Department of Electrical & Computer Engineering
McGill University

Experiment 1

Audio Effects in MATLAB

1.1 Purpose

In this experiment, you will implement a few basic audio effect algorithms in order to become familiar with the programming tools you will use during the class, namely MATLAB and SIMULINK.

1.2 Reverberation

In any natural environment, the sounds we perceive depend on the listening conditions, and in particular on the acoustic environment. The same signal played in a cathedral or in a small room will not "sound" the same. The room in which the listening experience takes place shapes the way any sound is perceived. *Reverberation* occurs when copies of an audio signal reach the ear with different delays and different amplitudes, after taking different paths and having bounced against walls and surrounding objects. Each room (with a fixed configuration of furniture and occupants) can be "measured" in terms of how it shapes the perception of sound. For a fixed position of the sound source and of the microphone, the different delayed and attenuated versions of the original sound can be thought of as being produced by a linear time invariant system. The amount of signal reflected off a wall or object depends on the absorption of the object. Walls treated with curtains or other sound-absorbing material will lead to less reverberation. The perceived signal at the location of the microphone[1] is

$$y[n] = \sum_i \alpha_i x[n - D_i] = h[n] * x[n], \tag{1.1}$$

where $h[n] = \sum_i \alpha_i \delta[n - D_i]$. One can measure the impulse response of a room by simply producing an impulsive audio signal and recording the signal captured by the microphone. For a large room, the impulse response can extend to a good fraction of a second and thus extend a large number of samples, particularly at high sampling rates.

In practice, simulated reverberations are not usually produced using impulse responses measured from actual rooms. The following sections will introduce basic building blocks to construct a synthetic reverberation tool.

[1] We assume sampled signals in this experiment.

1.2.1 FIR comb filter

An FIR comb filter with delay M is simply represented by an impulse response

$$h[n] = \delta[n] + G\delta[n - M],$$

where G is a scalar gain. Such a comb filter can generate one echo with a time delay corresponding to M sampling periods, and an attenuation/gain defined by G.

- Explain why such a filter is called a comb filter.

1.2.2 IIR reverberator model

Fig. 1.1 shows the system block diagram of a basic IIR reverberator, with two parameters, a feedback gain G and a delay M. For reasonably sized rooms, M is usually chosen so that it corresponds to a time delay between 25 and 100 ms.

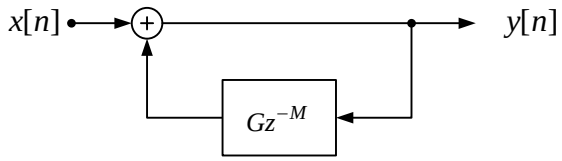

Fig. 1.1 Block diagram of a simple IIR reverberator building block.

- Compute the impulse response of a reverberator like the one in Fig. 1.1. Explain what kind of room geometry would produce such an impulse response.
- Compute the frequency response of the IIR reverberator. Use MATLAB to plot it for sample values of M and G. What are the conditions for stability?
- Implement an IIR reverberator building block both in MATLAB and SIMULINK. Replicate the block to create a number of reverberators in parallel, with different parameters for each reverberator.
- Especially in the case of speech signals, the simple IIR reverberator can produce sounds with a "metallic" quality. Try putting a simple FIR filter into the feedback loop to improve the "quality". Consider the case that high frequencies are absorbed (at a reflection point) more than low frequencies. What type of FIR filter should you use (lowpass, highpass, bandpass)?

1.2.3 Allpass reverberator model

An all-pass filter can give more substance to the reverberant signal and better simulates the sound in a reverberant room than the system considered above. An all-pass filter modifies the phase response, while leaving the amplitude response unchanged. A very easy parametrization of all-pass systems consists in using the same coefficients for the numerator and denominator, but in reversed order, such as e.g.

$$A(z) = \frac{a + bz^{-1}}{b + az^{-1}},$$

for any real a and b. This is a first-order filter. For use in a reverberator, the delay is changed to D samples, as shown in Fig. 1.2.

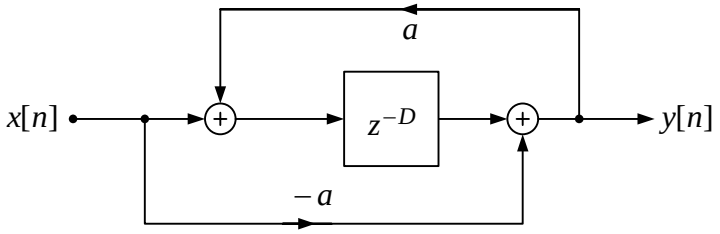

Fig. 1.2 Implementation of an all-pass reverberator transfer function.

- Comment on the effect of this impulse response on the signal and how it can approximate a reverberator. Check that the implementation in Fig. 1.2 yields an all-pass filter. To do so plot its frequency response. Also plot its impulse response. What are the conditions on a to ensure stability?
- Cascade a few allpass reverberators with the IIR-based system above and try to adjust the parameters so that you can simulate different types of rooms.
- Use an FIR filter to model the so-called "early-reflections". This filter can be placed in parallel with the above reverberator. In this case, the IIR/Allpass reverberator output should be delayed so that the early reflections reach the output before the IIR/allpass reverberated signals.

1.3 Ring Modulation

Ring modulation consists in modulating a signal so that the resulting signal is still in the audio range, but appears to have modified frequency content. The modulation consists in a multiplication by a sine wave of frequency f_0, so that in the output, the frequencies appear as shifts of the original frequencies by $\pm f_0$. Ring modulation is implemented by a multiplication in the time domain by a sinusoidal wave of frequency f_0.

In the extreme case, setting f_0 to half of the sampling frequency has the effect of inverting every second sample. The result is a frequency inversion, i.e. low frequencies become high frequencies and vice-versa.

- Let the sinusoid be $s[n] = \cos(2\pi f_0/f_s + \phi)$, where f_0 is set to half of the sampling frequency (f_s). Show that frequency inversion happens for this case. What are the requirements on the phase ϕ for this to occur?
- Implement a ring modulator. Test it on a speech signal. By adjusting the value of f_0, you will be able to make speech sound robotic.

1.4 Vibrato, Flanging and Chorus

The effects of vibrato, flanging and chorus are closely related in that they depend on the implementation of fractional delays.

1.4.1 Varying Fractional delay implementation

The three effects studied in this section rely on a time-varying delay element. The input-output relationship defining this element is given in time by:

$$y[n] = x[n - D[n]],$$

where the dependence of the delay D on the time index has been made explicit. In order to enable smooth variations of the delay $D[n]$, one often has to consider non-integer values of D. The values of a discrete-time signal between the sampling instants are obtained by interpolation. More precisely, the value of $x[n]$ at a non-integer time t is obtained by using ideal bandpass reconstruction of the underlying analog signal, and by taking the value $x(t)$ at the corresponding time instant:

$$x(t) = \sum_{k=-\infty}^{\infty} x[k] \frac{\sin \pi(t-k)}{\pi(t-k)}.$$

In practice, the infinite sum is replaced by a simpler interpolation. For instance, linear interpolation between sample points can be used. Fig. 1.3 shows a block diagram representation of a variable delay.

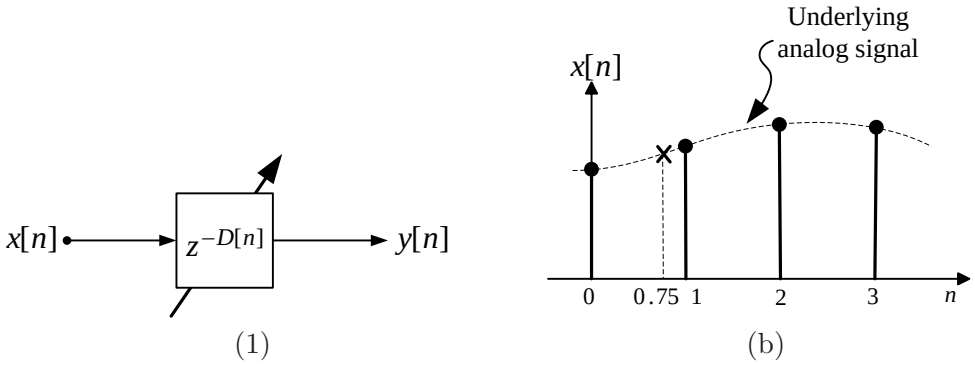

Fig. 1.3 (a) Block-diagram representation of a variable delay; (b) illustration of interpolation in the time domain, where one wants to acquire the signal value at time 0.75

1.4.2 Implementing the effects

Vibrato is an effect implemented directly by the structure in Fig. 1.3, where the delay $D[n]$ is periodically varied around an average value by a low-frequency oscillator (typically of frequency 5–15 Hz). This results in a periodic variation of the pitch (perceived frequency) of the signal. Typical average delay values are between 5 and 10 ms.

Flanging is obtained by an FIR comb filter where the delay is made variable, as illustrated in Fig. 1.4. The delay is again varied in a periodic way around some small value (less than 15 ms), with a low frequency oscillator (around 1 Hz). The resulting signal will be affected by a very distinctive periodic phase distortion.

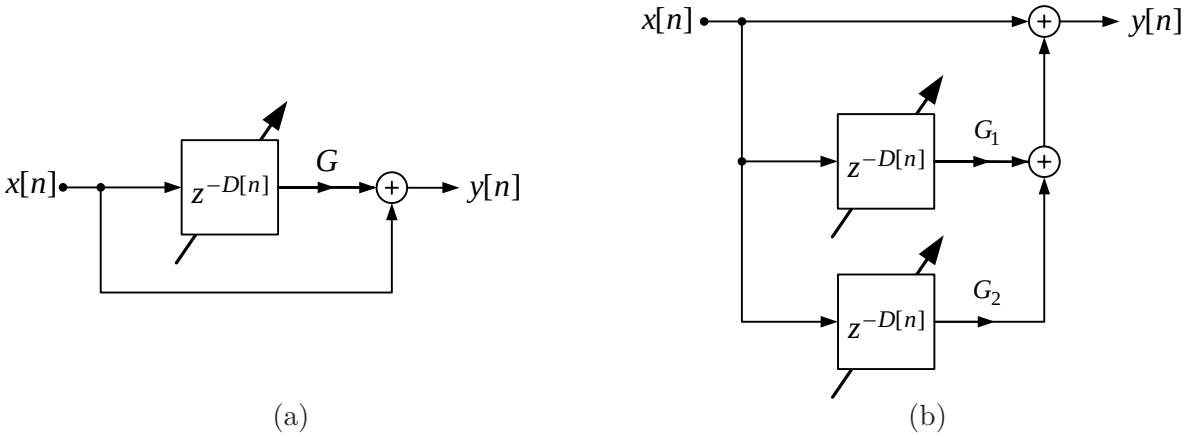

Fig. 1.4 Block diagram of a (a) flanging and (b) chorus effect implementation.

Chorus is an effect that simulates the presence of several sources playing in imperfect unison. It is implemented by combining the original signal with several copies delayed by a varied number of samples. Typical delays in the variable delay lines are 10 to 25 ms, and they are affected by small, slowly changing random variations. Fig. 1.4 shows a possible implementation.

- Implement all three effects in MATLAB.

1.5 Shelving Filter

Shelving filters are used to change the tonal characteristics of an audio signal. You will examine a lowpass boost filter. The filter is shown in Fig. 1.5. The filter $A(z)$ is an allpass filter of the form

$$A(z) = \frac{\alpha - z^{-1}}{1 - \alpha z^{-1}},$$

where the parameter α is determined by the normalized cutoff frequency,

$$\alpha = \frac{1 - \tan(\omega_c/2)}{1 + \tan(\omega_c/2)}.$$

The amount of boost will be determined by the value of K, where K varies between 0 and 1.

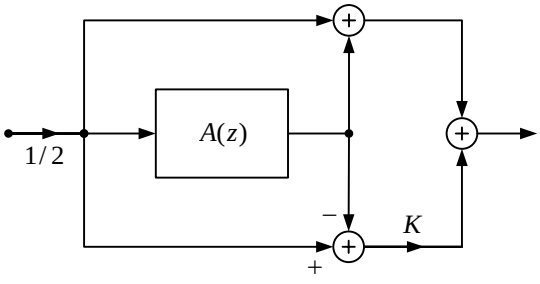

Fig. 1.5 Shelving filter

- Show that the frequency response of the shelving filter is of the form

$$H(z) = \frac{K}{2}[1 - A(z)] + \frac{1}{2}[1 + A(z)].$$

- Choose ω_c to give a cutoff frequency of 300 Hz and find the corresponding value of α. This value will depend on the sampling frequency of the audio signal. Note that not all of the audio files have the same sampling frequency.
- What is the amount of boost in dB for $K = 10$.
- Plot the magnitude of the frequency response for value of α found above and $K = 10$.
- Implement the filter and try it out on an audio file.

1.6 A Few Hints

- You will be processing a lot of data. Carefully write your MATLAB code so as to vectorize as much as possible most of your operations. Consider using functions such as `ones`, `zeros` and the operations `./`, `.*`, to avoid `for` loops. Moreover, try to use conditions on vectors: for example, `a(a<0)=0` will clip all negative values in vector `a` to 0.
- In order to load the audio files, use the MATLAB commands `auread` or `wavread`, depending on the file type. Note that some of your files have a stereo recording. They will give you a matrix with two rows or two columns. Select one of the channels for your processing, leaving out the other.
- In order to play a sound from MATLAB, use the command `sound`. Bring headphones to the lab in order to listen to samples. SIMULINK's DSP Blockset contains an output to WAV format block, that can also be directly output to the computer's line out.
- Write your code in reusable modules. For instance, the variable delay module appears several times; implement it as a reusable module.
- It is possible in MATLAB to program a flanging script (linear interpolation) without any `for` loop.