­

Fall

**Amplitude Modulation**

**Project**

2022

Communication

Name Sec BN

Zeyad Tarek 1 28

Abdelrahman Hamdy 1 36

**Explanation of our work:**

Starting off by reading our 3 input signals from the “Input” folder, we make sure that the sampling frequency in all these signals is the same for consistent modulation. Then, we choose the first channel for our messages and find the length of each one along with the maximum length. An extra step is done which I found helped in eliminating noise and greatly reducing the overlap between the 3 signals which is to up-sample the signals with a sample ratio of 3 before modulation and then down sample after demodulation to restore the signal back to its original state. This is done using a function called “resample” which takes 3 arguments: The signal itself, x, and y. The sample rate is basically multiplied by the ratio of x/y. In order to be able to sum all modulated signals of each input, we need to make sure that they all have the same length so the next crucial step is to set their lengths to this maximum length and fill the gaps with zeros. We also adjust the time and frequency intervals on this max length and the original sampling frequency.

***Carrier Frequency:***

The carrier frequency is set to be the width of the signal with the largest width to ensure that when the signals are shifted during modulation, they do not overlap and fail to restore the input message signals. As for the second carrier frequency, it has to satisfy a condition where it covers the other signals by 2 \* Bandwidth plus another half of the width which is the same carrier frequency of the first. As a conclusion, we have two frequencies to work with: Fc\_1 which is the half width of the message signal drawn in the frequency domain and Fc\_2 which is triple that width.

***Modulation:***

We perform Double-Sideband Suppressed Carrier (DSB-SC) modulation on the first input signal with a cosine carrier of frequency Fc\_1. As for the second and third input signals, we use QAM (Quadrature amplitude modulation) where two signals are sent simultaneously over the same bandwidth (on the same carrier frequency Fc\_2) for efficient use of the bandwidth, but one is a sine function and the other is cosine. The difference in phase between these 2 functions helps us restore them separately without having to worry about the overlap that could happen.

***What about the final modulated signal?***

Now this should be the easiest step as all we have to do is sum the three modulated signals obtained from the DSB-SC and QAM to get our resulting modulated signal, ready for being used for demodulation.

***Demodulation:***

Our passband frequency for the low pass filter is the same as the original carrier frequency as all we need to do is filter each signal alone and output it for testing. For the synchronous demodulation with zero phase shift and frequency addition, the LPF takes the final modulated signal, the sampling frequency and the passband frequency. The signal is multiplied by 3 carriers, one for each input signal, obtaining an output of half the original signal in amplitude so we simply finish by multiplying by 2 and passing the necessary parameters mentioned above to the low pass filter to get our audio back again.

***Demodulation with phase shift:***

The same exact steps done with the synchronous demodulation will be done here but the only difference is that we add a phase shift with the carrier inside the cosine function. As required, we have a phase shift of 10, 30, and 90 degrees (all converted to radians by a simple calculation of course) which will affect the output signal. Phase shift causes attenuation of the signal which means decreasing its amplitude. However, there is a special case for phase shift 90 because it causes the first signal to be almost gone and can’t be heard as cosine (90) is zero. For the other 2 signals, we used QAM by applying cosine and sine for the same frequency and when adding a 90 degree phase shift, the two signals interchange. One takes place of the other and vice versa because 90 phase shift causes cosine to become sine and the opposite as well.

***Demodulation with a different local carrier frequency:***

A constant is added with the carrier frequency that is calculated with Omega and then we observe the effect of that on the carrier when it’s multiplied by the modulated signal. For this, we have 2 cases where we add this constant on Fc: 2Hz and 10 Hz. The effect adding a 2Hz constant is distortion of the signal and a bit of noise will be clearly heard in the background but we would still be able to identify the audio from the main core of the signal as the effect wasn’t that much drastic. On the other hand, with 10 Hz the signal will be mostly disrupted by random noises and we would fail to know which audio is currently playing.

***Message 1 Signal:***

Timeline

Description automatically generated

***Message 2 Signal:***

Timeline

Description automatically generated with medium confidence

***Message 3 Signal:***

Timeline

Description automatically generated

***Message 1 Modulated Signal:***

Timeline

Description automatically generated

***Message 2 Modulated Signal:***

Timeline

Description automatically generated

***Message 3 Modulated Signal:***

A picture containing chart

Description automatically generated

***Sum of the three modulated input signals:***

A picture containing timeline

Description automatically generated

***Message 1 Demodulated signal with synchronous carrier:***

***Graphical user interface

Description automatically generated with low confidence***

***Message 2 Demodulated signal with synchronous carrier:***

***Timeline

Description automatically generated***

***Message 3 Demodulated signal with synchronous carrier:***

***Timeline

Description automatically generated***

***Message 1*** ***demodulated PhaseShift10:***

***Timeline

Description automatically generated with medium confidence***

***Message 2*** ***demodulated PhaseShift10:***

***Timeline

Description automatically generated***

***Message 3*** ***demodulated PhaseShift10:***

***Timeline

Description automatically generated***

***Message 1 demodulated PhaseShift30:***

***Timeline

Description automatically generated***

***Message 2 demodulated PhaseShift30:***

***Timeline

Description automatically generated***

***Message 3 demodulated PhaseShift30:***

***Chart, timeline

Description automatically generated***

***Message 1 demodulated PhaseShift90:***

***Timeline

Description automatically generated with medium confidence***

***Message 2 demodulated PhaseShift90:***

***Timeline

Description automatically generated***

***Message 3 demodulated PhaseShift90:***

***Timeline

Description automatically generated***

***Message 1 demodulated Carrier frequency increased by 2 Hz:***

***Chart

Description automatically generated***

***Message 2 demodulated Carrier frequency increased by 2 Hz:***

***Timeline

Description automatically generated***

***Message 3 demodulated Carrier frequency increased by 2 Hz:***

***Timeline

Description automatically generated with medium confidence***

***Message 1 demodulated Carrier frequency increased by 10 Hz:***

***A picture containing chart

Description automatically generated***

***Message 2 demodulated Carrier frequency increased by 10 Hz:***

***Timeline

Description automatically generated***

***Message 3 demodulated Carrier frequency increased by 10 Hz:***

***Timeline

Description automatically generated***

***Code:***

% Close all opened figures

close all;

% Read the input audio

[message\_1, fs\_1] = audioread('Input/audio\_1.m4a');

[message\_2, fs\_2] = audioread('Input/audio\_2.m4a');

[message\_3, fs\_3] = audioread('Input/audio\_3.m4a');

% Set sample ratio

global RESAMPLE\_FACTOR;

RESAMPLE\_FACTOR = 3;

% Select first channel for the three signals & Up-sample

message\_1 = resample(message\_1(:, 1), RESAMPLE\_FACTOR, 1);

message\_2 = resample(message\_2(:, 1), RESAMPLE\_FACTOR, 1);

message\_3 = resample(message\_3(:, 1), RESAMPLE\_FACTOR, 1);

% Set Sampling Frequency (They are all the same)

fs = fs\_1;

% Get length of the 3 signals

N\_1 = length(message\_1);

N\_2 = length(message\_2);

N\_3 = length(message\_3);

% Get the maximum length among them

max\_N = max([N\_1, N\_2, N\_3]);

% Adjust all lengths of the 3 input signals to be the same (having max length)

message\_1 = [message\_1;zeros(max\_N-N\_1, 1)];

message\_2 = [message\_2;zeros(max\_N-N\_2, 1)];

message\_3 = [message\_3;zeros(max\_N-N\_3, 1)];

% Time & Frequency intervals

time = linspace(0, max\_N/fs, max\_N);

df = fs/2;

freq = -df : fs/max\_N: df - fs/max\_N;

% Amplitude and phase in frequency domain

message\_f1 = fftshift(fft(message\_1));

message\_f2 = fftshift(fft(message\_2));

message\_f3 = fftshift(fft(message\_3));

phase\_1 = unwrap(angle(message\_f1));

phase\_2 = unwrap(angle(message\_f2));

phase\_3 = unwrap(angle(message\_f3));

% Carrier Frequency

fc\_1 = 5000;

fc\_2 = 3 \* fc\_1;

% Omega = 2piF

wc\_1 = 2\*pi \* fc\_1;

wc\_2 = 2\*pi \* fc\_2;

% Carrier Equation in time domain

carrier\_t1 = cos(wc\_1 \* time);

carrier\_t2 = cos(wc\_2 \* time);

carrier\_t3 = sin(wc\_2 \* time);

% Carrier amplitude and phase in frequency domain

carrier\_f1 = fftshift(fft(carrier\_t1));

carrier\_f2 = fftshift(fft(carrier\_t2));

carrier\_f3 = fftshift(fft(carrier\_t3));

phase\_carrier1 = unwrap(angle(carrier\_f1));

phase\_carrier2 = unwrap(angle(carrier\_f2));

phase\_carrier3 = unwrap(angle(carrier\_f3));

% Apply DSB-SC Modulation

modulatedSignal\_t1 = message\_1' .\* carrier\_t1;

modulatedSignal\_t2 = message\_2' .\* carrier\_t2;

modulatedSignal\_t3 = message\_3' .\* carrier\_t3;

% Modulated Signals in frequency domain

modulatedSignal\_f1 = fftshift(fft(modulatedSignal\_t1));

modulatedSignal\_f2 = fftshift(fft(modulatedSignal\_t2));

modulatedSignal\_f3 = fftshift(fft(modulatedSignal\_t3));

phase\_mod1 = unwrap(angle(modulatedSignal\_f1));

phase\_mod2 = unwrap(angle(modulatedSignal\_f2));

phase\_mod3 = unwrap(angle(modulatedSignal\_f3));

% Add modulated signals

finalModulatedSignal = modulatedSignal\_t1 + modulatedSignal\_t2 + modulatedSignal\_t3;

% Get modulated signal amplitude and phase in frequency domain

finalModulatedSignal\_f = fftshift(fft(finalModulatedSignal));

phase\_mod = unwrap(angle(finalModulatedSignal\_f));

% Plot Message Signals

draw(time, freq.', 1, message\_1, abs(message\_f1), phase\_1, 'Message 1');

draw(time, freq.', 2, message\_2, abs(message\_f2), phase\_2, 'Message 2');

draw(time, freq.', 3, message\_3, abs(message\_f3), phase\_3, 'Message 3');

% Plot Modulated Signals

draw(time, freq.', 4, modulatedSignal\_t1, abs(modulatedSignal\_f1), phase\_mod1, 'Modulated 1');

draw(time, freq.', 5, modulatedSignal\_t2, abs(modulatedSignal\_f2), phase\_mod2, 'Modulated 2');

draw(time, freq.', 6, modulatedSignal\_t3, abs(modulatedSignal\_f3), phase\_mod3, 'Modulated 3');

% Plot the Final Modulated Signal

draw(time, freq.', 7, finalModulatedSignal, abs(finalModulatedSignal\_f), phase\_mod, 'FINAL Modulated Signal');

% Figure counter for Plotting

global j;

j = 8;

% Set passband frequency

fp = fc\_1;

% Synchronous Demodulation

demodulate(finalModulatedSignal, carrier\_t1, fs, fp, "Sync\_1");

demodulate(finalModulatedSignal, carrier\_t2, fs, fp, "Sync\_2");

demodulate(finalModulatedSignal, carrier\_t3, fs, fp, "Sync\_3");

% Phase Shift = 10

[carrier1\_10, carrier2\_10, carrier3\_10] = generateCarriersWithPhase(fc\_1, fc\_2, time, 10);

demodulate(finalModulatedSignal, carrier1\_10, fs, fp, "PhaseShift10\_1");

demodulate(finalModulatedSignal, carrier2\_10, fs, fp, "PhaseShift10\_2");

demodulate(finalModulatedSignal, carrier3\_10, fs, fp, "PhaseShift10\_3");

% Phase Shift = 30

[carrier1\_30, carrier2\_30, carrier3\_30] = generateCarriersWithPhase(fc\_1, fc\_2, time, 30);

demodulate(finalModulatedSignal, carrier1\_30, fs, fp, "PhaseShift30\_1");

demodulate(finalModulatedSignal, carrier2\_30, fs, fp, "PhaseShift30\_2");

demodulate(finalModulatedSignal, carrier3\_30, fs, fp, "PhaseShift30\_3");

% Phase Shift = 90

[carrier1\_90, carrier2\_90, carrier3\_90] = generateCarriersWithPhase(fc\_1, fc\_2, time, 90);

demodulate(finalModulatedSignal, carrier1\_90, fs, fp, "PhaseShift90\_1");

demodulate(finalModulatedSignal, carrier2\_90, fs, fp, "PhaseShift90\_2");

demodulate(finalModulatedSignal, carrier3\_90, fs, fp, "PhaseShift90\_3");

% Demodulation with local carrier frequency different than Fc by 2 Hz

[carrier1\_2Hz, carrier2\_2Hz, carrier3\_2Hz] = generateCarriersWithDifferentFc(fc\_1, fc\_2, time, 2);

demodulate(finalModulatedSignal, carrier1\_2Hz, fs, fp, "Fc2Hz\_1");

demodulate(finalModulatedSignal, carrier2\_2Hz, fs, fp, "Fc2Hz\_2");

demodulate(finalModulatedSignal, carrier3\_2Hz, fs, fp, "Fc2Hz\_3");

% Demodulation with local carrier frequency different than Fc by 10 Hz

[carrier1\_10Hz, carrier2\_10Hz, carrier3\_10Hz] = generateCarriersWithDifferentFc(fc\_1, fc\_2, time, 10);

demodulate(finalModulatedSignal, carrier1\_10Hz, fs, fp, "Fc10Hz\_1");

demodulate(finalModulatedSignal, carrier2\_10Hz, fs, fp, "Fc10Hz\_2");

demodulate(finalModulatedSignal, carrier3\_10Hz, fs, fp, "Fc10Hz\_3");

function draw(time, freq, i, mt, mf, phase, T)

% Plot Message in Time Domain

figure(i)

subplot(3, 1, 1)

plot(time, mt)

xlabel('Time')

ylabel('Amp')

title(strcat(T, ' Signal (Time)'))

% Plot Message Amplitude in Frequency Domain

subplot(3, 1, 2)

plot(freq, mf)

xlabel('Freq')

ylabel('Amp')

title(strcat(T, ' Signal Amplitude (Freq)'))

% Plot Message Phase in Frequency Domain

subplot(3, 1, 3)

plot(freq, phase)

xlabel('Freq')

ylabel('Phase')

title(strcat(T, ' Signal Phase (Freq)'))

end

function demodulate(signal, carrier, fs, fp, filename)

% Re-multiply by the carrier to restore the signal with half magnitude

% So we double the output to get the exact original signal

demodulatedSignal = 2 \* (signal .\* carrier);

% Low pass filter to get each message alone

lpf = lowpass(demodulatedSignal, fp, fs);

% Down-sample the signal back to its original Fs

global RESAMPLE\_FACTOR;

lpf = resample(lpf, 1, RESAMPLE\_FACTOR);

% Get demodulated signal amplitude and phase in Frequency Domain

lpf\_f = fftshift(fft(lpf));

phase\_demod = unwrap(angle(lpf\_f));

% Plot the demodulated signal

N = length(lpf);

global j;

time = linspace(0, N/fs, N);

freq = -(fs/2) : fs/N: (fs/2) - fs/N;

draw(time, freq.', j, lpf, abs(lpf\_f), phase\_demod, filename);

j = j + 1;

% Output the final result to an audio file

audiowrite(strcat("Output/", filename, '.m4a'), lpf, fs);

end

function [carrier1, carrier2, carrier3] = generateCarriersWithPhase(fc\_1, fc\_2, time, deg)

% Phase Shift in radians

phaseShift = (deg \* pi) / 180;

% Omega = 2piF

wc\_1 = 2\*pi \* fc\_1;

wc\_2 = 2\*pi \* fc\_2;

% Get carrier for each signal with an additional phase shift

carrier1 = cos((wc\_1 \* time) + phaseShift);

carrier2 = cos((wc\_2 \* time) + phaseShift);

carrier3 = sin((wc\_2 \* time) + phaseShift);

end

function [carrier1, carrier2, carrier3] = generateCarriersWithDifferentFc(fc\_1, fc\_2, time, dfc)

% Get new omega with a constant added to the carrier frequency

wc\_1 = 2\*pi \* (fc\_1 + dfc);

wc\_2 = 2\*pi \* (fc\_2 + dfc);

% Get new carriers just like we did in synchronous demodulation

carrier1 = cos(wc\_1 \* time);

carrier2 = cos(wc\_2 \* time);

carrier3 = sin(wc\_2 \* time);

end