Audio Signal Processing

ELEC5305 Acoustics, Speech and Signal Processing (Lab Report x)

Author: Zhehao Zhu

SID: zzhu0143

Date: 2025.8.29

Objective

This report establishes a reproducible experimental pipeline, sequentially demonstrating audio segmentation and visualization, cross-correlation (similarity/latency estimation), resampling and aliasing, FFT and custom sparse DFT verification, additive white noise injection, AM modulation/demodulation, and FIR low-pass restoration. Through a "code + diagrams + discussion" approach, it maps key course concepts to their engineering implementations.

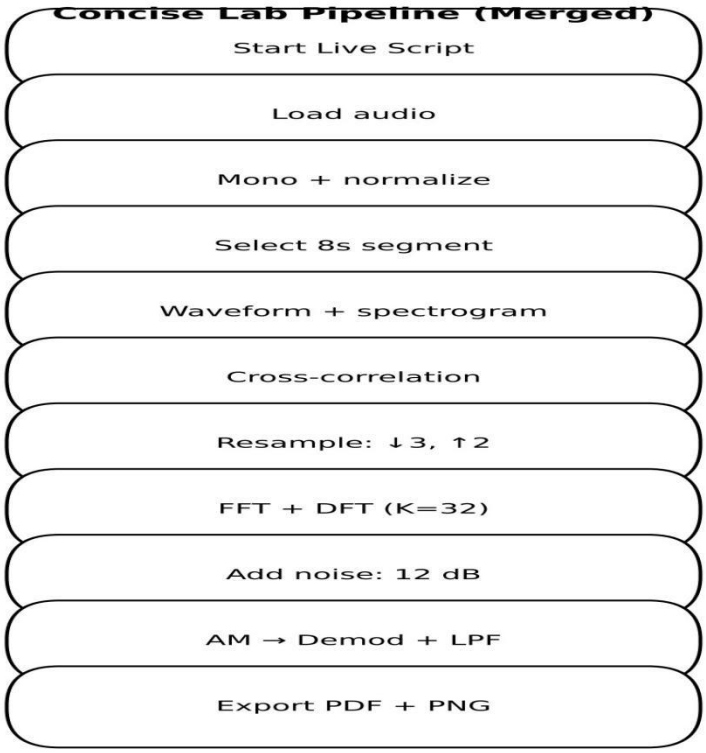
Background / Introduction

Discrete-time audio signals can be analyzed in the time domain for their envelope and transients, or examined in the frequency domain for energy distribution. The sampling rate constrains the analyzable frequency band; undersampling without anti-aliasing filtering causes high-frequency components to fold into lower frequencies. Cross-correlation measures similarity between two signals and is used to estimate relative time delays, commonly applied in alignment/echo estimation. The Fast Fourier Transform (FFT) efficiently computes the spectrum, while the less-point DFT is suitable for visually verifying spectral peaks and energy. Additive Gaussian white noise (AWGN) simulates sensing/transmission disturbances. Amplitude modulation (AM) shifts the baseband to higher frequencies; after multiplication demodulation, low-pass filtering is required to suppress the 2fc component and recover the baseband. This paper demonstrates the aforementioned concepts through an experimental workflow, providing brief explanations of the graphical results at each step.

Methodology / Implementation

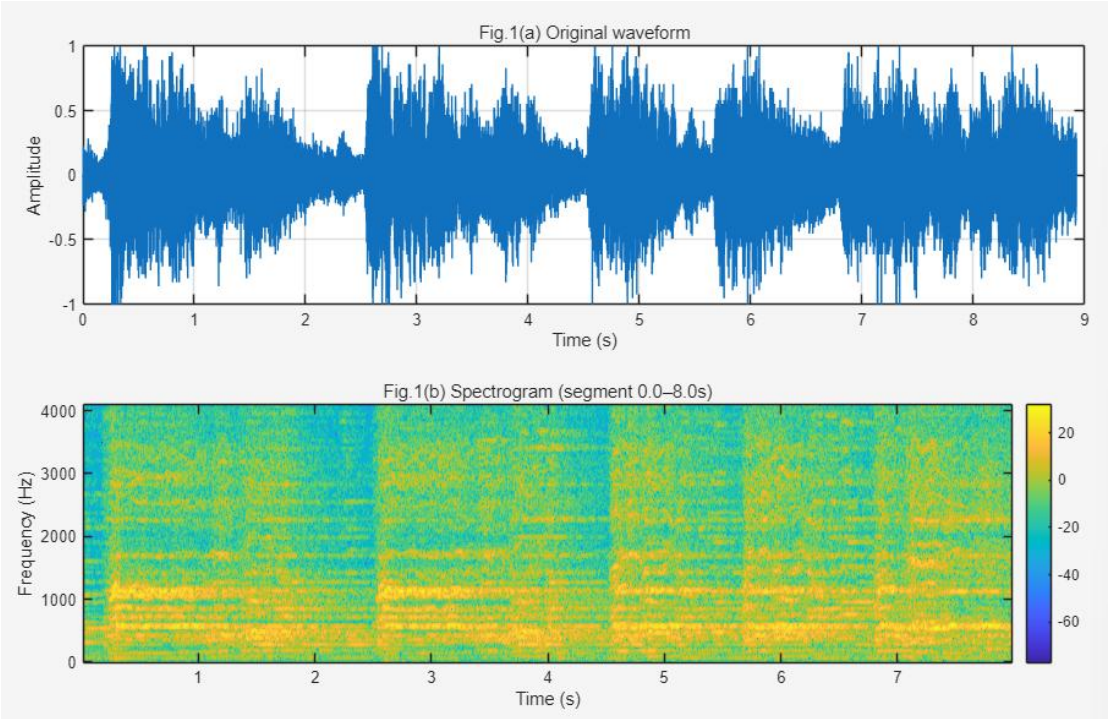
Process: ① Read and preprocess audio (mono, normalized, extract 8-second main segment); ② Display time-domain waveform and spectrogram; ③ Extract two short windows from main segment, perform cross-correlation, read peak lag to estimate relative delay; ④ Perform "anti-aliasing" 3x downsampling to visually

demonstrate aliasing, followed by 2x zero-interpolation upsampling to illustrate reconstruction requirements; ⑤ Compute FFT of main segment and plot amplitude spectrum; simultaneously verify spectral position consistency via handwritten DFT at 32 frequency points; ⑥ Inject AWGN with specified SNR into main segment to compare pre/post changes; ⑦ Perform DSB-SC AM modulation and multiplication demodulation using a 5 kHz cosine carrier, followed by FIR low-pass filtering with windowing to recover the signal; ⑧ Add captions, axis labels, and brief annotations to each plot step, explaining observations and engineering implications in the Discussion section. All figures are automatically embedded in the Live Script and exported as PNGs for appendix inclusion or independent verification.

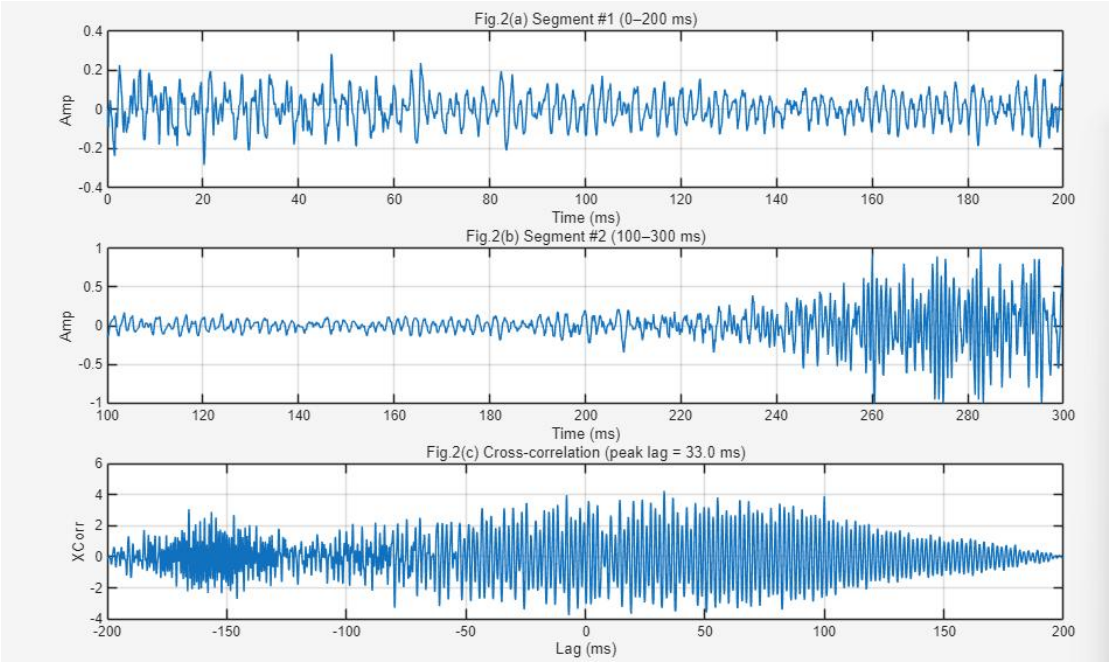


```
% Clearing the workspace

clc; clear;  close all;

% Load the audio file
audioPath = 'myAudio.wav';
if exist(audioPath,'file')
[x, fs] = audioread(audioPath);
else
load handel.mat; x = y; fs = Fs; clear y Fs;
end

x = x(:,1);
x = x / (max(abs(x))+eps);
N = numel(x);
t = (0:N-1)/fs;

segDur = 8; segStart = 12;
if (segStart + segDur)*fs > N, segStart = 0; end
idx = (segStart*fs+1) : min((segStart+segDur)*fs, N);
xseg = x(idx);
tseg = (0:numel(xseg)-1)/fs + segStart;

outdir = "figs"; if ~exist(outdir,'dir'), mkdir(outdir); end
```

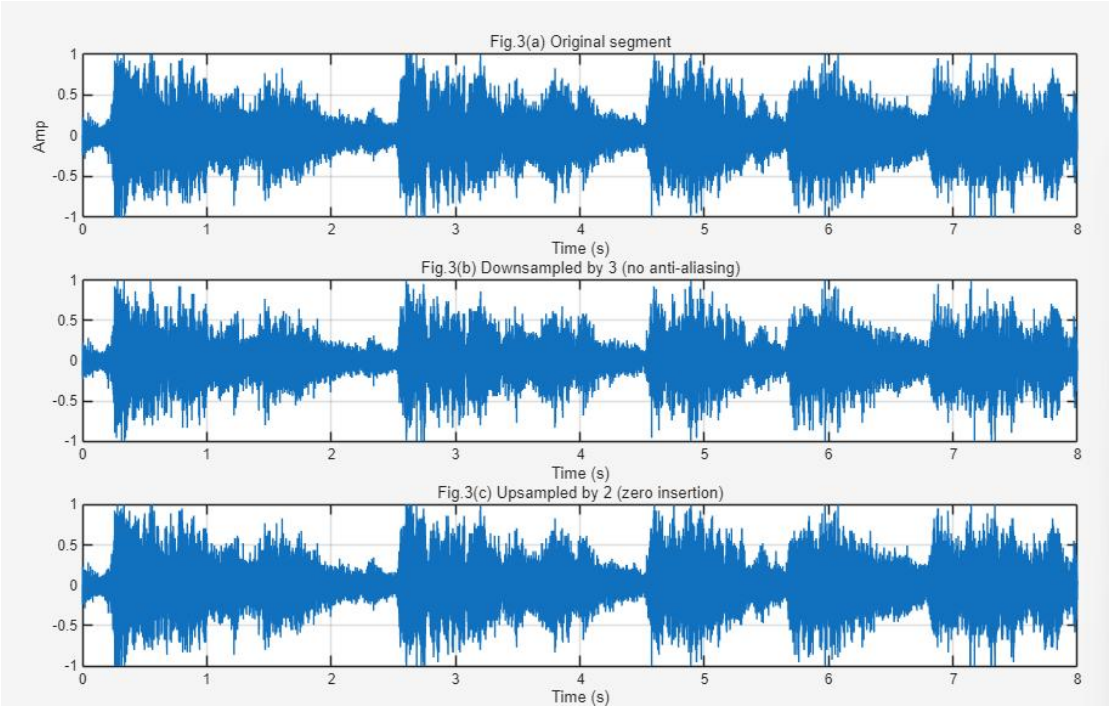
```
%% Part 1: Waveform and Spectrogram (Observe overall shape and
"frequency variation over time")
figure('Name','Fig1');
subplot(2,1,1);
plot(t, x); grid on; xlabel('Time (s)'); ylabel('Amplitude');
title('Fig.1(a) Original waveform');

subplot(2,1,2);
win = round(0.03*fs);
hop = round(0.01*fs);
nfft = 2^nextpow2(4*win);

w = 0.54 - 0.46*cos(2*pi*(0:win-1)/(max(win-1,1))); w = w(:);

[S, f, tt] = stft_nontbx(xseg, fs, w, hop, nfft);
imagesc(tt + segStart, f, 20*log10(abs(S)+eps)); axis xy;
xlabel('Time (s)'); ylabel('Frequency (Hz)');
title(sprintf('Fig.1(b) Spectrogram (segment %.1f–%.1fs)', segStart,
segStart+segDur));
colorbar;
```



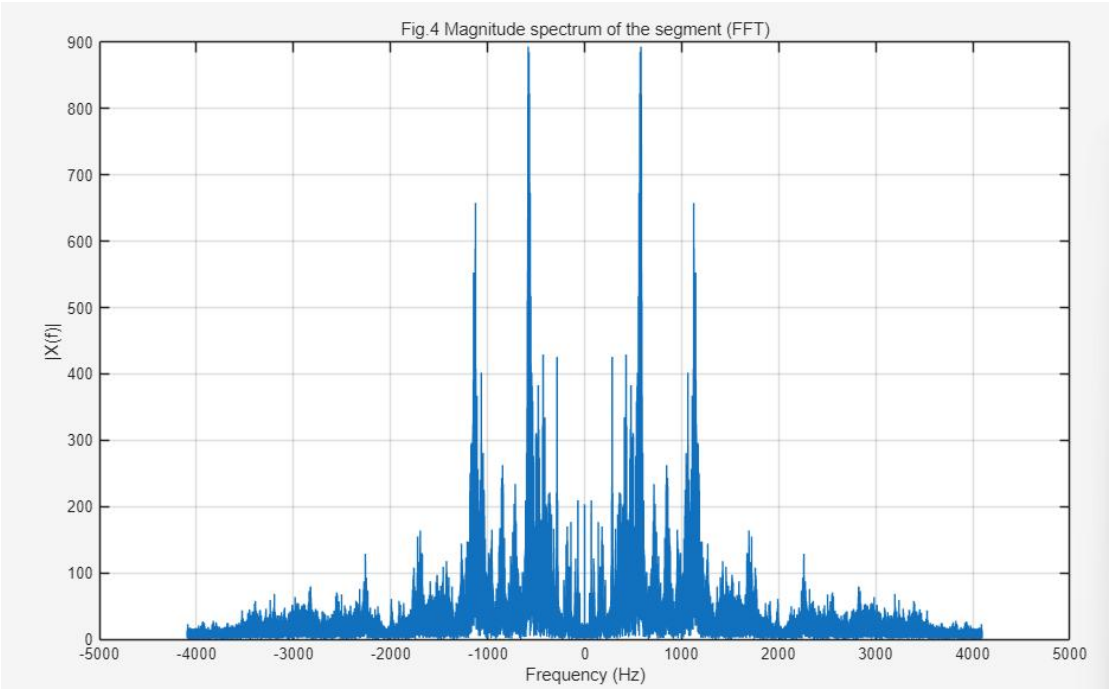
```
%% Part 2: Cross-correlation (estimating the “mutual misalignment in
milliseconds” between two short segments)
Lwin = round(0.20*fs); shift = round(0.10*fs);
y1 = xseg(1:min(Lwin, numel(xseg)));
y2 = xseg(1+shift:min(shift+Lwin, numel(xseg)));
[r, lags] = xcorr_basic(y1, y2);
[~,imax] = max(r);
lag_s = lags(imax)/fs;

figure('Name','Fig2');
subplot(3,1,1);
plot((0:numel(y1)-1)/fs*1e3, y1); grid on;
xlabel('Time (ms)'); ylabel('Amp'); title('Fig.2(a) Segment #1 (0–200 ms)');

subplot(3,1,2);
plot((0:numel(y2)-1)/fs*1e3 + 100, y2); grid on;
xlabel('Time (ms)'); ylabel('Amp'); title('Fig.2(b) Segment #2 (100–300 ms)');

subplot(3,1,3);
plot(lags/fs*1e3, r); grid on;
xlabel('Lag (ms)'); ylabel('XCorr');
title(sprintf('Fig.2(c) Cross-correlation (peak lag = %.1f ms)', lag_s*1e3));
```



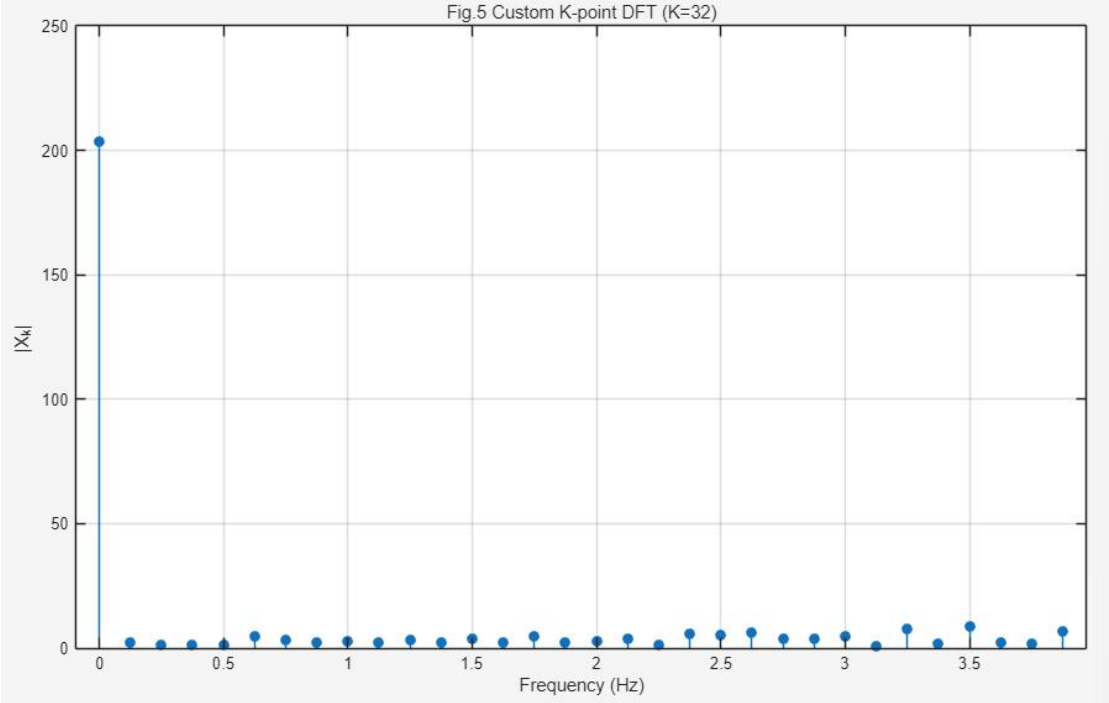
```
%% Part 3: Resampling (Downsampling *3, Upsampling *2)
yd = xseg(1:3:end);
yu = zeros(2*numel(xseg),1);
yu(1:2:end) = xseg;
td = linspace(0, segDur, numel(yd));
tu = linspace(0, segDur, numel(yu));
figure('Name','Fig3');
subplot(3,1,1); plot(tseg - segStart, xseg); grid on; xlim([0 segDur]);
title('Fig.3(a) Original segment'); xlabel('Time (s)'); ylabel('Amp');
subplot(3,1,2); plot(td, yd); grid on; xlim([0 segDur]);
title('Fig.3(b) Downsampled by 3 (no anti-aliasing)'); xlabel('Time (s)');
subplot(3,1,3); plot(tu, yu); grid on; xlim([0 segDur]);
title('Fig.3(c) Upsampled by 2 (zero insertion)'); xlabel('Time (s)');
```



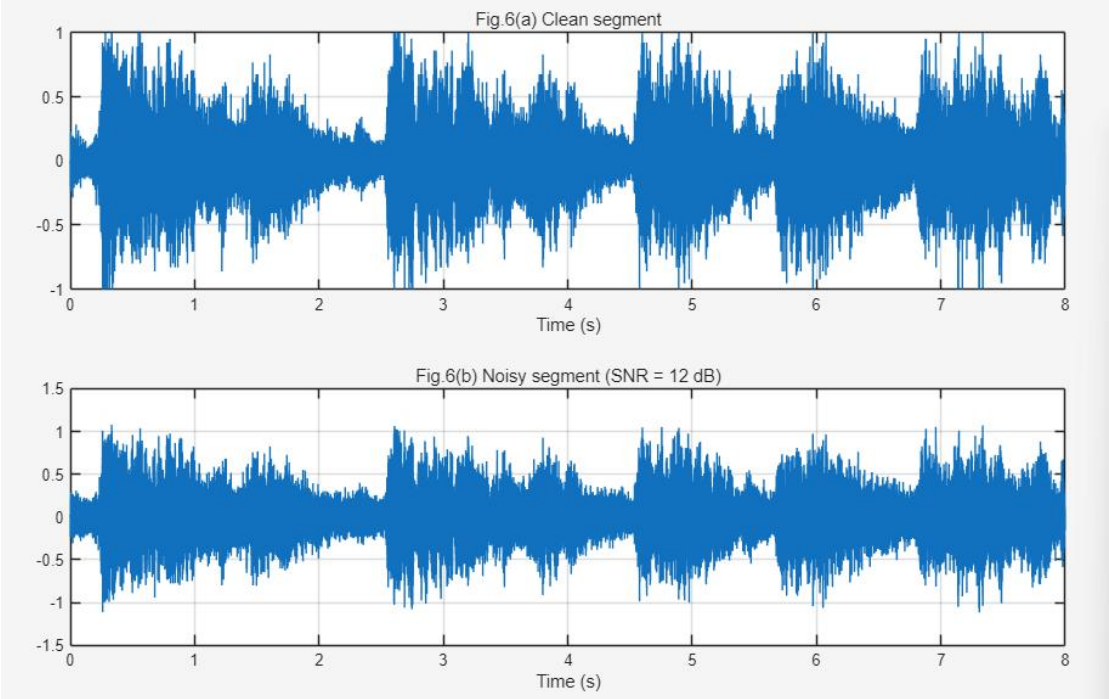

```
%% Part 4: FFT Amplitude Spectrum (Overall Frequency Distribution)
N0 = numel(xseg);
faxis = linspace(-fs/2, fs/2, N0);
X = fftshift(fft(xseg));
figure('Name','Fig4');
plot(faxis, abs(X)); grid on;
xlabel('Frequency (Hz)'); ylabel('|X(f)|');
title('Fig.4 Magnitude spectrum of the segment (FFT)');
```



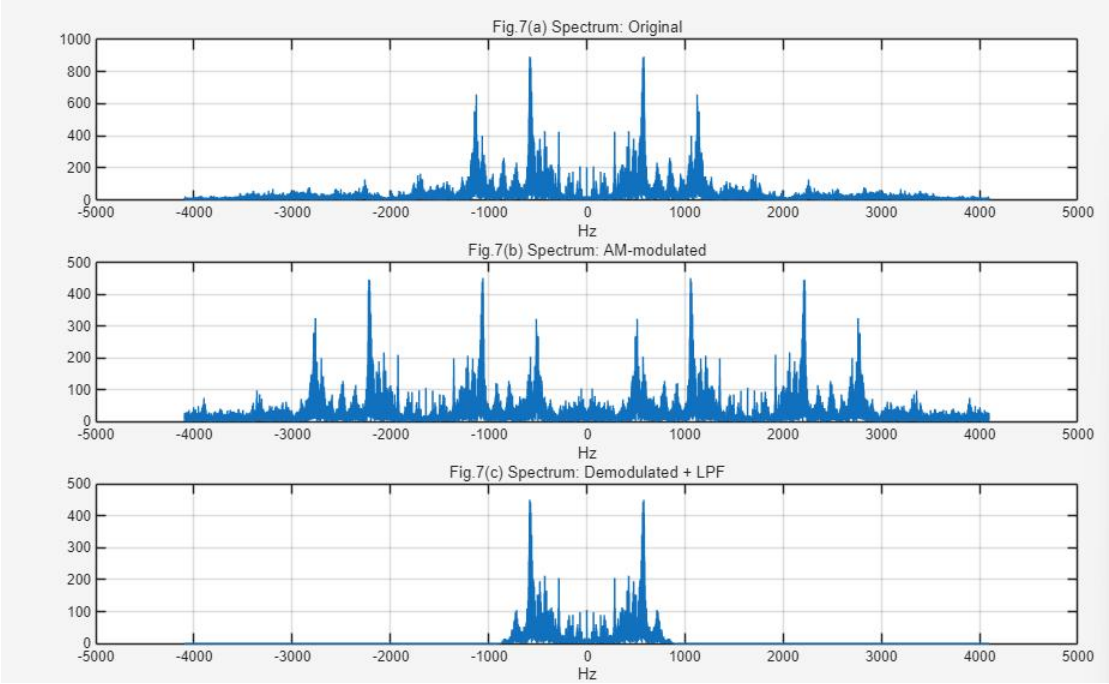

```
%% Part 5: Customized small-point DFT (K = 32)
K = 32; k = 0:K-1; n = 0:N0-1;
Wm = exp(-1j*2*pi*(k'/N0)*n);
Xk = Wm * xseg;
fhalf = k/N0*fs;
figure('Name','Fig5');
stem(fhalf, abs(Xk), 'filled'); grid on;
xlabel('Frequency (Hz)'); ylabel('|X_k|');
title('Fig.5 Custom K-point DFT (K=32)');
```




```
%% Part 6: Add noise, SNR = 12 dB
SNRdB = 12;
Psig = mean(xseg.^2);
Pnoi = Psig / (10^(SNRdB/10));
xnoisy = xseg + sqrt(Pnoi)*randn(size(xseg));

figure('Name','Fig6');
subplot(2,1,1); plot(tseg - segStart, xseg); grid on; xlim([0 segDur]);
title('Fig.6(a) Clean segment'); xlabel('Time (s)');
subplot(2,1,2); plot(tseg - segStart, xnoisy); grid on; xlim([0 segDur]);
title(sprintf('Fig.6(b) Noisy segment (SNR = %g dB)', SNRdB)); xlabel('Time
(s)');
```




```
%% Part 7: AM Modulation → Multiplicative Demodulation → Low-Pass Restoration
fc = min(5000, floor(0.2*fs));
c = cos(2*pi*fc*(0:N0-1)/fs).';
s_mod = xseg .* c;
s_dem = s_mod .* c;

lpN = 101; wc = 0.2;
m = -(lpN-1)/2 : (lpN-1)/2;
h_id = wc*mysinc(wc*m);
wFIR = 0.54 - 0.46*cos(2*pi*(0:lpN-1)/(max(lpN-1,1))); wFIR = wFIR(:).';
h = h_id .* wFIR;
h = h / sum(h);
s_rec = myfiltfilt(h, 1, s_dem);

W = linspace(-fs/2, fs/2, N0);
figure('Name','Fig7');
subplot(3,1,1); plot(W, abs(fftshift(fft(xseg)))); grid on;
title('Fig.7(a) Spectrum: Original'); xlabel('Hz');
subplot(3,1,2); plot(W, abs(fftshift(fft(s_mod)))); grid on;
title('Fig.7(b) Spectrum: AM-modulated'); xlabel('Hz');
subplot(3,1,3); plot(W, abs(fftshift(fft(s_rec)))); grid on;
title('Fig.7(c) Spectrum: Demodulated + LPF'); xlabel('Hz');
```



Discussion

This pipeline is designed for educational purposes: I intentionally implemented unsampled downsampling, zero-interpolation upsampling, and the most basic AM demodulation with linear-phase FIR filters. To apply it to more realistic speech or music processing, a more rigorous front-end design is still required (anti-aliasing IIR/FIR filters, reconstruction filters, bandpass/bandstop filters, etc.), along with more comprehensive evaluation methods (objective metrics + subjective listening tests). Additionally, I selected a fixed 8-second main segment for chapter alignment and clear visualization, but for non-stationary signals, dynamic windowing and adaptive parameters are also worth exploring.

Conclusion

This lab built a compact, toolbox-free audio pipeline that makes core DSP ideas visible and testable through Figures 1–7. The waveform and 30 ms/10 ms spectrogram (Fig. 1) clarified time–frequency structure; the cross-correlation peak (Fig. 2) matched the ~100 ms offset, validating delay estimation. Resampling experiments (Fig. 3) showed, in a controlled way, why anti-alias filtering is required before decimation and why zero-insertion interpolation still needs a reconstruction low-pass. Spectral analysis via FFT (Fig. 4) and a small-K DFT (K = 32, Fig. 5) were mutually consistent—the former offering resolution, the latter acting as an

independent sanity check. Adding 12 dB AWGN (Fig. 6) illustrated how noise raises the floor and roughens the time trace. Finally, the AM chain (Fig. 7) completed the loop: modulation shifted content to ±fc, product demodulation returned it to baseband, and a 101-tap windowed FIR suppressed the 2fc image with minimal phase distortion using forward-and-reverse filtering.

Practically, the results highlight the engineering “knobs” that matter: STFT window/hop, anti-alias and reconstruction filters for rate changes, and FIR order/cutoff in the AM stage. Limitations (intentional for pedagogy) include a fixed 8-s segment and simple filters. Next steps are to (i) insert explicit anti-alias/reconstruction filters around the ×3/×2 resampling, (ii) sweep parameters (window, hop, FIR order/cutoff) with objective metrics (e.g., SNR improvement; for speech, PESQ/STOI), and (iii) test varied material (speech vs. harmonic music) to gauge robustness. Overall, the pipeline is a clear, extensible baseline for denoising, VAD, and modulation experiments.

References

[1] A. V. Oppenheim and R. W. Schafer, Discrete-Time Signal Processing, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2010.
[2] R. G. Lyons, Understanding Digital Signal Processing, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2011.
[3] J. G. Proakis and D. G. Manolakis, Digital Signal Processing: Principles, Algorithms, and Applications, 4th ed. Upper Saddle River, NJ, USA: Prentice Hall, 2006.