BELMONT UNIVERSITY, CURB COLLEGE OF ENTERTAINMENT AND MUSIC BUSINESS

AET 5420 Homework 2

Audio Signal Processing

February 28th, 2024

SYSTEMS WITH MEMORY

Many audio systems are based on storing the samples of a signal in memory to delay them over time. Examples of these systems include: echo effects and spectral filtering. These effects are created by a linear combination (addition operation) of the current and previous samples in a signal.

1 TEMPO SYNCHRONIZED ECHO

One audio effect created by using time delay is the echo effect. In this effect, a signal is delayed for a longer time period than a listener's echo threshold (~40 ms). When the signal is delayed from 40-150 ms, the effect is described as a *slap echo*, reminiscent of creating the effect using a tape machine. When the signal is delayed by more than 150 ms, it is common to synchronize the timing of the delayed with the rhythm of the input signal. This prevents the effect from sounding "out-of-rhythm." Tempo synchronized delay is typically accomplished in software by interpreting the tempo of a song in beats per minute (BPM).

1.1 STEREO PING-PONG ECHO

A special type of echo effect is the stereo ping-pong delay. In this effect, the delayed repetitions of the signal alternate between the left side of the stereo field and the right side of the stereo field. It is common to use feed-back delay with the ping-pong effect to have slowly decaying repetitions on each side. An example of a block diagram for a ping-pong echo is shown in Fig. 1.1.

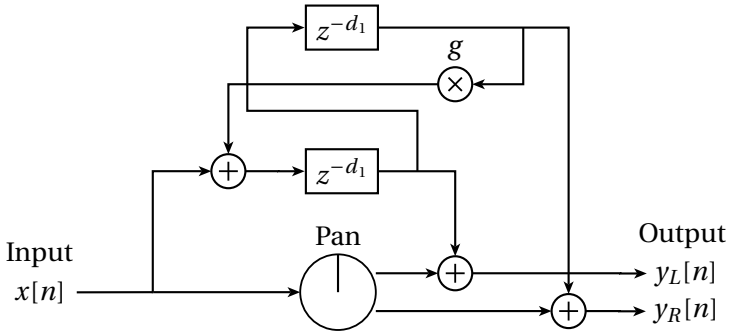

Figure 1.1: Block diagram of ping-pong echo

This block diagram creates an echo effect which alternates in a pattern (L,R,L,R,L,R,...) based on the length of delay, d_1.

1.2 Problem

For this problem, you will create a *function* (m-file) in MATLAB to create a tempo-synchronized stereo ping-pong echo with a **different pattern (L,C,R,L,C,R,L,...)**. Additionally, you will submit a drawing of the block diagram to create this effect.

- Name the function - pingpong.m
- The output variable, y, of the function should be the processed, stereo audio file.
- There should be five (5) input variables
  - x : mono signal to be processed
  - Fs : sampling rate
  - BPM : tempo in beats per minute
  - note : note duration (4 = whole, 2 = half, 1 = quarter, 0.5 = 8th, 0.25 = 16th)
  - feedback : amount of feedback as a percentage (0-100)
  - wet : control for dry/wet (% wet, where dry = 100-wet).
- At the start of the function, determine the delay time in samples for the effect.
  - Convert BPM to samples per second
  - Make sure to factor in the note duration
- Next, convert the feedback percentage to a feedback gain amount (0-1).
- Use a loop to create the output signal based on the audio effect
  - Index the individual samples of the input and output signals while creating the effect

- Use the space below to draw a block diagram of the effect to visualize what signals are combined for the left and right channels
- You should create additional variables representing the delayed signal in the left channel and right channel
- Use square-law panning functions when determining the amplitude of the various signals in the left and right channels
- The loop should create an output signal twice the length of the input signal, allowing for the tail of the echoes to decay

Use the provided test script, **pingpongTest.m**, and sound files, **click60.wav** and **click89**, to test your function. You should change some of the values in the test script to verify your function works properly. Remember to add comments to your code to explain what each command is accomplishing.

Use the following space for drawing the block diagram:

2 TIME-DOMAIN CONVOLUTION

Convolution is the mathematical operation of processing a signal going through a system represented by an impulse response. Essentially, each input sample, $x[n]$, is independently sent though the system, $h[m]$, to contribute to the output signal, $y[n]$, at different times. Therefore, the output signal is created by summing together a sequence of impulse responses due

to each input sample. In Fig. 2.1, a block diagram is shown for the relationship between the input and output of a linear system. The mathematical operation for convolution is written as: $y[n] = x[n] * h[m]$, where the symbol $*$ is used to denote convolution, and not multiplication.

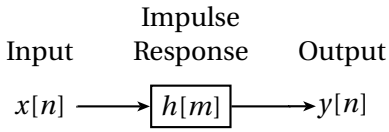

Figure 2.1: Block diagram of a generalized linear system

In Fig. 2.2, an example is shown relating a sine-wave input signal, system impulse response, and processed output signal.

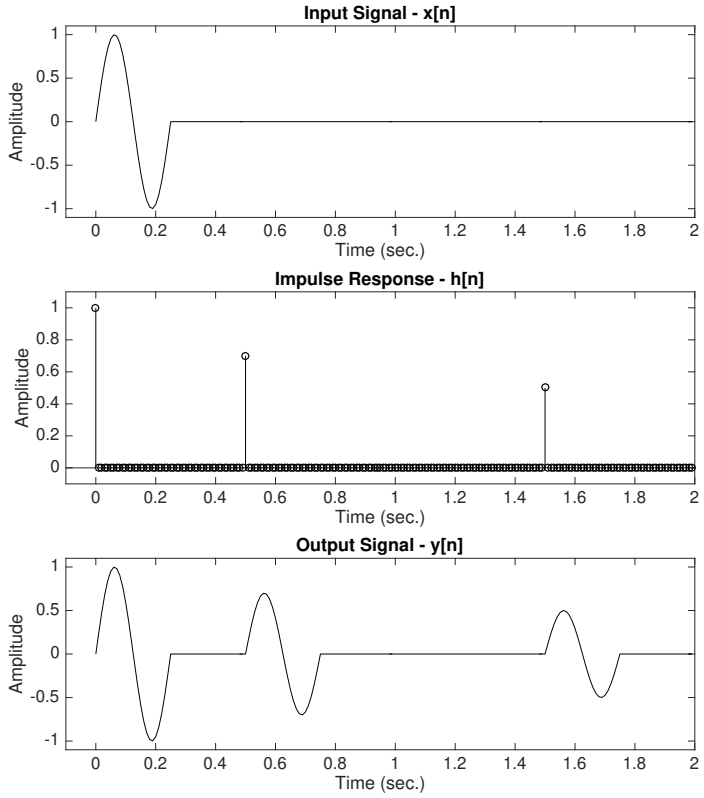

Figure 2.2: Visual example of convolution

2.1 Mathematical Equation for Convolution

The following equation for convolution can be used to determine the amplitude of the output signal, $y[n]$, at any sample number, $n = 1,2,3,...$

$$y[n] = x[n] * h[m] = \sum_{m=1}^{M} x[n-m+1] \cdot h[m], \text{ where M = } \texttt{length(h)} \tag{2.1}$$

Conceptually, the input signal is fed through the system by starting with the first sample and iterating to the end. This is accomplished by time-reversing the input signal then performing an element-wise multiplication with the processing system as the signal passes through.

2.2 Problem

For this problem, you will create a *function* (m-file) in MATLAB to perform the convolution operation. This function should work for signals, `x`, of an arbitrary length. Similarly, the function should work for an impulse response, `h`, with an arbitrary length. The syntax for your function should be: `y = userConv(x,h)`. Assume mono signals and systems. Your function should perform Equation 2.1 by using the inner product operation for each sample of the output signal. Use the provided test script to verify your function performs identically when compared to the built-in MATLAB function.

3 Frequency-Domain Convolution

An identical process to time-domain convolution can be performed using multiplication in the frequency domain. As it was shown in class:

$$\begin{aligned} X[k] &= \mathscr{F}(x[n]) \\ H[k] &= \mathscr{F}(h[n]) \\ Y[k] &= X[k] \cdot H[k] = \mathscr{F}(x[n] * h[m]) \\ y[n] &= \mathscr{F}^{-1}(Y[k]) \end{aligned}$$

Note, `length(y[n]) = length(x[n]) + length(h[n])-1`. Therefore, the frequency-domain version of each signal must have the same length of $y[n]$ in order to perform element-wise multiplication between the frequency bins. To accomplish this, the original time-domain signals should be zero-padded to the necessary length prior to performing the Fourier Transform.

3.1 Problem

For this problem, you will create another *function* (m-file) in MATLAB to perform the convolution operation using frequency-domain processing. This function should work for a signal, `x`, and an impulse response, `h`, with arbitrary lengths. The syntax for your function should be: `y = convFreq(x,h)`. Assume mono signals and systems. Your function should perform the sequence of steps listed above. Use the provided test script to verify your function performs identically when compared to the built-in MATLAB function.

4 SUBMISSION

To submit your homework, create a single zip file that contains your MATLAB functions, the test scripts, and figure files. Name the zip file: xxxxx_AET5420_HW2.zip, where *xxxxx* is your last name. Email the zip file to: eric.tarr@belmont.edu before the start of class on February 28th.