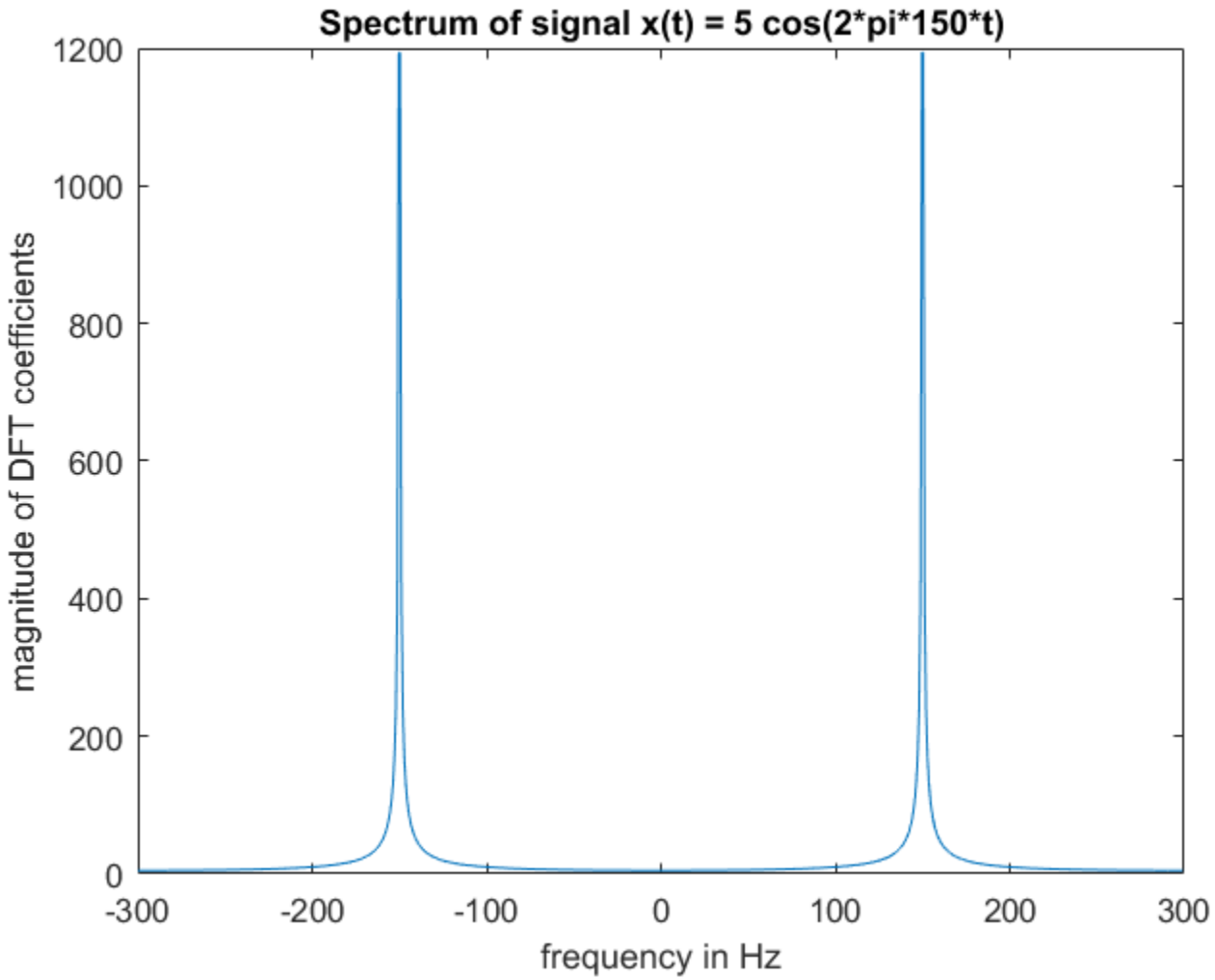


Section 3.2

```
% The following script show the Matlab command fftshift automatically wrap
% the DFT output from 1/2 to 1 to -1/2 to 0:
figure(2);

X = fft(x);
fhata = (0:N-1)/N;
subplot(2,1,1);
plot(fhata,abs(X));
fhatb = (-N/2:N/2-1)/N;

subplot(2,1,2);
plot(fhatb,fftshift(abs(X)));

% we can scale the frequency axis to convert from units of cycles/sample to
% cycles/second (Hz). This is done by scaling the frequency values from k/N
% with fs
figure(3);
fHz = (-N/2:N/2-1)/N*fs;
plot(fHz,fftshift(abs(X)));
xlabel('frequency in Hz');
ylabel('magnitude of DFT coefficients');
title('Spectrum of signal x(t) = A cos(2*pi*f*t+phi)');

% Now generate samples of a sinusoidal signal with A= 5, f= 150Hz, phi= 0,
% and fs= 600 over a time period of T= 1:25 seconds. How many samples are
% generated? Plot the spectrum of the signal. Is it what you expected?
A=5;
f=150; %Hz
phi = 0; % phase
fs=600; % Hz

Tx = 1/fs;
T = 1.25;
N = T*fs;
tn = (0:N-1)/fs;
x = A*cos(2*pi*f0*Ts*(0:N-1) + phi);

% Plots the second sinusoidal signal
X = fft(x);
fHz = (-N/2:N/2-1)/N*fs;
figure(4);
plot(fHz, fftshift(abs(X)));

xlabel('frequency in Hz');
ylabel('magnitude of DFT coefficients');
title('Spectrum of signal x(t) = 5 cos(2*pi*150*t)');
```


3000
2000
1000
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
3000
2000
1000
0
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5


Spectrum of signal x(t) = A cos(2*pi*f*t+phi)
magnitude of DFT coefficients
frequency in Hz
2500
2000
1500
1000
500
0
-200
-150
-100
-50
0
50
100
150
200



Section 4.1

Spectrum Analysis and Leakage Mathematically explain why the spectrum of the 150Hz sinusoid in the warm-up did not consist of two perfect peaks (see Section 13-5.5 in the textbook). This effect is often called spectral leakage".

```
% There is nothing to do in section 4.1
```

Section 4.2 Digital Music Equalizer

The DFT is the ideal tool for designing a digital music equalizer. We can boost (or suppress) various frequencies in a song using the DFT. The rap song rap.mat is in the folder "music clips" folder that you can download from the moodle site. The signal is 1048576 samples in length. The sampling rate is 44.1kHz. You can play the clip by executing the commands % load rap.mat followed by soundsc(x,fs) in Matlab.

```
load rap.mat; % variable name: rap
fs=44100; % sampling frquency 44.1 KHz

N = length(x); % should be 1048576
```

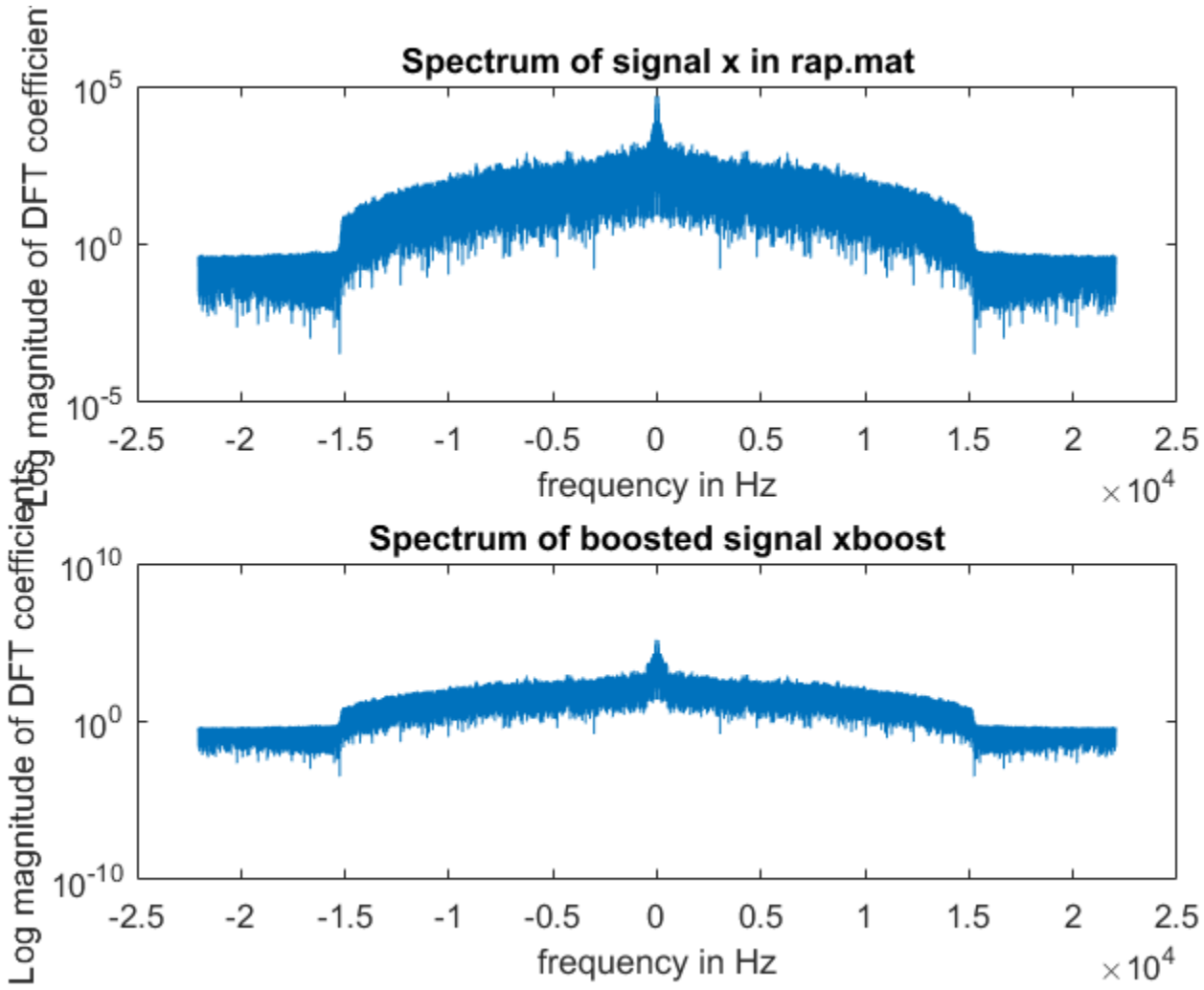
```
% soundsc(x,fs);

% 4.2.a Given the length and the sampling rate, how many seconds of
% the song were recorded?
% Answer: 1048576/44100 = 23.7772 seconds recorded

T = N/fs;

fprintf('Total duration of this music clip rap.mat is %g sec.\n',T)

% 4.2.b Use the Matlab command fft to compute the DFT of the signal and
% plot its spectrum.(plot the log of the magnitude of the DFT, using
% semilogy instead of plot, and scale the frequency axis to display Hz).

X = fft(x);
fHz = (-N/2:N/2-1)/N*fs;

subplot(2,1,1)
semilogy(fHz,fftshift(abs(X)));

xlabel('frequency in Hz');
ylabel('Log magnitude of DFT coefficients');
title('Spectrum of signal x in rap.mat');

% Do you observe a distinct drop in energy above a certain frequency?
% What would you suggest is the effective bandwidth (i.e., the maximum
% frequency) of the signal? Was it oversampled?
%
% Answer: Yes, the spectrum drops above 15 KHz. That would also be the
% bandwidth of the signal. Since fs = 44.1 KHz > 2*(15) KHz, this signal is
% sampled above the Nyquist rate (over-sampled).

% 4.2.c
% To "boost" the bass, amplify the low frequency components of the signal
% corresponding to frequencies at 500Hz and below by a factor of 3.
% Determine which DFT coefficients should be amplied given this specification.
% Note: you need to amplify both the positive and negative frequencies
% in order to preserve the symmetry of the DFT coefficients.
% After amplifying the low frequency DFT coe
```

```
cients, reconstruct a new
% signal using the ifft command. Listen to the boosted track and
% plot its spectrum to compare with the original spectrum plotted in b.
%
% first identify digital frequency k1/N corresponding to 500 Hz
fcutoff=500; % cutoff frequency for frequency boost
Ampfactor=3; % amplification factor

Xboost = X;

for k = 0:N-1;
    if (k*fs/N) <= fcutoff
        Xboost(k+1) = Xboost(k+1)*Ampfactor;
        Xboost(N-k) = Xboost(N-k)*Ampfactor;
    end
end

xboost = abs(ifft(Xboost));

% soundsc(xboost,fs);
subplot(2,1,2),
semilogy(fHz,fftshift(abs(Xboost)));
xlabel('frequency in Hz');
ylabel('Log magnitude of DFT coefficients');
title('Spectrum of boosted signal xboost');


% 4.2.d

type MEGAbass.m
```

```
Total duration of this music clip rap.mat is 23.7772 sec.
```

```
function xboost = MEGAbass(x,AmplificationFactor,BassCutoffFreq,fs)



xboost = x;

for k = 0:N-1;

    if (k*fs/N) <= fcutoff

        xboost(k+1) = xboost(k+1)*Ampfactor;

        xboost(N-k) = xboost(N-k)*Ampfactor;

    end
```

```
end
```



Published with MATLAB® R2016a