Part A

A.1

1) Plot of $x_1(t) = \cos(2\pi 550t)$:

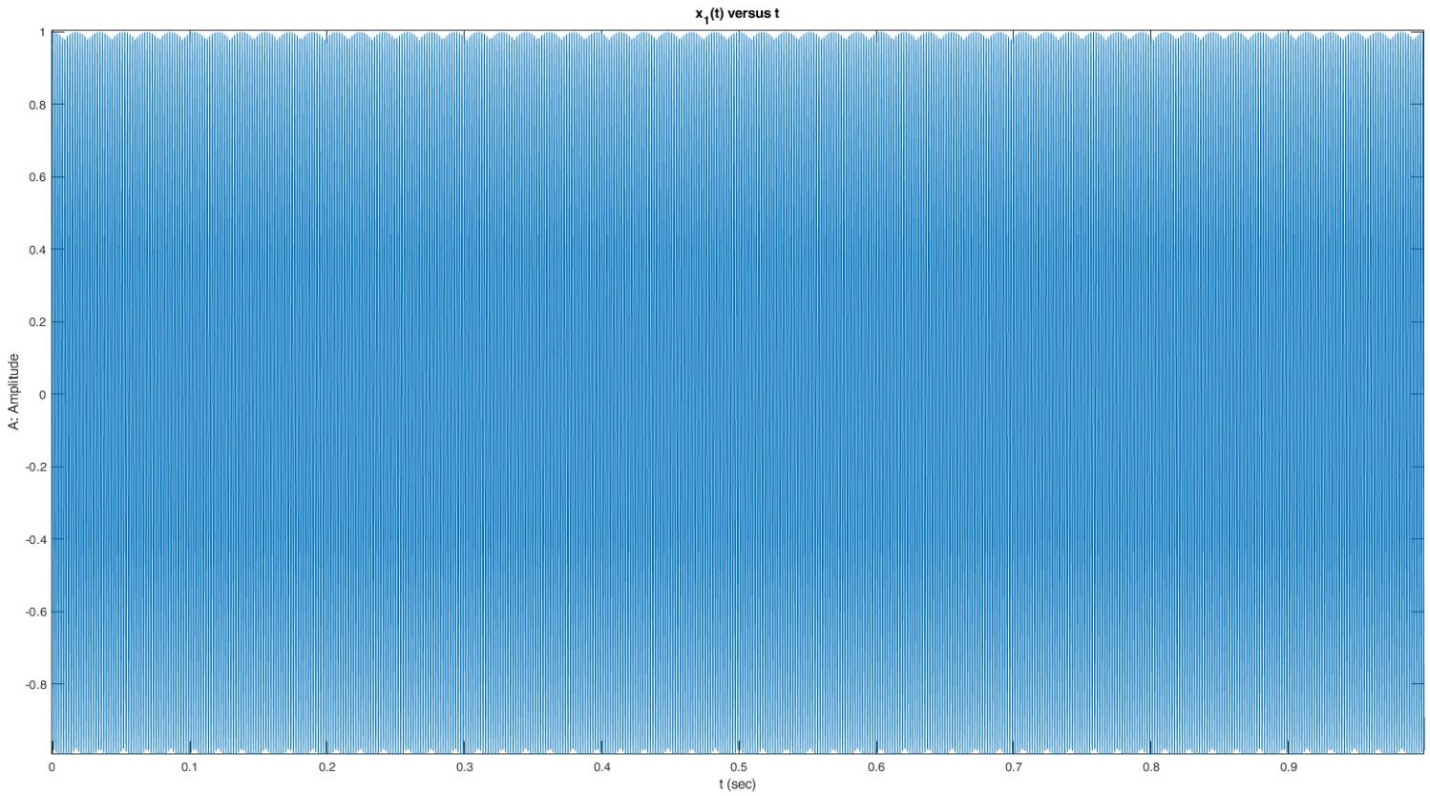

Figure 1: Plot of $x_1(t)$ versus t (sec)

It sounds like the sound you hear when you call a number and wait for someone to pick up.

2) When the frequency is decreased to 430 Hz, the pitch decreases.

3) When the frequency is increased to 750 Hz, the pitch increases.

4)
```
% Bilkent ID: 21901548

f1 = 154;
f2 = 190;
y = 2 * cos(2*pi*f1*t) + 3 * sin(2*pi*f2*t);
fmax = f2;

% plot of y
plot(t, y);
title('y(t) versus t');
xlabel('t: Time (sec)');
ylabel('Amplitude');
```

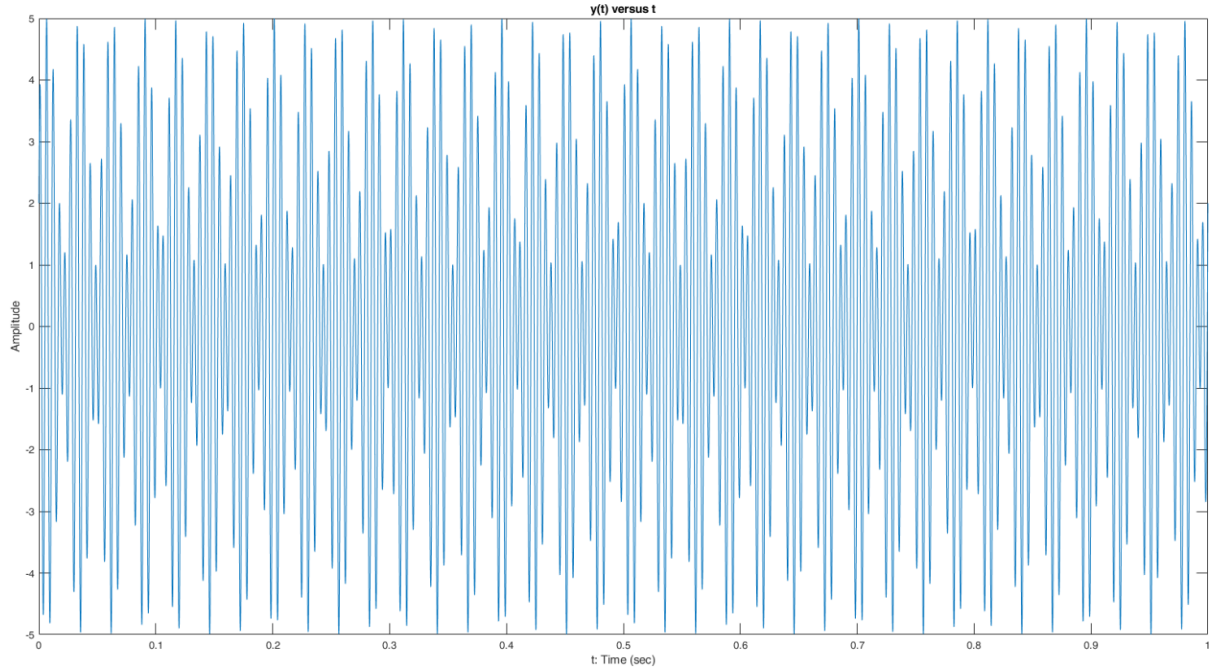

Figure 2: Plot of y(t) versus time

Sampling y(t) with rate fmax:

```
% sampling rate fmax
fs = fmax;
t1 = [0 : 1/fs : 1];
y1 = 2 * cos(2*pi*f1*t1) + 3 * sin(2*pi*f2*t1);
stem(t1, y1);
title('y_1(t) versus t');
xlabel('t: Time (sec)');
ylabel('Amplitude');
```

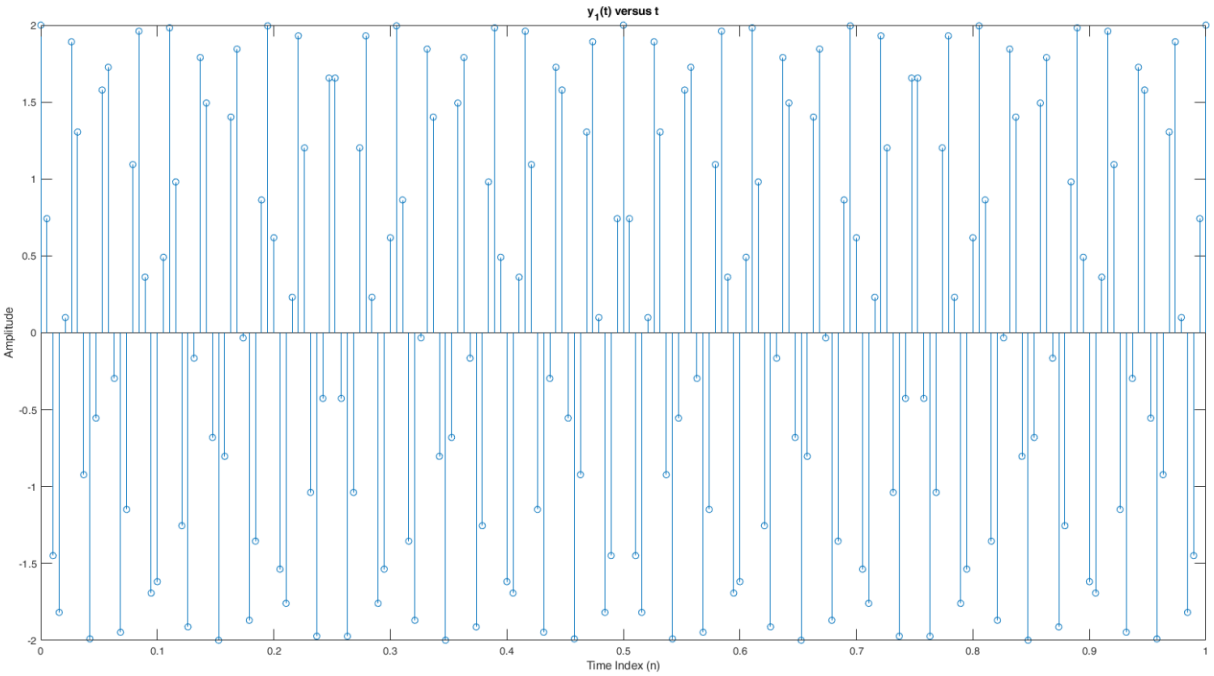

Figure 3: Plot of y(t) versus time sampled with f_s = fmax

Sampling y(t) with rate 2fmax:

```
% sampling rate 2fmax
fs = 2 * fmax;
t2 = [0 : 1/fs : 1];
y2 = 2 * cos(2*pi*f1*t2) + 3 * sin(2*pi*f2*t2);
stem(t2, y2);
title('y_2(t) versus t');
xlabel('t: Time (sec)');
ylabel('Amplitude');
```

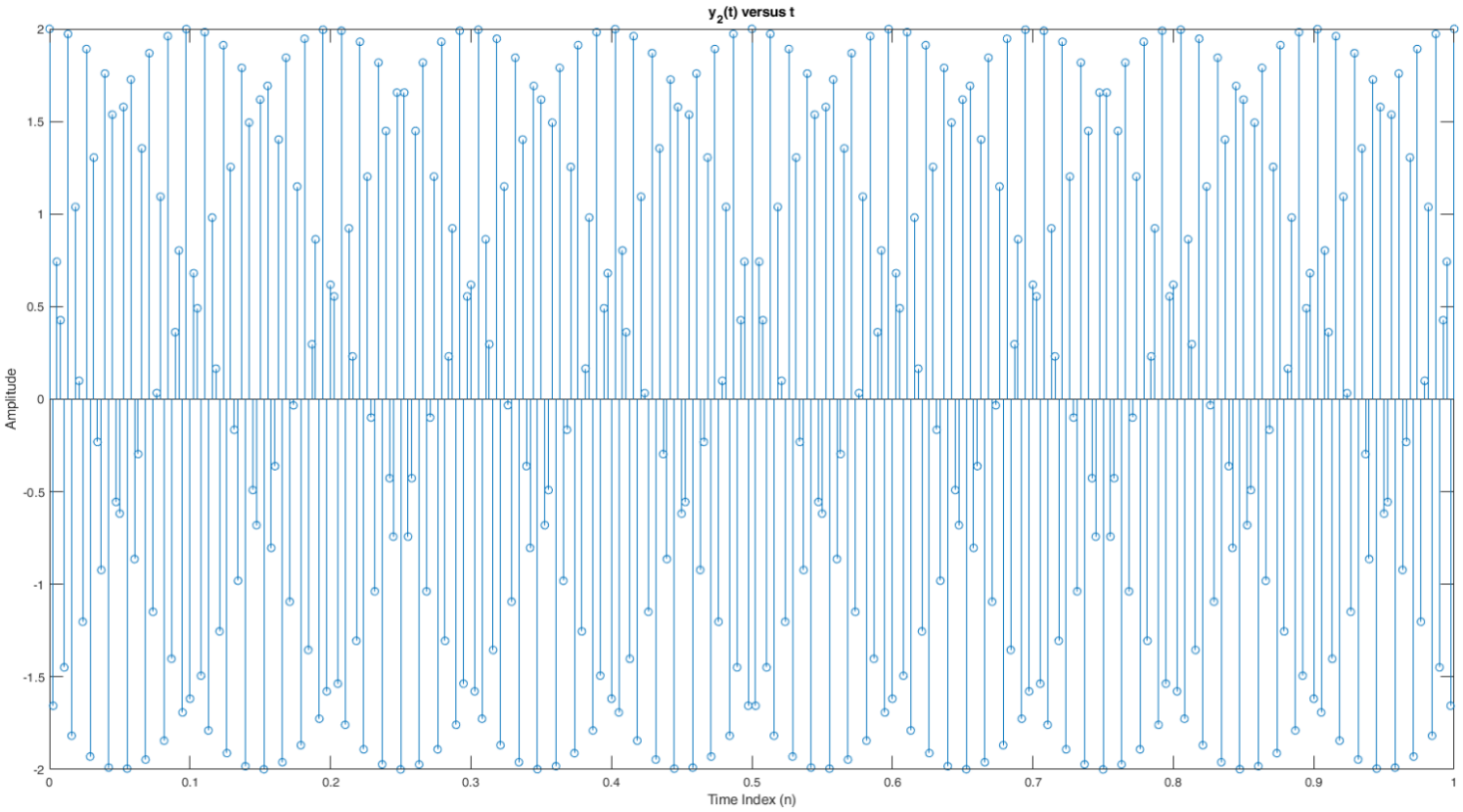

Figure 4: Plot of y(t) versus time sampled with fs = 2fmax

Sampling y(t) with rate 4fmax:

```
% sampling rate 4fmax
fs = 4 * fmax;
t3 = [0 : 1/fs : 1];
y3 = 2 * cos(2*pi*f1*t3) + 3 * sin(2*pi*f2*t3);
stem(t3, y3);
title('y_3(t) versus t');
xlabel('t: Time (sec)');
ylabel('Amplitude');
```

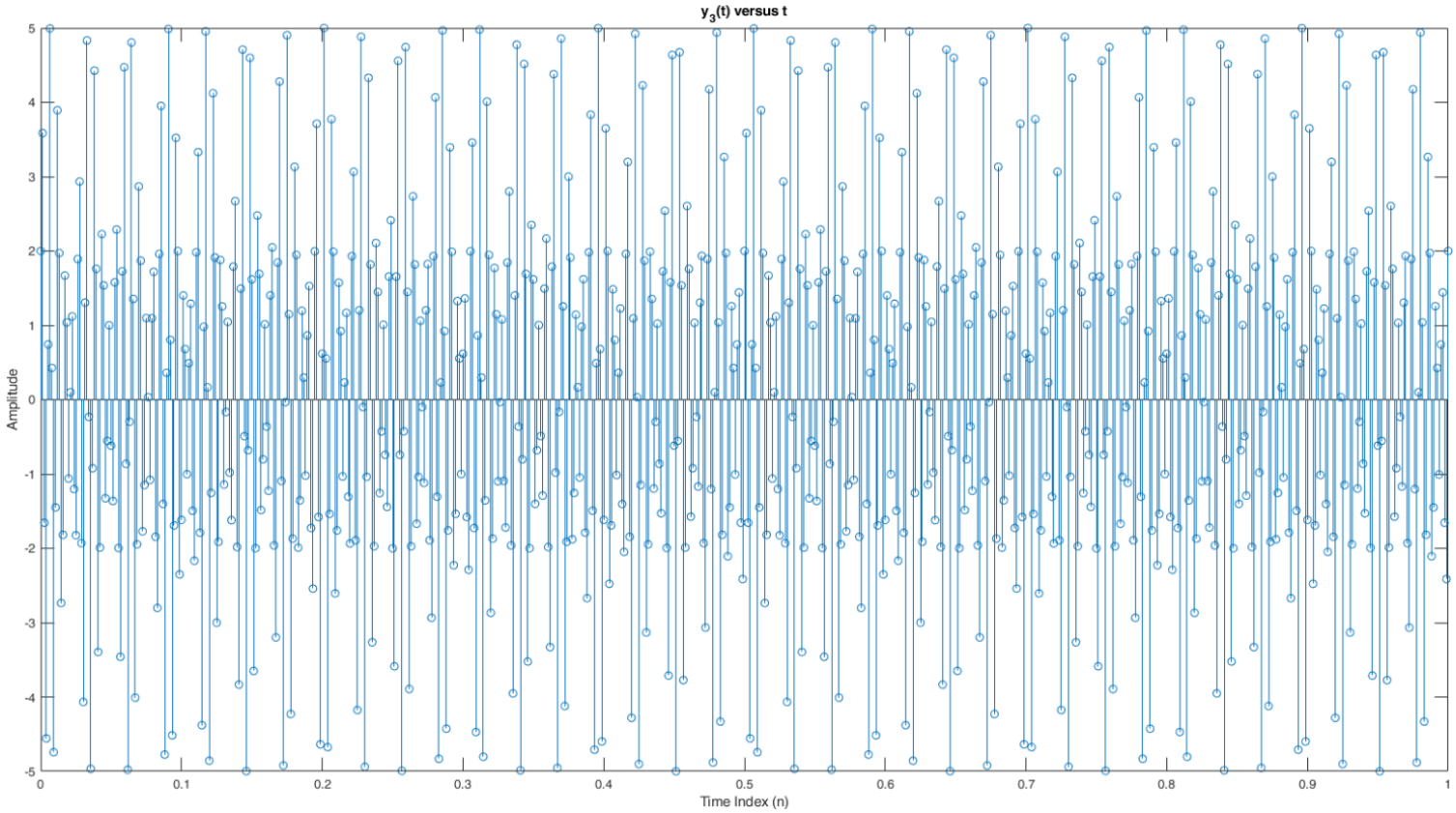

Figure 5: Plot of y(t) versus time sampled with fs = 4fmax

5)

The below method was used for reconstruction (I couldn't actually get this code to work very well, however what the output would look like is displayed below):

```
function[reconstructed] = reconstruction(sampledSignal, numberOfSamples, samplingPeriod, size)
    reconstructed = zeros(1, size);
    t = 0;
    for k = 1 : size
        for n = 1 : numberOfSamples
            reconstructed(1, k) = reconstructed(1, k) + sampledSignal(1, n) * (sin(pi * (t - n * samplingPeriod)) / samplingPeriod) / (pi * (t - n * samplingPeriod) / samplingPeriod);
        end
        t = t + 1/8192;
    end
end
% reconstructing signal (sampling rate fmax)
reconstructed = reconstruction(y1, size(y1, 2), 1 / fmax, size(t, 2));
plot(t1, y1, 'o', t,reconstructed);
title('y_1(t) reconstructed versus t');
xlabel('t: Time (sec)');
ylabel('Amplitude');
```

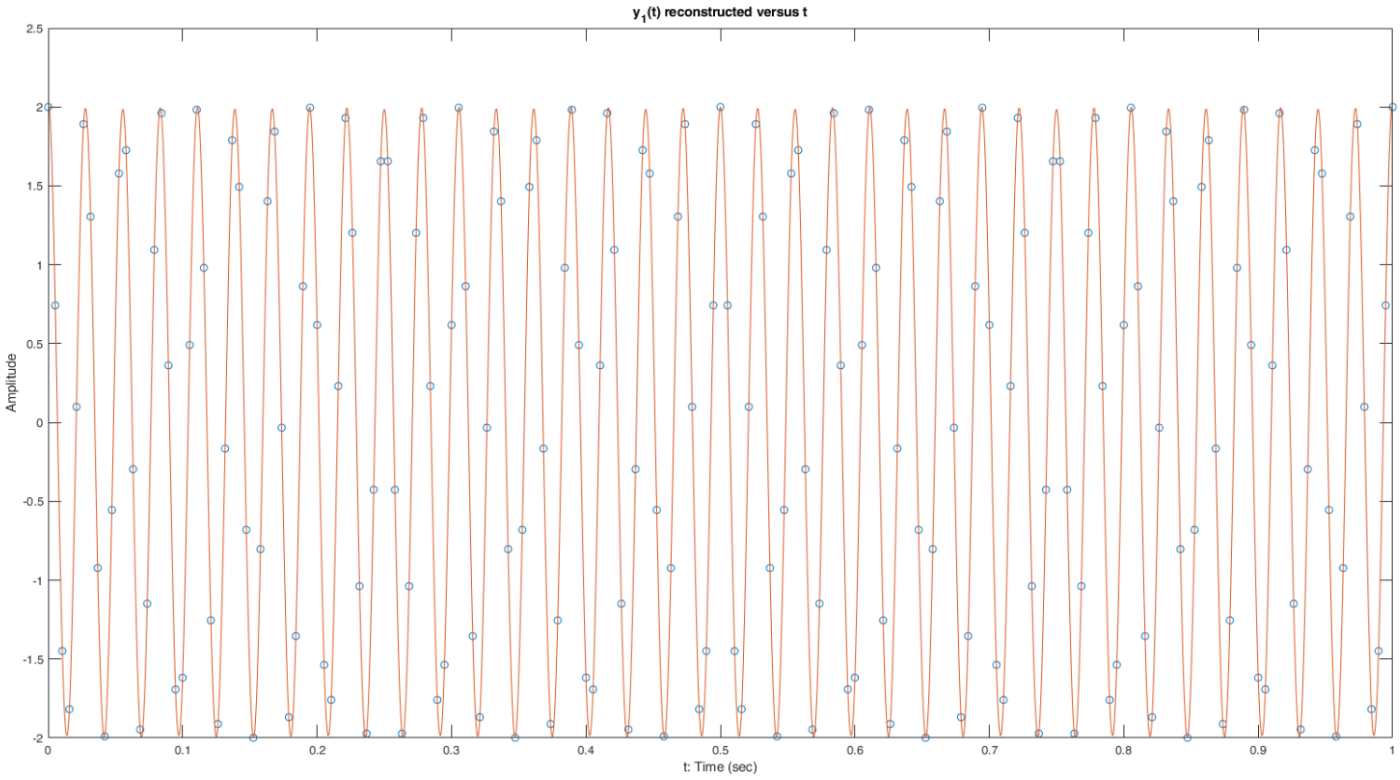

Figure 6: Plot of y(t) reconstructed from sample with rate fs=fmax

```
% reconstructing signal (sampling rate 2fmax)
reconstructed = reconstruction(y2, size(y2, 2), 1 / (2*fmax), size(t, 2));
plot(t2, y2, 'o', t, reconstructed);
title('y_2(t) reconstructed versus t');
xlabel('t: Time (sec)');
ylabel('Amplitude');
```

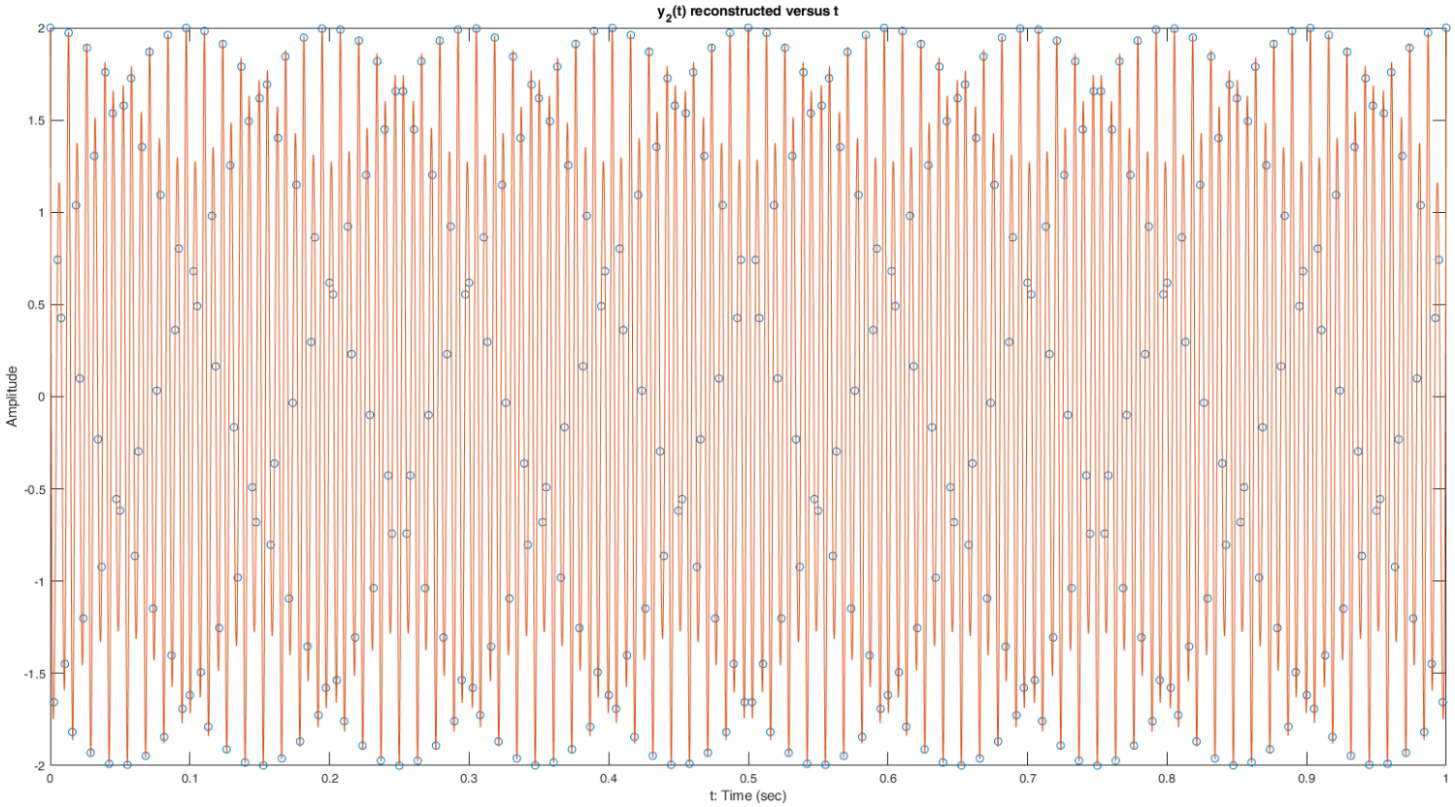

Figure 7: Plot of y(t) reconstructed from sample with rate fs=2fmax

```
% reconstructing signal (sampling rate 4fmax)
reconstructed = reconstruction(y3, size(y3, 2), 1 / (4*fmax), size(t, 2));
plot(t3, y3, 'o', t, reconstructed);
title('y_3(t) reconstructed versus t');
xlabel('t: Time (sec)');
ylabel('Amplitude');
```

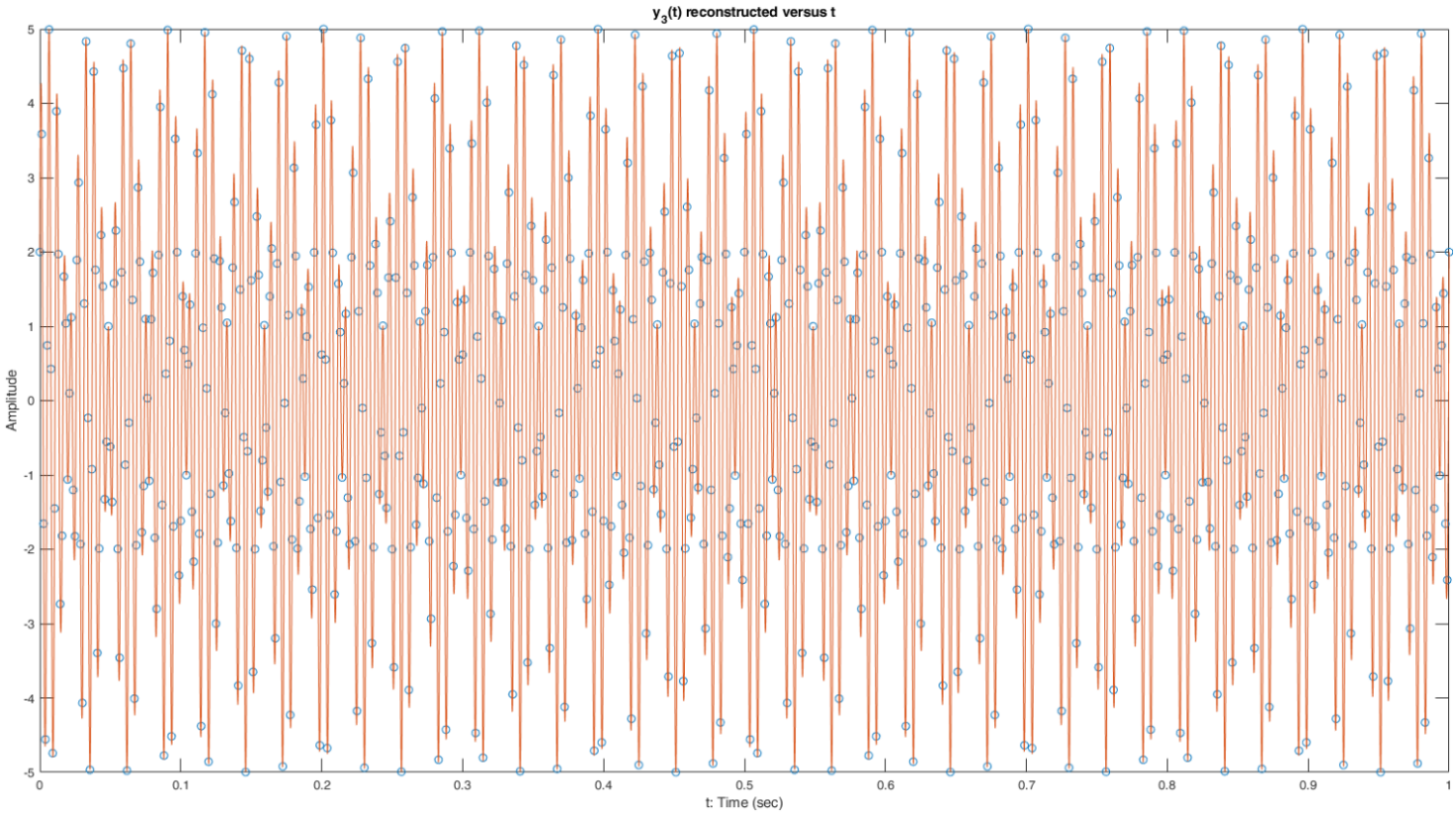

Figure 8: Plot of y(t) reconstructed from sample with rate fs=4fmax

As can be seen from the above graphs, the one that is most similar to the original signal is the last one, which is the reconstruction of the original signal from the sample which was sampled at the rate fs = 4fmax. The first one didn't yield a good result as it was sampled at half the Nyquist rate, at fs = fmax, thus it was undersampled. The last one was oversampled, at a rate of fs = 4fmax, which eliminated the problem of aliasing and yielded a good reconstruction of the signal. The second one, which is the reconstruction of the original signal from the samples taken at the borderline rate of fs = 2fmax should have theoretically been able to reconstruct the signal exactly, however it was insufficient at doing so. This could be due to finite precision issues of computers that result in numbers not being represented exactly and being approximated, or truncated to be able to store them. To eliminate this issue, I have tried sampling at the rate fs = 2.5fmax, which yielded a better result. You can see in the below plot the original signal and the reconstructed signal, however this sampling rate was still not enough to reconstruct the signal exactly.

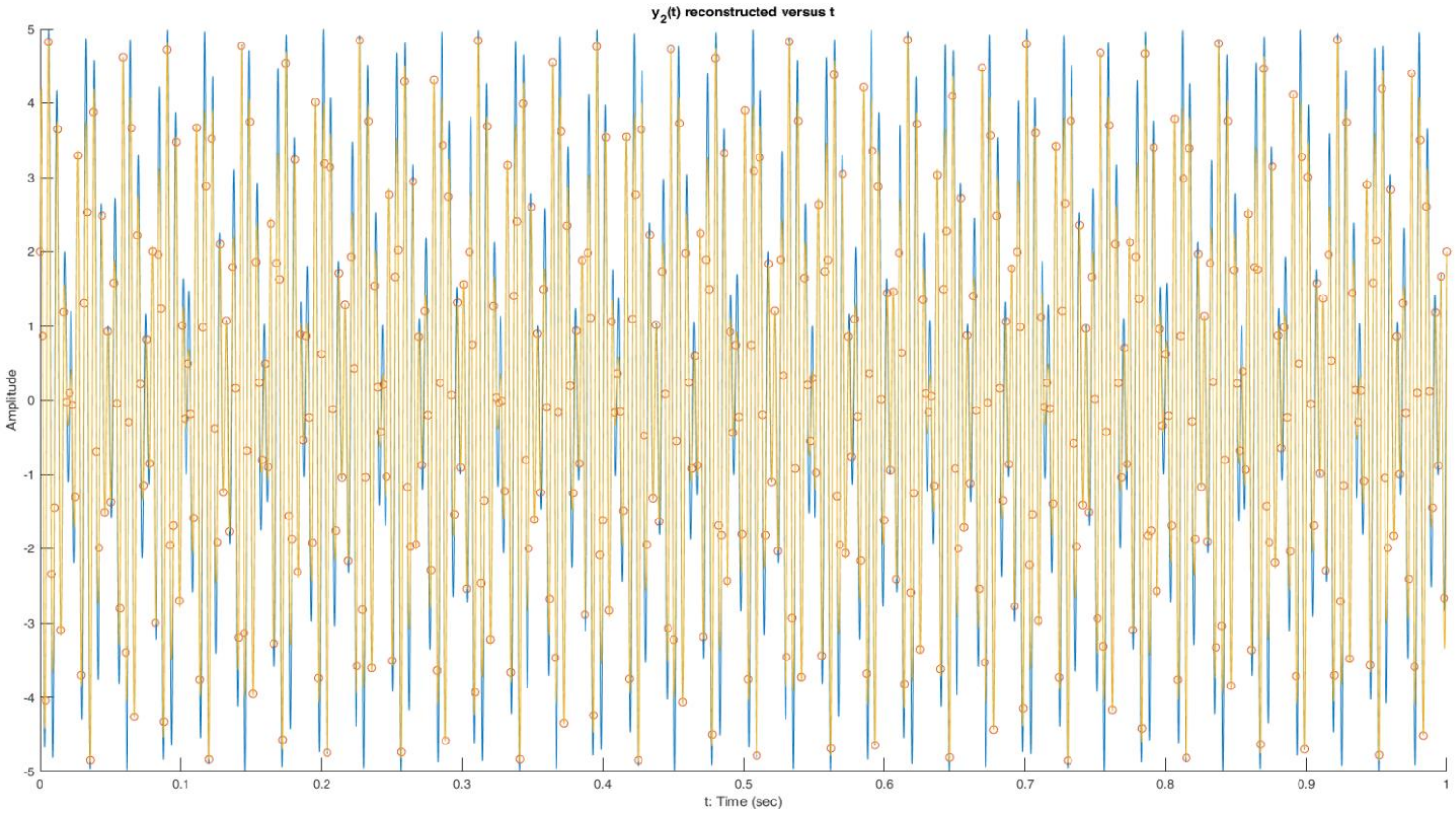

Figure 9: Plot of y(t) reconstructed from sample with rate fs=2.5fmax (original signal is in blue, reconstructed signal is in yellow)

A.2

1)

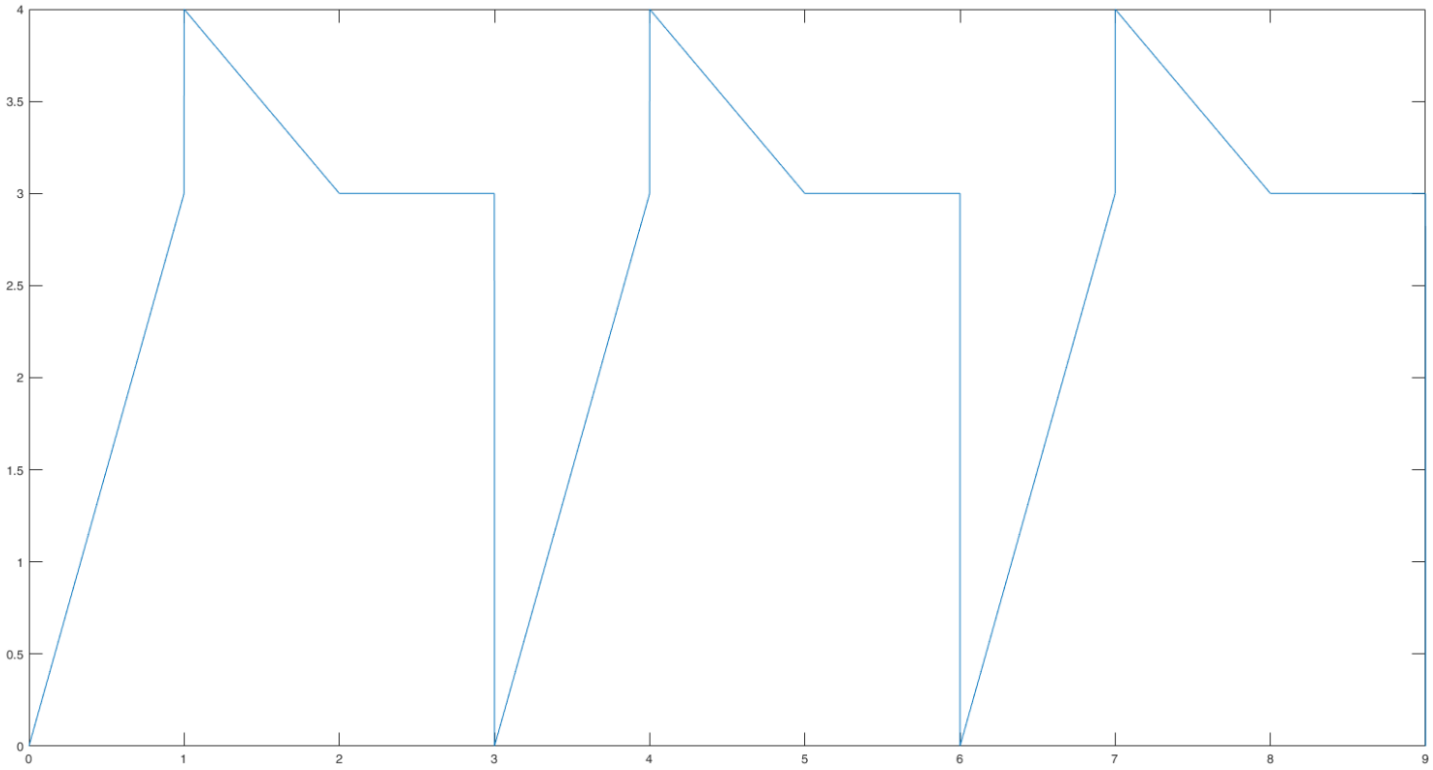

Figure 10: Plot of x(t) for 3 periods

2) I used Wolfram to compute the fourier series coefficients. I entered the below integral:

$$\int_0^1 3te^{i2\frac{pi}{3}kt}\,dt + \int_1^2 (5-t)e^{i2\frac{pi}{3}kt}\,dt + \int_2^3 3e^{i2\frac{pi}{3}kt}\,dt$$

The output (a_k) in 3 different forms:

Alternate forms

$$\frac{3\left(-6i\pi e^{2i\pi k}k - 3e^{(4i\pi k)/3} + 2e^{(2i\pi k)/3}(6+i\pi k) - 9\right)}{4\pi^2 k^2}$$

$$\frac{3\left(2i\pi e^{(2i\pi k)/3}k - 6i\pi e^{2i\pi k}k + 12e^{(2i\pi k)/3} - 3e^{(4i\pi k)/3} - 9\right)}{4\pi^2 k^2}$$

$$\frac{1}{4\pi^2 k^2}3\left(2i\pi e^{(2i\pi k)/3}k - 6i\pi e^{(4i\pi k)/3}k + 12e^{(2i\pi k)/3} - 3e^{(4i\pi k)/3} + 12\pi e^{(5i\pi k)/3}k\sin\left(\frac{\pi k}{3}\right) - 9\right)$$

3) I have used a Matlab code to compute the total energy, which I found to be 9. In this case, the numerator must be 9 * 0.95 = 8.55. I used the same Matlab code to compute N where this answer is reached. I found that N must be equal to 10, in which case the numerator becomes 8.56, close enough to what I calculated. The Matlab code I used is given below:

```
y = @(k) (3 * (-6*1i*pi*exp(2*1i*pi*k)*k - 3*exp(4*1i*pi*k/3) +
2*exp(2*1i*pi*k/3)*(6+1i*pi*k)-9) / (4*pi.^2*k.^2));

sum = 0;

for ind = -10 : 10
    if ~isnan(y(ind))
        sum = sum + abs(y(ind)).^2;
    end
end
```

4)

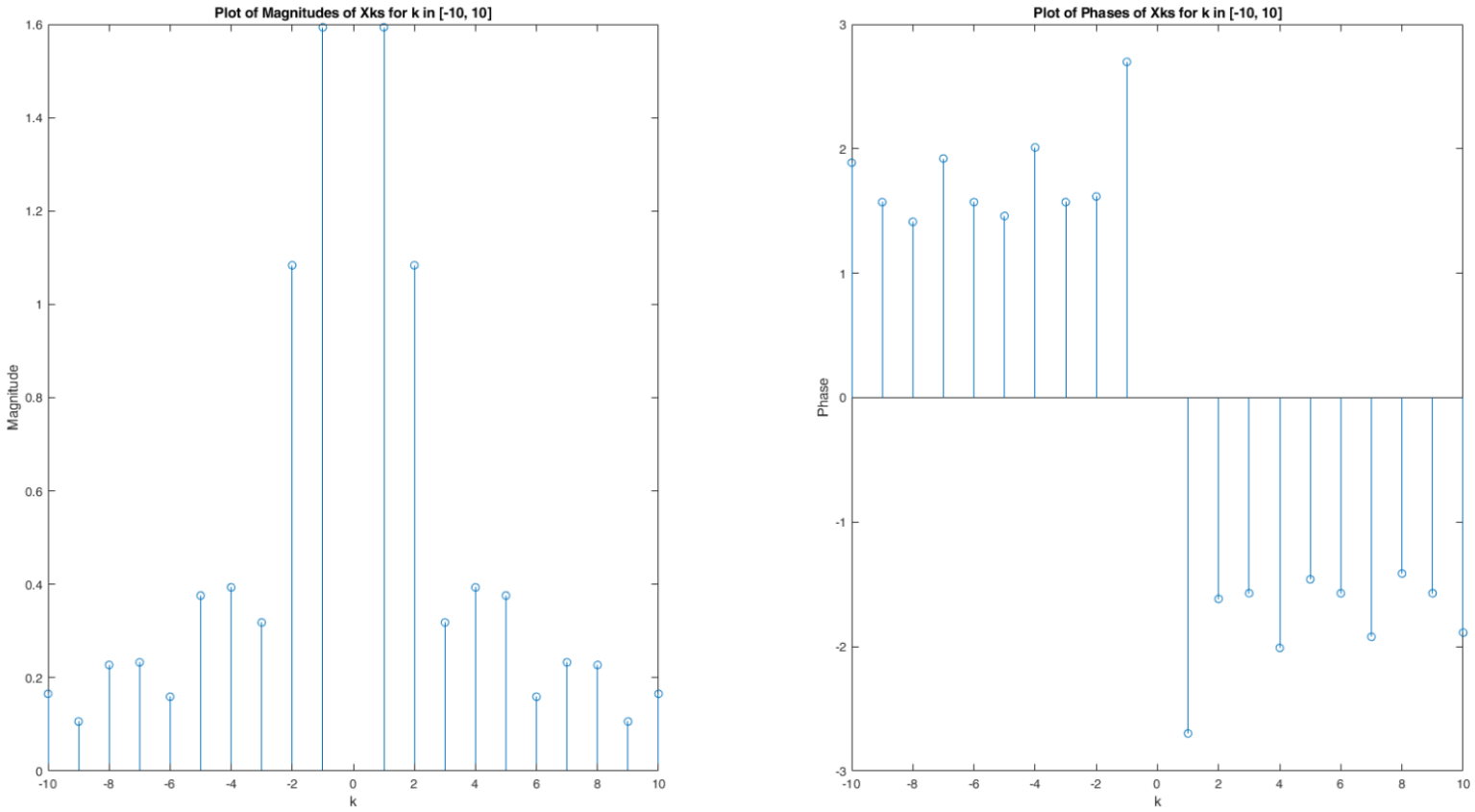

Figure 11: Magnitude and Phase plots

Part B

1) I have used the Virtual Piano site for recording.

2) In the recording, the F3 note is played. According to given site on the homework document, this note has a frequency of 174.61 Hz, so the fundamental period must be 1/174.61 = 0.005727048852 seconds.

I have extracted a portion of soundArray that includes about a 100 periods of the original signal to create partOfSoundArray. Both of the plots for these signals can be seen below.

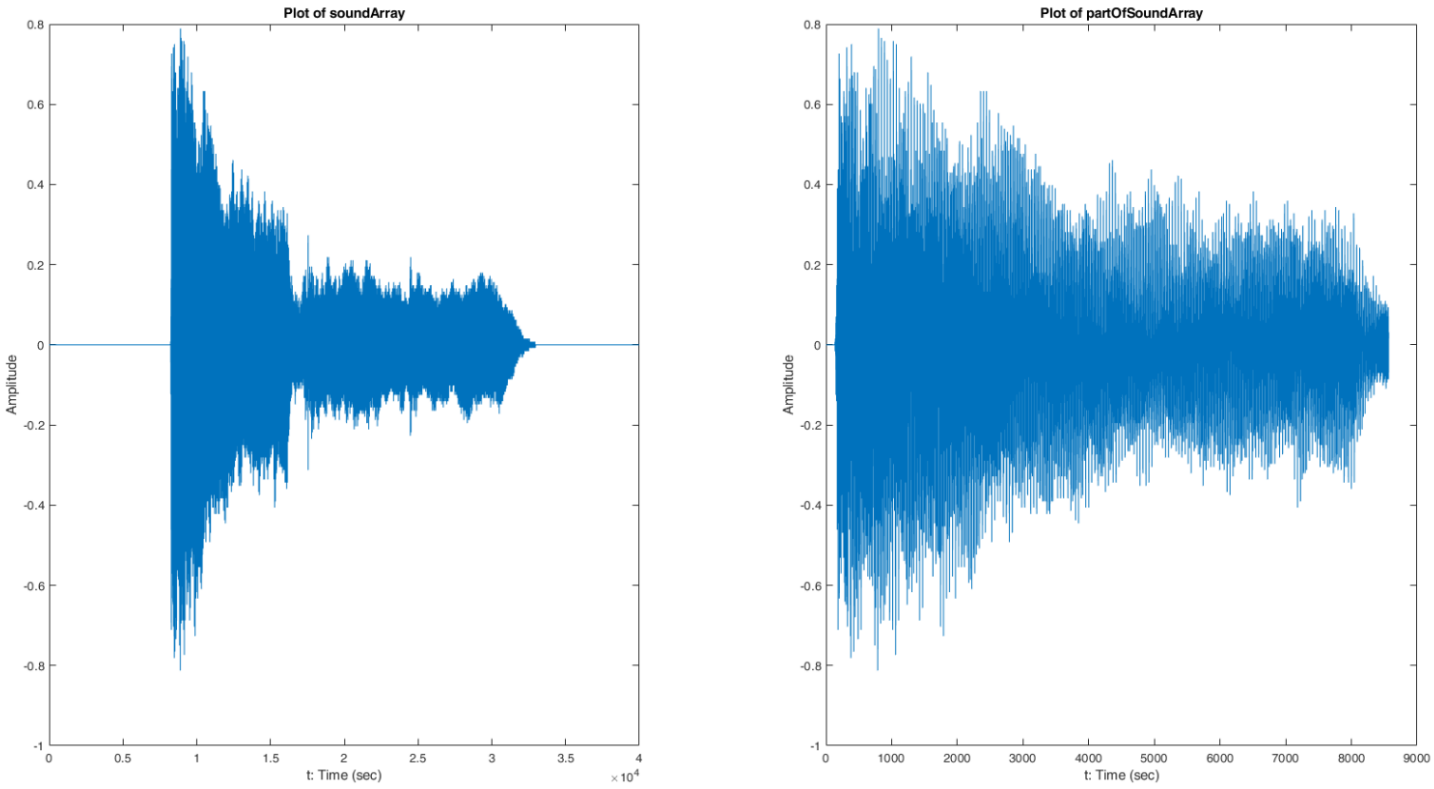

Figure 12: Plots of soundArray and partOfSoundArray

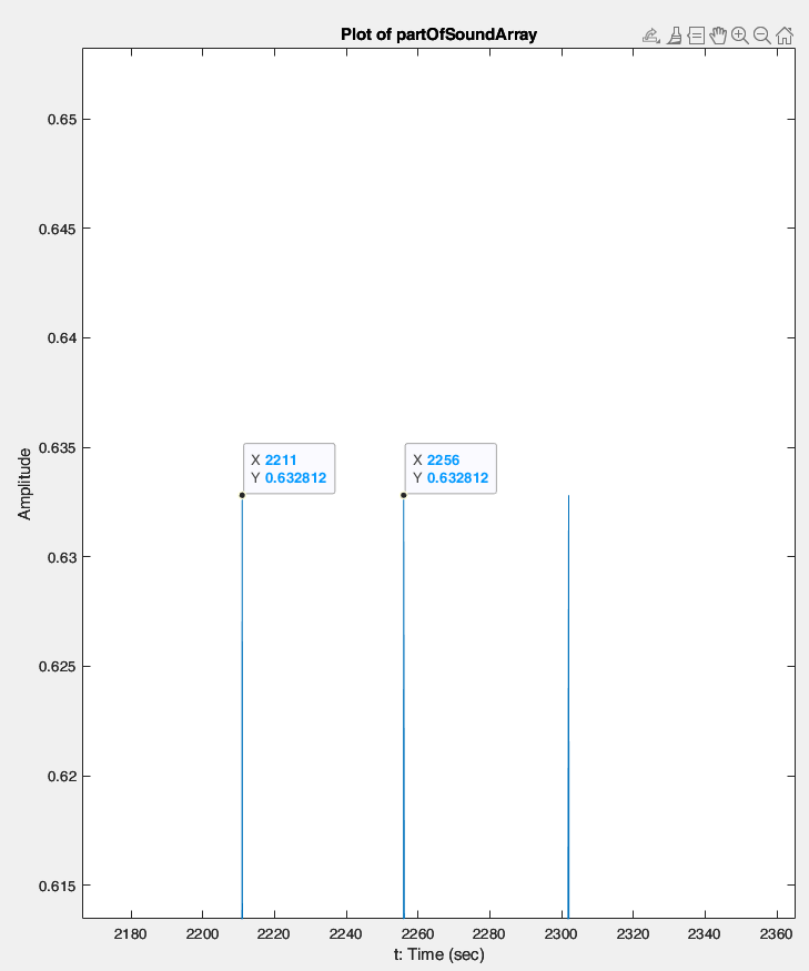

Figure 13: Zoomed partOfSoundArray

Figure 13 shows a section of partOfSoundArray. As can be seen, there are 2256 – 2211 = 45 samples between the two peaks. The sampling frequency is 8 kHz, so the sampling period is 1 / 8000 = 0.000125 seconds. This means, between these two peaks there are 45 * 0.000125 = 0.0056 seconds. I had calculated the period of this note to be 0.0057 seconds, so my analytical calculation is justified with my observation.

3)

```
f = 174;
w = 2*pi*f;
N = 370;
result = fs(firstPeriodOfPartOfSoundArray, w, N);
stem(-N:N, result);
title('Fourier Coefficients');
xlabel('k');
ylabel('ak');
function[ak] = fs(sound, w, N)
   ak = zeros(1,2 * N+1);
   f = w/(2*pi);
   T = 1 / f;
   t = 0 : 1/8000 : T;
   t = t(1:45);
   for k = -N : N
       ak(1, k + N + 1) = f * real( trapz(( exp( -1i * k * w *
t)).*(sound.')));
   end
end
```

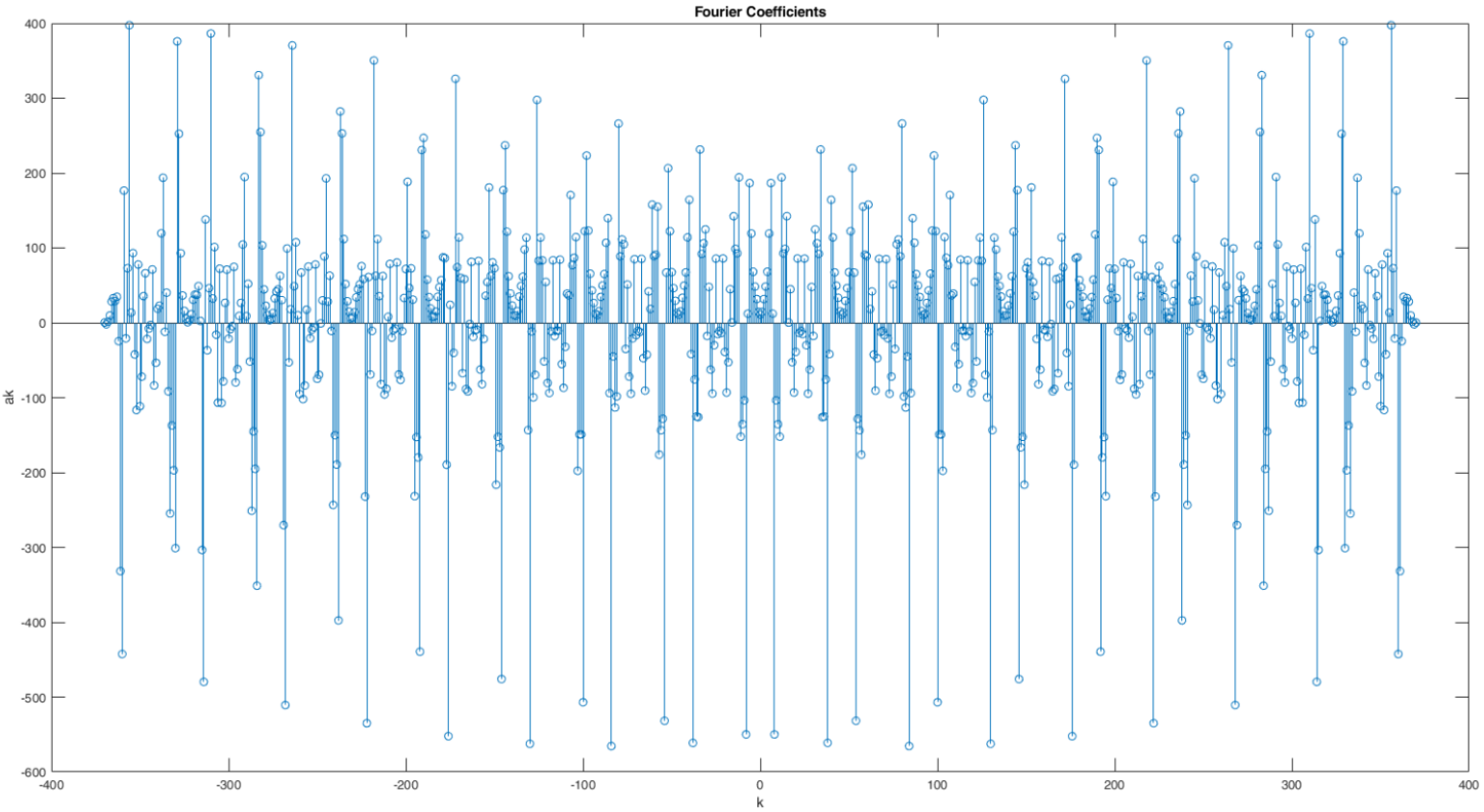

Figure 14: Fourier Series Coefficients of Sound Signal

4)

I have recorded the F3 note on a guitar and a bass guitar (both provided by my friend), in addition to the piano. The fundamental period is again 1/174.61 = 0.005727048852 seconds.

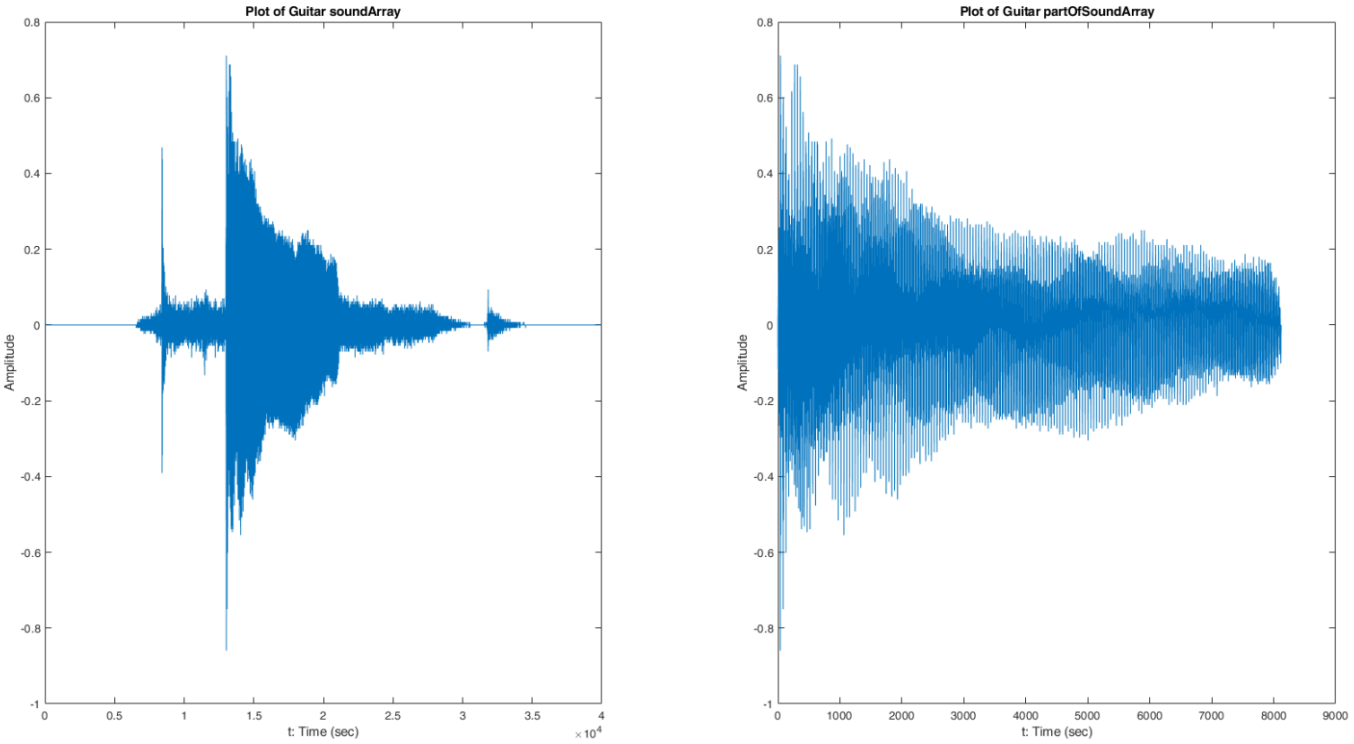

Figure 15: Plots of soundArray and partOfSoundArray for Guitar

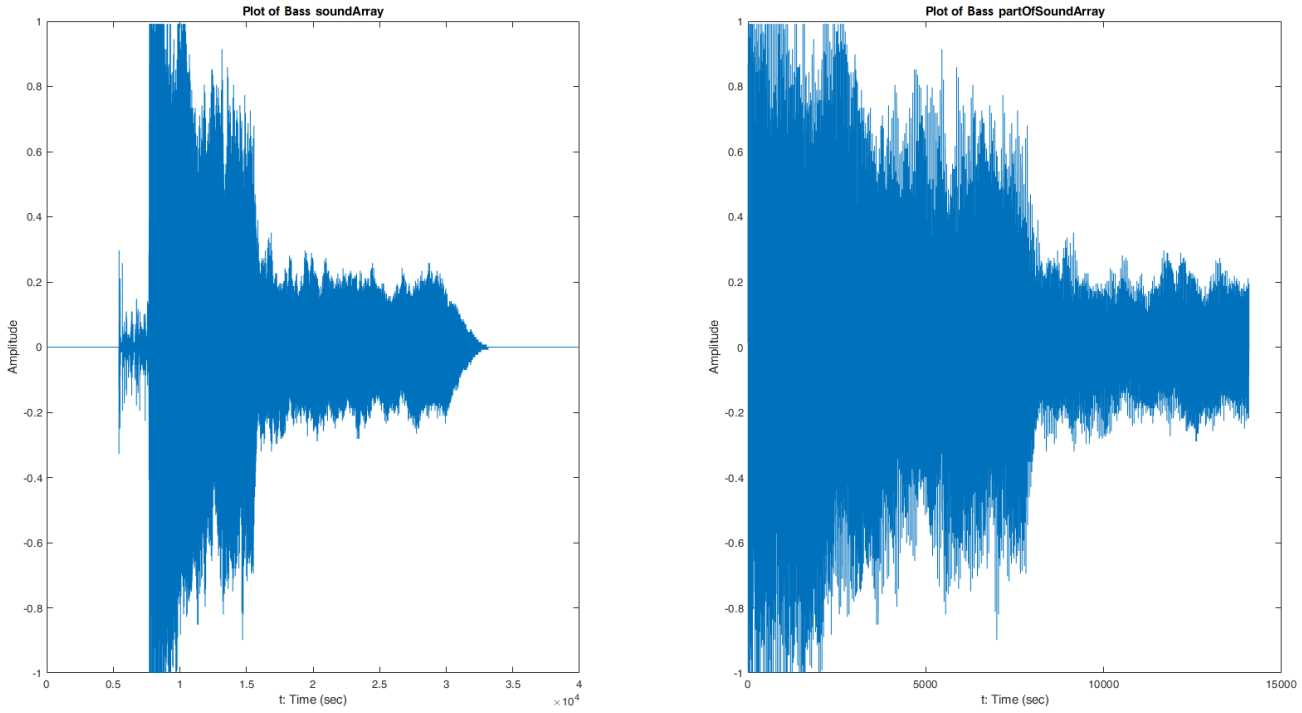

Figure 16: Plots of soundArray and partOfSoundArray for Bass Guitar

When we observe these graphs, we can see the analytical calculation for the fundamental period is justified, as in the previous case for the piano.

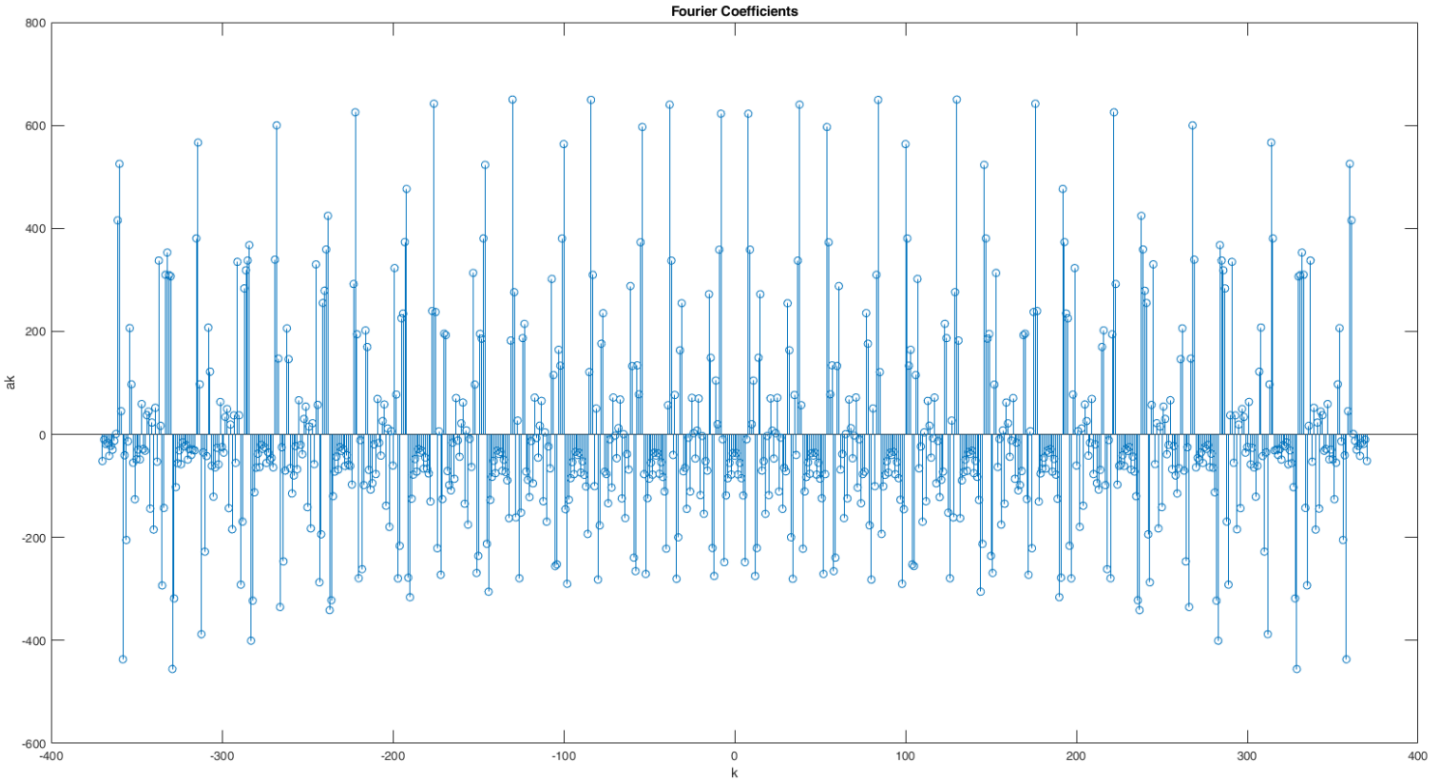

Figure 17: Guitar FSCs

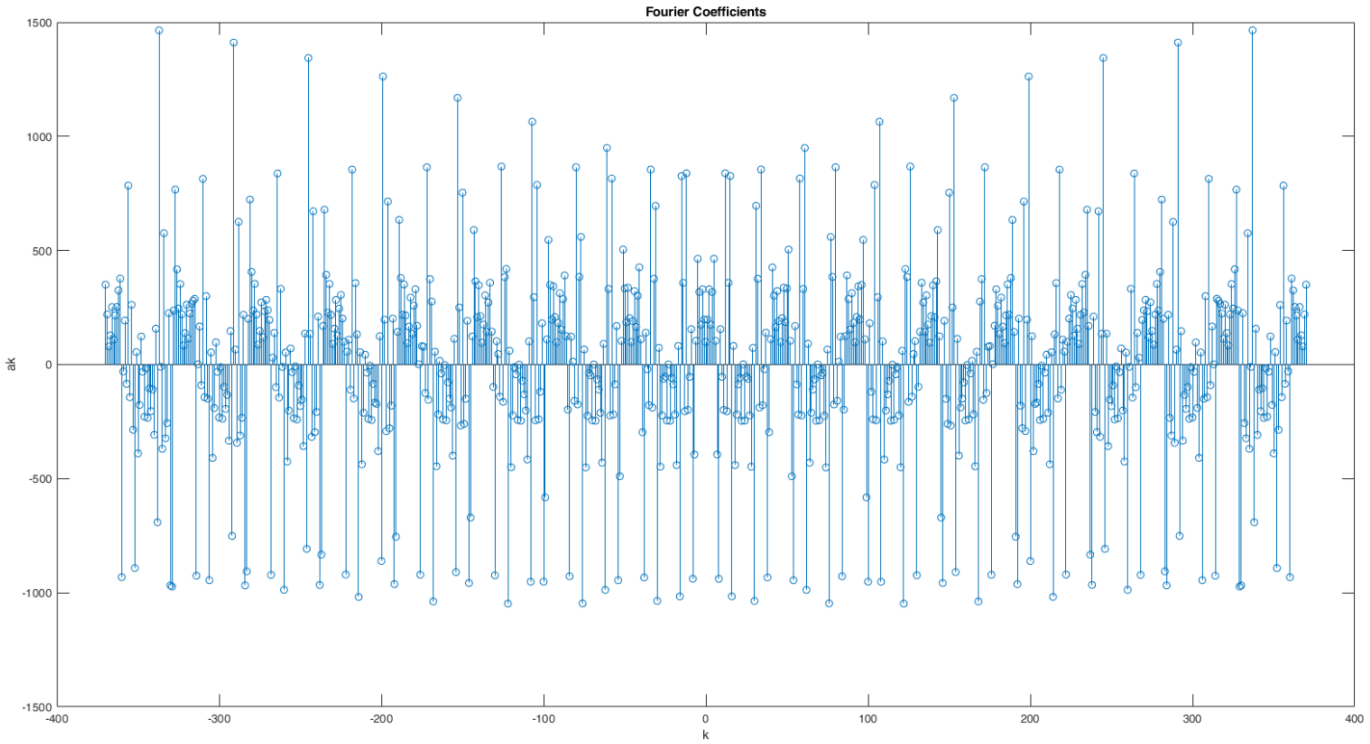

Figure 18: Bass Guitar FSCs

The FSCs computed for the piano, guitar and the bass seem mostly the same in terms of the shapes of the graphs, however the values differ between them for some k's. This could be due to the errors in recording the sound, such as crackling noise etc., as well as the fact that I have used virtual sounds for the piano, but a real guitar and bass for the guitar sound.

```
% PART B Question 4

% F3 note on guitar
guitarSoundArray = load("GuitarF3soundArray.mat").soundArray;
guitarRecObj = load("GuitarF3recObj.mat").recObj;
guitarPartOfSoundArray = guitarSoundArray(12991 : 21110);
% subplot(1,2,1);
plot(guitarSoundArray);
title('Plot of Guitar soundArray');
xlabel('t: Time (sec)');
ylabel('Amplitude');

% subplot(1,2,2);
plot(guitarPartOfSoundArray);
title('Plot of Guitar partOfSoundArray');
xlabel('t: Time (sec)');
ylabel('Amplitude');

% F3 note on bass
bassSoundArray = load("BassF3soundArray.mat").soundArray;
bassRecObj = load("BassF3recObj.mat").recObj;
bassPartOfSoundArray = bassSoundArray(7710 : 21796);
% subplot(1,2,1);
plot(bassSoundArray);
title('Plot of Bass soundArray');
xlabel('t: Time (sec)');
ylabel('Amplitude');

% subplot(1,2,2);
plot(bassPartOfSoundArray);
title('Plot of Bass partOfSoundArray');
xlabel('t: Time (sec)');
ylabel('Amplitude');

% FSC calculation
firstPeriodOfPartOfSoundArray = guitarPartOfSoundArray(1 : 45);
result = fs(firstPeriodOfPartOfSoundArray, w, N);
stem(-N:N, result);
title('Fourier Coefficients');
xlabel('k');
ylabel('ak');

firstPeriodOfPartOfSoundArray = bassPartOfSoundArray(1 : 45);
result = fs(firstPeriodOfPartOfSoundArray, w, N);
stem(-N:N, result);
title('Fourier Coefficients');
xlabel('k');
ylabel('ak');
```

5)

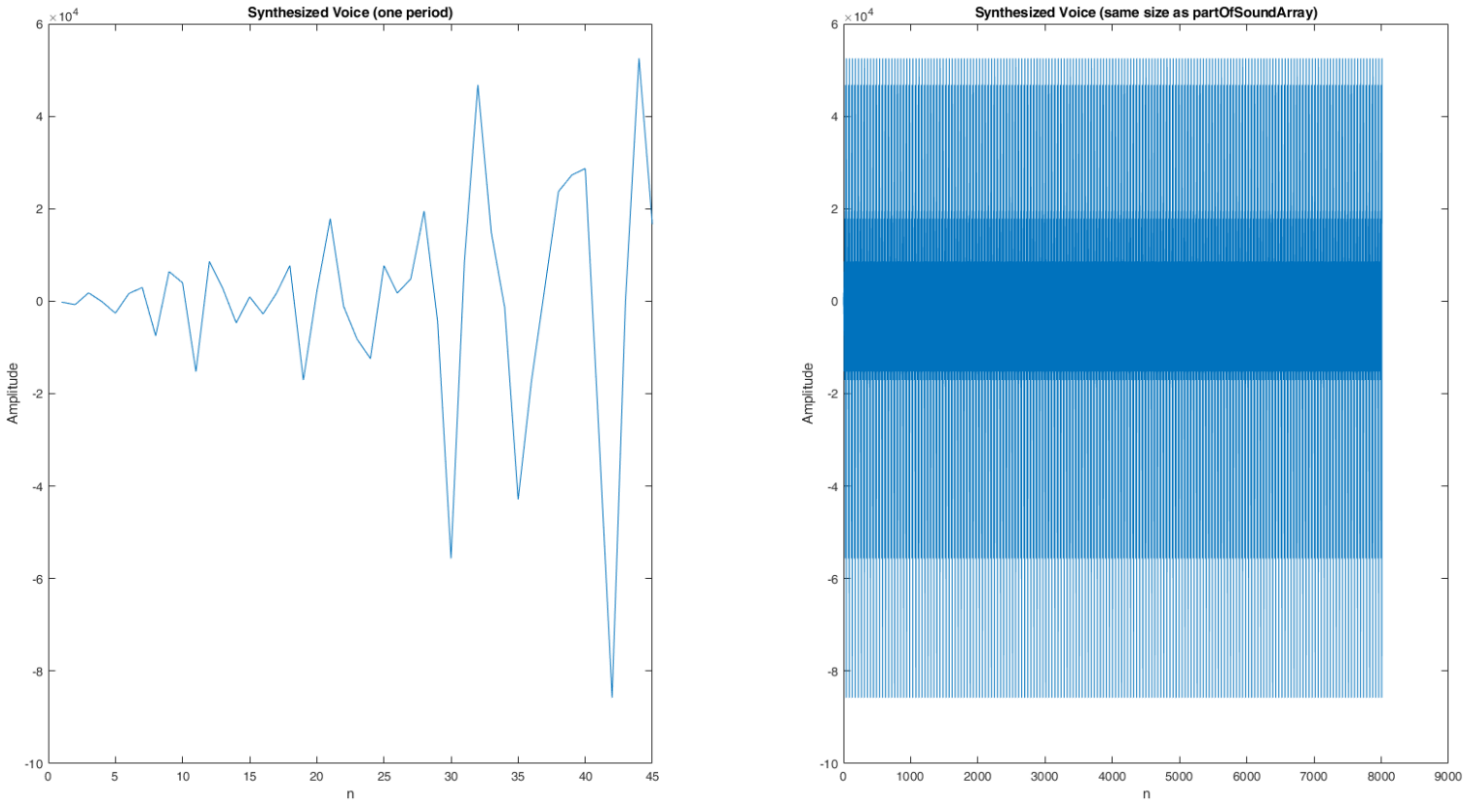

Figure 19: Synthesized voice plots

```
% function for synthesis
function[onePeriod] = synthesis(coefficients, w, N)
    f = w/(2*pi);
    T = 1 / f;
    t = 0 : 1/8000 : T;
    t = t(1:45);
    onePeriod = zeros(1, size(t, 2));

    for j = 1 : 45
        for k = -N : N
            onePeriod(1, j) = onePeriod(1, j) + coefficients(1, k + N + 1)
* exp( 1i * k * w * t(1, j));
        end
    end
end

% Part B Question 5
synthesized = synthesis(resultPiano, w, N);
subplot(1,2,1);
plot(real(synthesized));
title('Synthesized Voice (one period)');
xlabel('n');
ylabel('Amplitude');
% extending the array periodically
```

```
synthesized = repmat(synthesized, 178, 1);
synthesized = reshape(synthesized.',1,[]);
subplot(1,2,2);
plot(real(synthesized));
title('Synthesized Voice (same size as partOfSoundArray)');
xlabel('n');
ylabel('Amplitude');
audiowrite('handel.wav', real(synthesized), 8000);
```

6) This piece of code was used to generate the sound file:

```
audiowrite('handel.wav', real(synthesized), 8000);
```

The signal I hear is similar to the sound I had recorded, only with the difference of the timbre between two sounds. This shows that the synthesis operation was mostly successful. The difference of timbre between sounds could be due to the extension of the same period of the sound when resynthesizing.

7)

```
N1 = 1;
M = 164;
N2 = 2;
for k = 1 : size(resultPiano, 2)
    if resultPiano(1, k) > 0.0001
        N1 = k;
        break;
    end
end

synthesized = synthesis(resultPiano, w, (N2 - N1) / 2);
synthesized = repmat(synthesized, 178, 1);
synthesized = reshape(synthesized.',1,[]);
audiowrite('partial.wav', real(synthesized), 8000);
```

For this question, I have chosen M = 164. I manually increased N2 to see the changes in the voice files. When N2 was relatively low, I observed that the pitch of the sound was lower than the original recording. However, as I increased N2 the pitch increased and I could approximate the original recording better with the partial coefficients I had chosen.

8)

```
% Part B Question 8
newFSC = zeros(1, size(resultPiano, 2));
for k = 1 : size(resultPiano, 2)
    [theta,rho] = cart2pol(real(resultPiano(1, k)), imag(resultPiano(1,
k)));
    [x, y] = pol2cart(theta, 1);
    newFSC(1, k) = x + y * 1i;
end
newSynthesis = synthesis(newFSC, w, N);
newSynthesis = repmat(newSynthesis, 178, 1);
newSynthesis = reshape(newSynthesis.',1,[]);
```

```
audiowrite('mag1.wav', real(newSynthesis), 8000);
```

When I equated all of the magnitudes to one and resynthesized the sound, it sounded as if the pitch increased compared to the original one. The sound became more shrill.

9)

```
% Part B Question 9
newFSCPart9 = zeros(1, size(resultPiano, 2));
for k = 1 : size(resultPiano, 2)
    [theta,rho] = cart2pol(real(resultPiano(1, k)), imag(resultPiano(1,
k)));
    [x, y] = pol2cart(0, rho);
    newFSCPart9(1, k) = x + y * 1i;
end
newSynthesis = synthesis(newFSCPart9, w, N);
newSynthesis = repmat(newSynthesis, 178, 1);
newSynthesis = reshape(newSynthesis.',1,[]);
audiowrite('phase0.wav', real(newSynthesis), 8000);
```

When I equated all of the phases to 0, the pitch seemed to increase. There is a subtle difference between the results of part 8 and 9.

Complete Matlab Code

```
%% PART A
% A1

%A1.1
% plot of x_1(t)
t = [0 : 1/8192 : 1];
x1 = cos(2*pi*550*t);
plot(t, x1);
title('x_1(t) versus t');
xlabel('t: Time (sec)');
ylabel('A: Amplitude');

% soundsc(x1);

x2 = cos(2*pi*430*t);
% soundsc(x2);

x3 = cos(2*pi*750*t);
% soundsc(x3);

% Bilkent ID: 21901548
f1 = 154;
f2 = 190;
y = 2 * cos(2*pi*f1*t) + 3 * sin(2*pi*f2*t);
fmax = max(f1, f2);

% plot of y
% hold on;
plot(t, y);
title('y(t) versus t');
xlabel('t: Time (sec)');
ylabel('Amplitude');

% sampling rate fmax
fs = fmax;
t1 = [0 : 1/fs : 1];
y1 = 2 * cos(2*pi*f1*t1) + 3 * sin(2*pi*f2*t1);
stem(t1, y1);
title('y_1(t) versus t');
xlabel('Time Index (n)');
ylabel('Amplitude');

% sampling rate 2fmax
fs = 2 * fmax;
t2 = [0 : 1/fs : 1];
y2 = 2 * cos(2*pi*f1*t2) + 3 * sin(2*pi*f2*t2);
stem(t2, y2);
title('y_2(t) versus t');
xlabel('Time Index (n)');
```

```
ylabel('Amplitude');

% sampling rate 4fmax
fs = 4 * fmax;
t3 = [0 : 1/fs : 1];
y3 = 2 * cos(2*pi*f1*t3) + 3 * sin(2*pi*f2*t3);
stem(t3, y3);
title('y_3(t) versus t');
xlabel('Time Index (n)');
ylabel('Amplitude');

% reconstructing signal (sampling rate fmax)
reconstructed = reconstruction(y1, size(y1, 2), 1 / fmax, size(t, 2));
plot(t,reconstructed);
title('y_1(t) reconstructed versus t');
xlabel('t: Time (sec)');
ylabel('Amplitude');

% reconstructing signal (sampling rate 2fmax)
reconstructed = reconstruction(y2, size(y2, 2), 1 / (2*fmax), size(t, 2));
plot(t, reconstructed);
title('y_2(t) reconstructed versus t');
xlabel('t: Time (sec)');
ylabel('Amplitude');

% reconstructing signal (sampling rate 4fmax)
reconstructed = reconstruction(y3, size(y3, 2), 1 / (4*fmax), size(t, 2));
plot(t, reconstructed);
title('y_3(t) reconstructed versus t');
xlabel('t: Time (sec)');
ylabel('Amplitude');
% method for reconstruction
function[reconstructed] = reconstruction(sampledSignal, numberOfSamples,
samplingPeriod, size)
    reconstructed = zeros(1, size);
    t = 0;
    for k = 1 : size
        for n = 1 : numberOfSamples
            reconstructed(1, k) = reconstructed(1, k) + sampledSignal(1, n)
* (sin(pi * (t - n * samplingPeriod)) / samplingPeriod) / (pi * (t - n *
samplingPeriod) / samplingPeriod);
        end
        t = t + 1/8192;
    end
end

%% PART A
% A2
syms t
x(t) = piecewise(0 <= t & t < 1, 3*t, 1 <= t & t < 2, 5-t, 2 <= t & t < 3,
3);
```

```
fplot(x(mod(t, 3)), [0, 9]);

y = @(k) (3 * (-6*1i*pi*exp(2*1i*pi*k)*k - 3*exp(4*1i*pi*k/3) +
2*exp(2*1i*pi*k/3)*(6+1i*pi*k)-9) / (4*pi.^2*k.^2));

sum = 0;
magnitudes = zeros(1, 20);
phases = zeros(1, 20);
counter = 1;
indices = zeros(1, 20);
for ind = -10 : 10
    if ~isnan(y(ind))
        ak = y(ind);
        magnitudes(1, counter) = abs(ak);
        phases(1, counter) = angle(ak);
        indices(1, counter) = ind;
        counter = counter + 1;
    end
end

subplot(1,2,1);
stem(indices, magnitudes);
title('Plot of Magnitudes of Xks for k in [-10, 10]');
xlabel('k');
ylabel('Magnitude');

subplot(1,2,2);
stem(indices, phases);
title('Plot of Phases of Xks for k in [-10, 10]');
xlabel('k');
ylabel('Phase');

%% PART B
% recObj = audiorecorder;
% disp('Start recording.');
% recordblocking(recObj, 5);
% disp('End recording')
% play(recObj);
% soundArray = getaudiodata(recObj);
soundArray = load("soundArray.mat");
soundArray = soundArray.soundArray;
recObj = load("recObj.mat");
recObj = recObj.recObj;
% play(recObj);
% plot the sound signal
figure;
subplot(1,2,1);
plot(soundArray);
title('Plot of soundArray');
xlabel('t: Time (sec)');
```

```
ylabel('Amplitude');

partOfSoundArray = soundArray(8242 : 16667);
subplot(1,3,1);
plot(partOfSoundArray);
title('Plot of partOfSoundArray');
xlabel('t: Time (sec)');
ylabel('Amplitude');

firstPeriodOfPartOfSoundArray = partOfSoundArray(1 : 45);
% subplot(1,3,2);
% plot(firstPeriodOfPartOfSoundArray);
f = 174;
w = 2*pi*f;
N = 350;
resultPiano = fs(firstPeriodOfPartOfSoundArray, w, N);
stem(-N:N, resultPiano);
title('Fourier Coefficients');
xlabel('k');
ylabel('ak');

% PART B Question 4

% F3 note on guitar
guitarSoundArray = load("GuitarF3soundArray.mat").soundArray;
guitarRecObj = load("GuitarF3recObj.mat").recObj;
guitarPartOfSoundArray = guitarSoundArray(12991 : 21110);
subplot(1,2,1);
plot(guitarSoundArray);
title('Plot of Guitar soundArray');
xlabel('t: Time (sec)');
ylabel('Amplitude');

subplot(1,2,2);
plot(guitarPartOfSoundArray);
title('Plot of Guitar partOfSoundArray');
xlabel('t: Time (sec)');
ylabel('Amplitude');

% F3 note on bass
bassSoundArray = load("BassF3soundArray.mat").soundArray;
bassRecObj = load("BassF3recObj.mat").recObj;
bassPartOfSoundArray = bassSoundArray(7710 : 21796);
subplot(1,2,1);
plot(bassSoundArray);
title('Plot of Bass soundArray');
xlabel('t: Time (sec)');
ylabel('Amplitude');

subplot(1,2,2);
plot(bassPartOfSoundArray);
```

```
title('Plot of Bass partOfSoundArray');
xlabel('t: Time (sec)');
ylabel('Amplitude');

% FSC calculation
firstPeriodOfPartOfSoundArray = guitarPartOfSoundArray(1 : 45);
resultGuitar = fs(firstPeriodOfPartOfSoundArray, w, N);
stem(-N:N, resultGuitar);
title('Fourier Coefficients');
xlabel('k');
ylabel('ak');

firstPeriodOfPartOfSoundArray = bassPartOfSoundArray(1 : 45);
resultBass = fs(firstPeriodOfPartOfSoundArray, w, N);
stem(-N:N, resultBass);
title('Fourier Coefficients');
xlabel('k');
ylabel('ak');

% Part B Question 5
synthesized = synthesis(resultPiano, w, N);
subplot(1,2,1);
plot(real(synthesized));
title('Synthesized Voice (one period)');
xlabel('n');
ylabel('Amplitude');
synthesized = repmat(synthesized, 178, 1);
synthesized = reshape(synthesized.',1,[]);
subplot(1,2,2);
plot(real(synthesized));
title('Synthesized Voice (same size as partOfSoundArray)');
xlabel('n');
ylabel('Amplitude');

audiowrite('handel.wav', real(synthesized), 8000);

% Part B Question 7
N1 = 1;
M = 164;
N2 = 2;
for k = 1 : size(resultPiano, 2)
    if resultPiano(1, k) > 0.0001
        N1 = k;
        break;
    end
end

synthesized = synthesis(resultPiano, w, (N2 - N1) / 2);
synthesized = repmat(synthesized, 178, 1);
synthesized = reshape(synthesized.',1,[]);
audiowrite('partial.wav', real(synthesized), 8000);
```

```
% Part B Question 8
newFSC = zeros(1, size(resultPiano, 2));
for k = 1 : size(resultPiano, 2)
    [theta,rho] = cart2pol(real(resultPiano(1, k)), imag(resultPiano(1,
k)));
    [x, y] = pol2cart(theta, 1);
    newFSC(1, k) = x + y * 1i;
end

newSynthesis = synthesis(newFSC, w, N);
newSynthesis = repmat(newSynthesis, 178, 1);
newSynthesis = reshape(newSynthesis.',1,[]);
audiowrite('mag1.wav', real(newSynthesis), 8000);

% Part B Question 9
newFSCPart9 = zeros(1, size(resultPiano, 2));
for k = 1 : size(resultPiano, 2)
    [theta,rho] = cart2pol(real(resultPiano(1, k)), imag(resultPiano(1,
k)));
    [x, y] = pol2cart(0, rho);
    newFSCPart9(1, k) = x + y * 1i;
end

newSynthesis = synthesis(newFSCPart9, w, N);
newSynthesis = repmat(newSynthesis, 178, 1);
newSynthesis = reshape(newSynthesis.',1,[]);
audiowrite('phase0.wav', real(newSynthesis), 8000);

% function for synthesis
function[onePeriod] = synthesis(coefficients, w, N)
    f = w/(2*pi);
    T = 1 / f;
    t = 0 : 1/8000 : T;
    t = t(1:45);
    onePeriod = zeros(1, size(t, 2));

    for j = 1 : 45
        for k = -N : N
            onePeriod(1, j) = onePeriod(1, j) + coefficients(1, k + N + 1)
* exp( 1i * k * w * t(1, j));
        end
    end
end

function[ak] = fs(sound, w, N)
   ak = zeros(1,2 * N+1);
   f = w/(2*pi);
   T = 1 / f;
   t = 0 : 1/8000 : T;
   t = t(1:45);
```

```
    for k = -N : N
        ak(1, k + N + 1) = f * trapz(( exp( -1i * k * w * t)).*(sound.'));
    end
end
```

Index of comments