SASP

Homework 3

Homework 3 Report

Students

Riccardo Iaccarino
10868500
Xinmeng Luan
10876787

POLITECNICO
MILANO 1863

Introduction

The Leslie speaker is a famous amplifier and speaker used in electronic instruments. It utilizes a rotating chamber, often referred to as a "drum," positioned in front of the bass speaker. Additionally, there is a system of rotating horns in front of the treble driver. These elements combine to produce the distinct sound associated with this device. Musicians can manipulate the rotation speed of both the drum and horns using an external switch or pedal. They can switch between two settings: a slow speed known as "chorale" and a fast speed referred to as "tremolo."
We implement the Leslie speaker emulation in this homework using a computational efficient method provided in [1]. The flowchart in Fig. 1 shows the general idea of the implementation. As a digital processing, the Leslie speaker simulation operates on discrete-time audio signals. The input signal, $x[n]$, is sampled at a frequency Fs and undergoes processing to generate the output signal, $y[n]$. The simulation replicates the functionality of the crossover network, which splits the signal into two frequency bands.
Two separate filters are used to emulate the crossover network. These filters have identical cutoff frequencies and divide the input signal into bass and treble frequency bands, which corresponds to the rotation speed of the drum and horns controlled by an external switch or pedal that alternates between a slow and fast speed setting. Consequently, two distinct signal paths are formed, each dedicated to one frequency band.
Each band comprises two blocks: a frequency modulation block and an amplitude modulation block. The modulation signals, represented as Modulator 1 and Modulator 2, are shared between these two modulation blocks. The frequency modulation component is implemented using a spectral delay filter (SDF). The SDF consists of a series of N all-pass filters that are modulated by an external signal. The output signals from the SDFs are then subjected to amplitude modulation with the same modulation signal.
Finally, the two audio paths, representing the bass and treble bands, are combined by summing them together. This summation produces the final output signal, $y[n]$, which represents the simulated Leslie speaker effect.

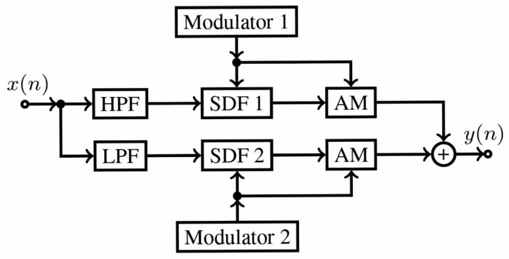

Figure 1: Flowchart of the Leslie speaker emulation.

Implementation Results

The `leslie.m` function has been implemented exploiting a system of buffers in which samples shift for each iteration in order to perform all the required operations, emulating a real-time audio effect. The two Butterworth filters of the crossover network have been modeled following the calculated difference equation with the Direct Form I. Similarly, the two SDFs blocks have been implemented combining the shift operation with a for loop that manages the summation. Finally the two amplitude modulation and the merge of bass and treble components are performed. The whole code is organized focused on its versatility, making possible any change in parameters and input (e.g. stereo signals can be handled). The results computed with the script `mainHW4.m` show that the output signal returns a MSE beneath the target threshold in both '`chorale`' and '`tremolo`' configurations, as shown in Tab.1.

	`chorale`	`tremolo`
MSE	3.113×10^{-10}	3.10989×10^{-10}

Table 1: Results of the implemented script.

Question 1

Vibrato is an audio effect consisting in a fast but slight variation in pitch around the played frequency. Despite this definition, which matches with the perceived effect, vibrato in digital audio processing is achieved by modulating the amount of delay in a delay circuit.

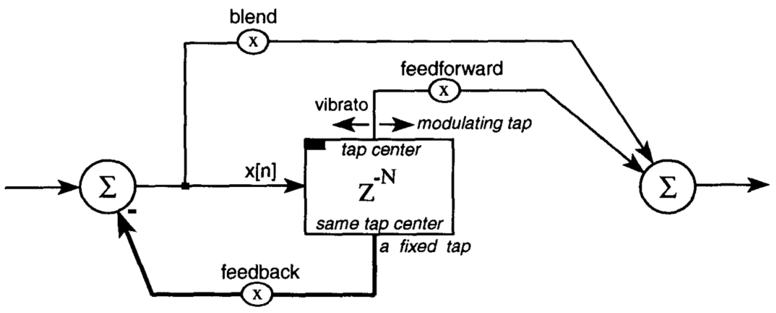

Figure 2: Industry standard audio effect structure.

One way of implementing this effect is described in the more general discussion of [2] about delay-based audio effects. In this study a modification of the industry standard chorus effect is proposed, with the adoption of the general scheme in Fig. 2 that can manage many different audio effects with just a few parameters. In particular, the vibrato effect can be obtained by putting to zero the **blend** and the **feedback** coefficient, hence leaving just the delayed signal, as shown in Fig. 3.

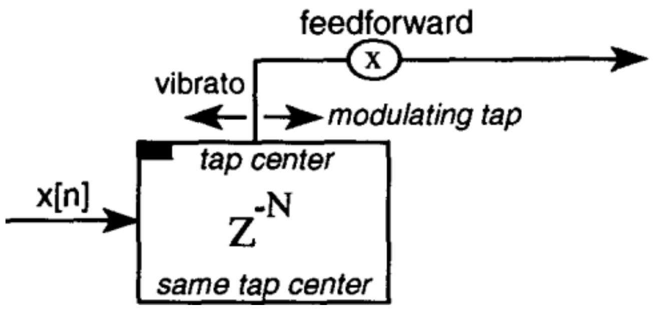

Figure 3: Vibrato effect computational scheme.

Given this structure, different design choices can be made about the type of delay implemented in the main block. Indeed the target solution is an ideal fractional delay, which is characterized by a unitary magnitude response and a constant group delay, and can be approximated with both FIR and IIR filters. From the first category we can find the least-squares FIR method which aims at minimizing the sum of the squared errors between the ideal and the approximated filter, making it a very intuitive solution but at the same time dangerous due to the Gibbs phenomenon that causes positive gains in the magnitude response. Another approach exploits window functions in order to truncate the ideal filter's impulse response, reducing Gibbs phenomenon but with a trade-off in terms of error. An improvement of these two solution is the General Least Squares (GLS) that windows the norm of the error in the frequency domain, which provides sensible improvements for the price of an increased amount of computational resources. The last suggested method involves Lagrange interpolation, whose goal is to make the error function maximally flat at low frequencies, imposing null derivatives; in this case the performance depends on the order of the filter with a clear distinction between odd (better phase) and even (better magnitude) ones.

Question 2

In our case the modulator signals of the two SDFs were in the form:

$$m'(n) = M_s m(n) + M_b \tag{1}$$

In order to discuss the stability of a chain of allpass filters we first have to understand the role of the modulator signal in the tranfer function of the filter itself. The expression of the latter for a cascade of N 1^{st} order allpass filters is:

$$H(z) = \left(\frac{a_1 + z^{-1}}{1 + a_1 z^{-1}}\right)^N \tag{2}$$

with a_1 being the modulator. Stability conditions can be found by looking at the poles of 2, which must lie inside the unitary circle in the complex plane. Moreover, the poles correspond to the modulator signal, so we can conclude that stability is obtained by imposing:

$$|m'(n)| < 1 \tag{3}$$

which is always true in our case, due to the values associated to the parameters M_s and M_b, but can be generalized for sinusoidal signals $m(n)$ as follows:

$$|M_s \pm M_b| < 1 \tag{4}$$

Question 3

The first all-pass filter in the chain of an SDF is

$$y(n) = m(n)x(n) + x(n-1) - m(n)y(n-1). \tag{5}$$

Its transfer function is

$$H(z) = \frac{a_1 + z^{-1}}{1 + a_1 z^{-1}}, a_1 \in [-1, 1], \tag{6}$$

where a_1 is the all-pass filter coefficient, and $a_1 \in [-1, 1]$ ensures the stability. The phase response is

$$\angle H(\omega) = -\omega + 2 \arctan\left(\frac{a_1 \sin\omega}{1 + a_1 \cos\omega}\right), \tag{7}$$

where $\omega = 2\pi f/Fs$ is the normalized angular frequency. The group delay is

$$\tau_g(\omega) = -\frac{\partial \angle H(\omega)}{\partial \omega} = \frac{1 - a_1^2}{1 + 2a_1\cos(\omega) + a_1^2}. \tag{8}$$

The diagram of the group delay of the first all-pass filter is shown in Fig. 4, with sampling frequency 44100 Hz.

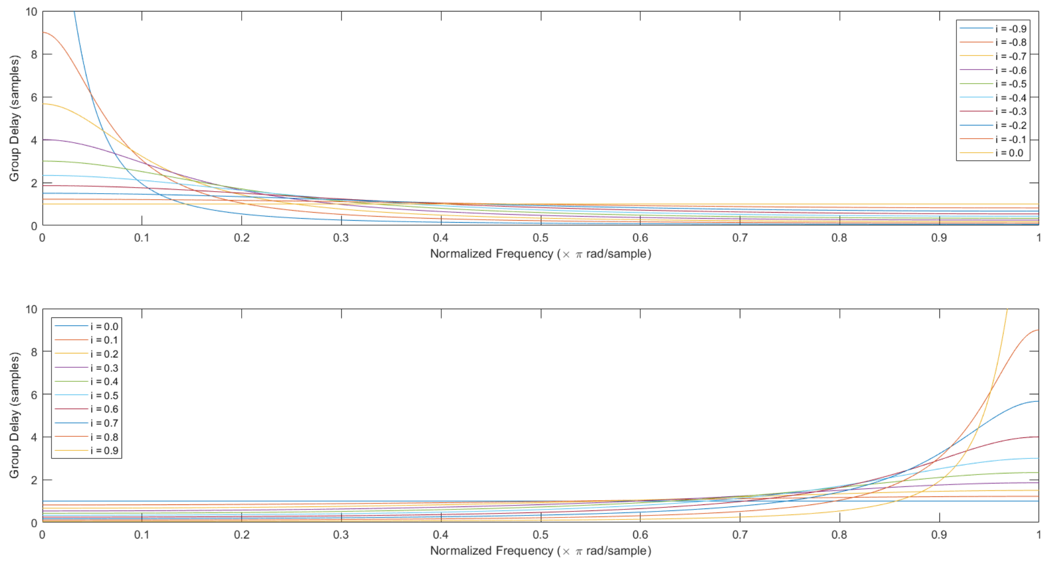

Figure 4: Group delay of the first all-pass filter for both negative and positive values of a_1.

First-order all-pass filters exhibit minimal group delay across different frequencies. Even in the case of the largest observed delay, which occurs when $a_1 = -0.9$, the delay is only 19 samples at low frequencies. This delay corresponds to approximately 0.4 ms at a sample rate of 44100 Hz. When an audio signal is filtered using this type of filter, the signal remains nearly unchanged. Even when the filtering process is repeated a few times, the audible effect is minimal.
However, if the same all-pass filter is applied repeatedly numerous times, such as 100 times, a noticeable effect can be heard. The repeated application of the filter causes a time separation between the low and high frequencies in the output signal, resulting in a chirp-like sound. This effect can also be observed by listening to the impulse response of the filter.
When the periodic signal with two spectral components, as described by the equation $u(n) = A_1 sin(\omega_1 n/Fs) + A_2 sin(\omega_2 n/Fs)$, with $\omega_1 \neq \omega_2$, is passed through an all-pass filter with the transfer function $H(z) = (a_1 + z^{-1})/(1 + a_1 z^{-1})$, the delay of both frequency components will not be equal in general. The phase shift introduced by the transfer function depends mainly on the modulator signal a_1, as shown in Fig. 5.

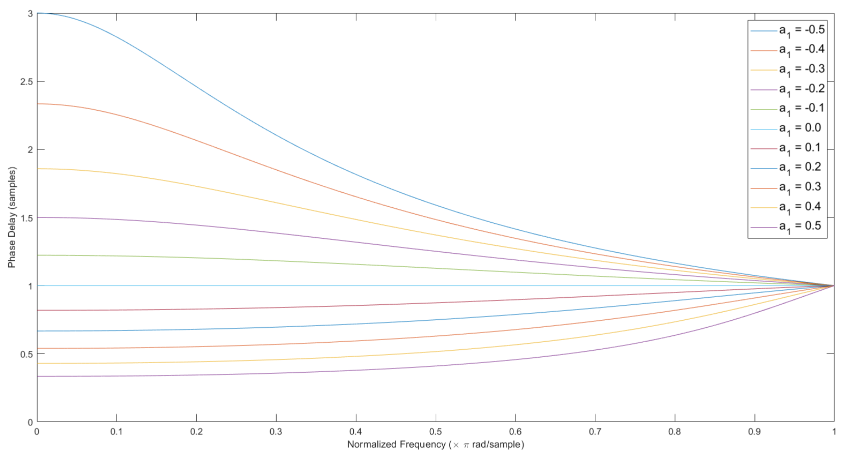

Figure 5: Phase delay of the first all-pass filter for different values of a_1.

Since we are seeking for a constant delay over frequency, we can choose optimal values of a_1 that approximate this condition. The ideal case would suggest $a_1 = 0$, but this gives us the integer delay filter $y(n) = x(n + 1)$, that may not be a case of interest. Indeed, if we want to implement a fractional delay using a first order all-pass filter, we have to select other values of a_1, which will reflect in the trend of the phase delay. From Fig. 5 we can observe that the more we get closer to the ideal case, the more linear the delay gets, especially at low frequencies. Hence we could assume similar delay for ω_1 and ω_2 only in a limited band of frequencies, that will depend on a_1 and on the tolerable error.

Question 4

When implementing a fractional delay using the Lagrange method, the most important parameter is the order of the filter, which depends directly on the target delay. For a delay $D = 3.3$ samples we must impose as a first condition that $N > \lfloor D \rfloor$, so we will start investigating our options from $N = 4$.

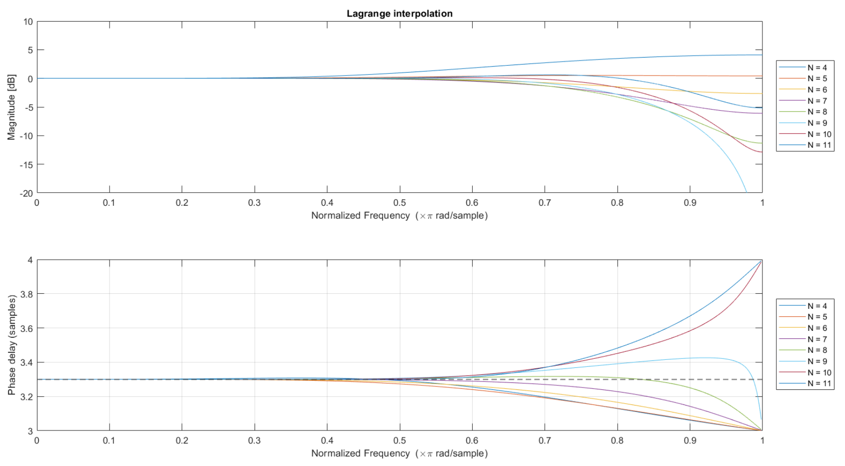

Figure 6: Magnitude responses (top) and phase delay curves (bottom) of a fractional delay $D = 3.3$ by Lagrange method.

We can see from the magnitude response and the phase delay of Fig. 6 that the accuracy of the fractional delay filter does not increase proportionally with N, argument that can be made instead for its complexity. In particular, with odd values of N we can observe a much more linear phase in spite of a worse magnitude response, whereas even orders have opposite behaviour. The excessive increase of the filter's order would just lead to instability of the magnitude and nonlinearity of the phase. We can also acknowledge these theoretical concepts by looking at the plotted results, indeed both the investigated quantities depart from the ideal values with high values of N.

Bibliography

[1] J. Pekonen, T. Philajamaki, and V. Valimaki. "Computationally Efficient Hammond Organ Synthesis". In: Proc. of the 14th Int. Conf. on Digital Audio Effects (DAFx-11) (Sept. 2011).

[2] J. Dattorro. Effect design part 2: Delay-line modulation and chorus. J. Audio Eng. Soc., 45(10):764–788, Oct. 1997.