LAB: Audio Signal Processing with Octave

# 1. Objectives

* Understand the basic concepts of audio signals and how they are represented in the time and frequency domains.
* Practice reading, playing, and processing audio signals using GNU Octave.
* Apply simple filters such as high-pass and low-pass filters.
* Create audio effects such as echo or distortion.

# 2. Preparation

* - Install GNU Octave: https://www.gnu.org/software/octave/  
  - Install audio signal processing packages:

pkg install -forge signal  
pkg install -forge audio  
pkg load signal  
pkg load audio

* - Prepare a sample `.wav` file (can be downloaded from freesound.org)

## 3. Step-by-Step Instructions

## Read and Play Audio Signal

Read file: `[x, fs] = audioread("sample.wav");`

Play sound: `sound(x, fs);`

Plot in time domain:

t = (0:length(x)-1)/fs;  
plot(t, x);  
xlabel("Thời gian (s)");  
ylabel("Biên độ");  
title("Tín hiệu âm thanh trong miền thời gian");

## Fourier Transform and Spectrum Analysis

Do Fourier transform and plot:

X = fft(x);  
f = (0:length(X)-1)\*fs/length(X);  
plot(f, abs(X));  
xlabel("Tần số (Hz)");  
ylabel("|X(f)|");  
title("Phổ tín hiệu âm thanh");

## Apply Filters

Lowpass filter:

fc = 1000;  
[b, a] = butter(6, fc/(fs/2), "low");  
y = filter(b, a, x);  
sound(y, fs);

High-pass fileter:

[b, a] = butter(6, fc/(fs/2), "high");  
y = filter(b, a, x);  
sound(y, fs);

## Create Echo Effect

delay = round(0.3 \* fs);  
atten = 0.6;  
echo\_signal = x;  
echo\_signal(delay+1:end) += atten \* x(1:end-delay);  
sound(echo\_signal, fs);

## Save Processed Audio

audiowrite("output.wav", echo\_signal, fs);

## 4. Extended Exercises

1. Compare the original audio with the low-pass and high-pass filtered versions. Plot and analyze the frequency spectrum.
2. Design a bandpass filter to retain frequencies from 1000Hz to 3000Hz. Listen and give feedback.
3. Create a robot voice effect by multiplying the signal with a high-frequency sine wave.
4. Record your voice, apply echo and filtering to simulate a 'cave' effect.