ECE 464 Computing Project 1

Thomas Snyder

October 2020

1 Introduction

This sequence of computing projects that will be completed through the term has the ultimate goal of studying the performance of a quadrature phase-shift keying (QPSK) transmitter and receiver pair. The implemented system will be a purely digital system. In the simulation, a random 2-bit sequence of will be generated, pulse-shaped, modulated, and transmitted over a simulated channel. A reception component will obtain the modulated signal, demodulate the signal, and recover the original 2-bit sequence by reversing the encoding process. Finally, experiments on performance of the digital system will be performed and the results are discussed in this report.

The project is broken into three modules. Module one encompasses generation of the signal to outputting the encoded and modulated signal to the simulated channel. The module also includes creation of the digital channel. Module two involves implementing and analog to digital (A/D) conversion to downshift the carrier frequency. Finally, module three requires simulating the receiver block and evaluating the performance of the whole system. This report is a working document throughout the term that will eventually include all three modules.

2 Methods

Sequence Generation The first section is generating the 2-bit sequence. A function was implemented, where given some number `N`, the function output two sequences of length `N`. The sequences $(b_1[n], b_2[n]$ were created according to the following specification:

$$b_1[n] = \begin{cases} 1 & x_1[n] \geq 0 \\ -1 & x_1[n] < 0 \end{cases}$$

and

$$b_2[n] = \begin{cases} 1 & x_2[n] \geq 0 \\ -1 & x_2[n] < 0 \end{cases}$$

where $x_1[n], x_2[n]$ are random zero-mean Gaussian distributions with variance of one. Generating the sequences was accomplished using the MATLAB `normrnd` function to generate x_1, x_2. To create b_1, b_2, each respective sequence was element-wise divided by the absolute value of itself. This algorithm will work for all values where x is not zero, but in the case where $x = 0$, the zero values are replaced with 1 to prevent division by zero. Replacing all zero values disrupts the distribution, making it slightly non-Gaussian, but the chances of the functions producing a zero are low, so this exception will not occur on a regular basis. The result of the division is b_1 and b_2. These sequences are then passed to the transmitter for pulse shaping and modulation. The generated sequence used in this experiment was 2000 samples long.

Pulse Shaping The first step to implementing the transmitter is to generate the pulse-shaping function. The pulse-shaping function will transform the impulses, which have infinite bandwidth in the frequency

domain, into a finite bandwidth signal. The chosen pulse-shaping function is known as the square-root-raised (SRR) cosine pulse, which is defined:

$$H_{rc}(e^{j\omega}) = \begin{cases} \sqrt{T} & 0 \leq |\omega| \leq \frac{\pi(1-\beta)}{T} \\ \sqrt{\frac{T}{2}\{1 + \cos[\frac{T}{2\beta}(|\omega| - \frac{(1-\beta)\pi}{T})]\}} & \frac{\pi(1-\beta)}{T} \leq |\omega| \leq \frac{\pi(1+\beta)}{T} \\ 0 & \text{Otherwise} \end{cases} \quad (1)$$

where I choose $\beta = 0.5$, and T is given as 4. This mathematical expression is implemented in its own function that takes β; an integer `N`, representing the number of samples; and T. The output is a SRR cosine pulse sampled with `N` samples. Note that this signal is bandlimited to $\frac{3\pi}{8}$, which makes for a band-limited pulse when an impulse is input to the system.

After the pulse was completed, I upsampled the input signals by a factor of 4, inserting three zeros after every pulse. This ensured that each pulse was sampled four times. The upsampled signal is then convolved with the inverse discrete Fourier transform of the SRR cosine pulse. Note that the inverse Fourier transform was sampled with 100 times the samples of the original SRR pulse in an attempt to mitigate time-domain aliasing. Once the convolution is complete, the sequences b_1, b_2 are now band-limited discrete time signals.

Upsampling Transmission in the channel and simulation as analog signals means that the discrete-time signals need to be upsampled once more to emulate analog signals in the digital system. In this case, they are upsampled by a factor of 20. But when a signal is resampled, it becomes periodic in the frequency domain and is scaled by $\frac{1}{T}$. Thus, a low-pass filter will be used to remove the periodicity and rescale the signal to its original magnitude.

The low-pass filter is implemented as a symmetric linear phase FIR filter. In short, the filter was created by setting the cutoff frequency to $\frac{\pi}{20}$ because the upsample will compress the original waveform by a factor of 20, and the new period will be $T = \frac{1}{20}$ while the original was $T = 1$. This creates the "analog" signal needed for transmission over the channel. This calculation was wrong, and the corrections are described in the **Results and Discussion** section.

Modulation The final step before transmission is modulation. b_1 will be modulated by a cosine wave and b_2 by a sine wave. Note that cosine and sine are $\frac{\pi}{2}$ radians out of phase, which is essential to preserve the information when the signals are summed. To modulate, each input sequence is simply elementwise multiplied by the respective modulation function. The modulation functions are computed with $\omega_c = 0.44\pi$ radians per sample. The MATLAB `cos` and `sin` functions are used to produce the necessary waveforms. Now the waveforms are sampled, and the finalized signal is ready to be output to the channel.

Channel Simulation The channel is simple, with additive zero-mean Gaussian noise with a variance of one. This is simply added into the signal to simulate the noise in the channel.

3 Results and Discussion

The first module resulted in a frequency-modulated waveform from a sequence of 2-bit symbols. These symbols were then output onto a channel with additive zero-mean Gaussian noise. Verification of the signal along the signal path can be done through analyzing the frequency response of the signal at various points. This section will focus on three main points: after the SRR cosine filter, after upsampling and filtering, and after modulation with a carrier frequency.

SRR Cosine Filter The cosine filter that interpolates the original impulse train sequence has a bandwidth of $\frac{3\pi}{8}$. The bandwidth of the signal after the pulse should not exceed $\pm\frac{3\pi}{8}$. Upon plotting the Fourier transform of the convolution between the input signal and the cosine pulse, the signal crosses the -2dB threshold at $\pm\frac{3\pi}{8}$. This can be seen in Figure 1. The time-domain pulse-shaped signal is shown in Figure 1.

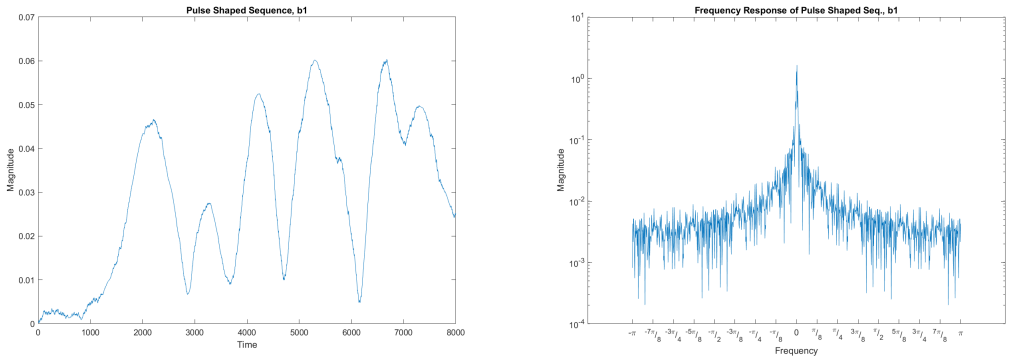

Figure 1: Sequence after SRR Cosine Pulse interpolation.

Upsampled Signal When a signal is upsampled by a factor, α, the frequency domain representation is expected to see a compression by a factor of α because of the scaling property of the Fourier transform. Additionally, the magnitude will be scaled by $\frac{1}{\alpha}$. The magnitude scaling can be compensated for and the frequency-domain periods that are created once the signal is sampled can be removed in an ideal low-pass filter (LPF). The upsampled and low-pass filtered signal in the time domain with LPF cutoff $\pi/20$ is shown in Figure 2. Note that this signal does look like the original, but is filled with harmonics; thus, it makes sense that the signal, when viewed in the frequency domain, still has periodic components (Figure 3).

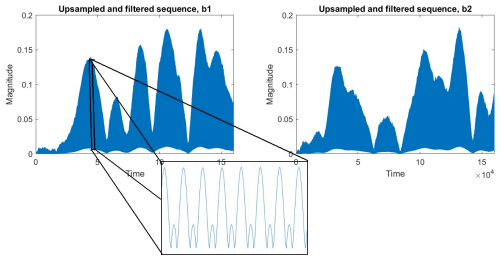

Figure 2: Signals after upsampling by a factor of 20.

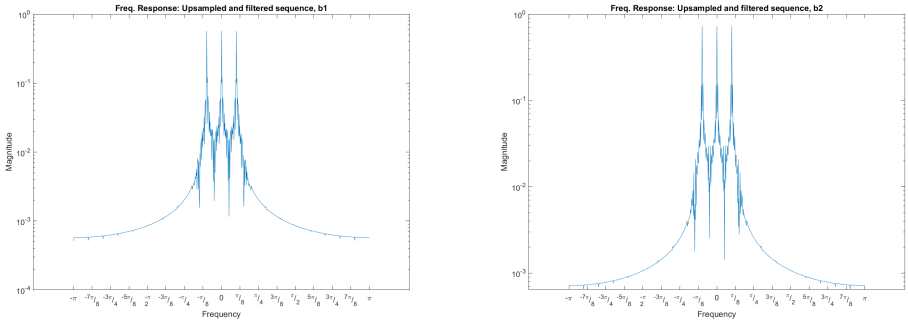

Figure 3: Frequency response of upsampled signal with LPF cutoff frequency $\omega_c = \frac{\pi}{20}$.

Decreasing the cutoff frequency of the LPF to $\omega_c = \frac{\pi}{40}$ yielded more successful results, but it was clear when looking at the frequency-domain signal that the artifacts of upsampling had not been completely removed. Upon carful consideration, it was observed that the signal had actually be upsampled by a factor of 80 from its original sequence (once by a factor of 4 and then a second time by a factor of 20). Thus, a cutoff frequency of $\omega_c = \frac{\pi}{80}$ was used. This successfully removed the artifacts in the upsampling. The magnitude multiplication factor of the LPF was also corrected to be 80. The final results of the upsampled signal are shown in Figure 4.

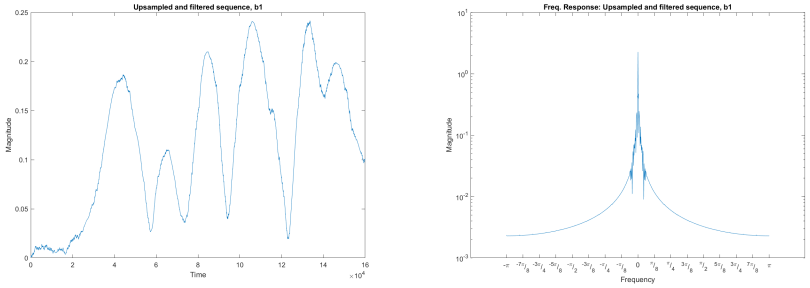

Figure 4: Frequency response of upsampled signal with LPF cutoff frequency $\omega_c = \frac{\pi}{80}$.

Signal Modulation with sine and cosine When a signal $x[n]$ is modulated with a carrier signal $c[n]$, the result is simply the multiplication of the two signals: $m[n] = x[n]c[n]$ where $m[n]$ is the message signal. If $c[n]$ is a sinusoidal signal with frequency ω, the signal can be broken into the sum of two complex exponentials. Thus, the resulting signal $m[n]$ is a shifted version of $x[n]$ in the frequency domain by the carrier frequency. When using a carrier frequency $\omega = 0.44\pi$, it is observed that modulation has been performed correctly because the Fourier transform of the modulated signals are versions of the Fourier transform of the input upsampled signal (see Figure 4), but shifted by $\pm 0.44\pi$ (see Figure 5).

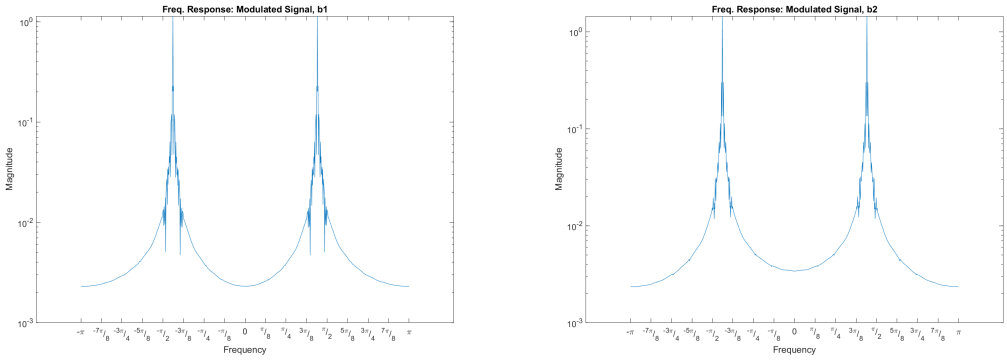

Figure 5: Frequency response of sequence after modulation with cosine and sine.

4 Conclusion

I put into practice my knowledge of sampling and filtering to create a system that successfully samples and modulates a discrete-time signal. The module required knowledge of sampling theorem, FIR filter design, and signal modulation. The new material I gained was on windowing methods for FIR filter design. An ideal filter is not feasibly possible, so windowing and different windows change the abruptness of passband behavior. The tradeoff essentially boils down to two variables: more abrupt, but more oscillation after the

cutoff; or, less abrupt, but less oscillation after cutoff (speaking of the frequency response). The cutoff frequency and amplitude of the FIR filter was also carefully considered, drawing on knowledge of sampling theory. I noticed that when the cutoff frequency was decreased, the time-domain signal compressed. By this observation, I was able to correct my original design of the low-pass filter and implement the correct cutoff frequency such that the signals looked identical before and after upsampling.