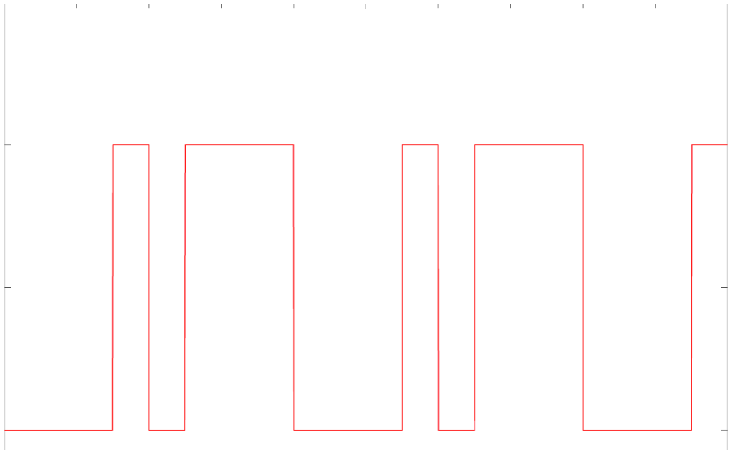
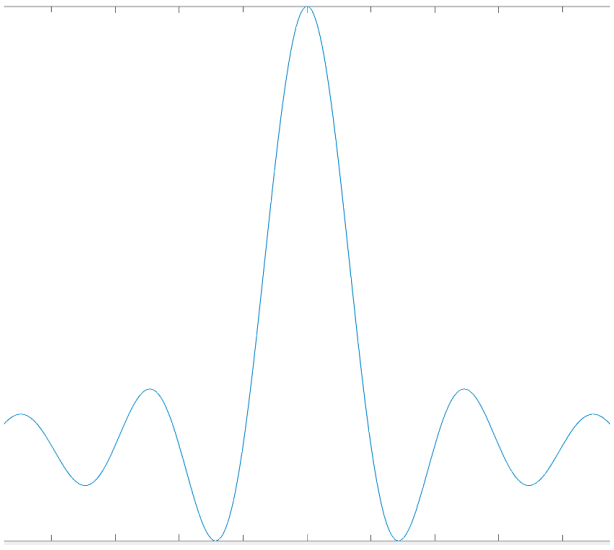
Pulse Shaping

Raw Data and Bandwidth

To understand pulse-shaping we must first take a step back and start from bitstream generation. The bitstream is a message composed of 0s and 1s e.g. binary data. The bitstream can be directly represented as a set of 0 and unit impulses which when combined together form a connecting series of rectangular pulses. Ex.



However, the issue with directly using these rectangular pulses deals with bandwidth. Bandwidth is the frequency range occupied by a signal, and in many cases the allowable transmission bandwidth is limited. Using rectangular pulses would create an infinite bandwidth signal since the transform of a rectangular pulse in the time domain to the frequency domain is a sinc function which has infinite bandwidth. Ex.



The Issue of Band-Limiting

The question then is “How do we create a filter which can band-limit our signal?”. To achieve this, many methods can be used such as using a lowpass and highness filter to get rid of the unwanted frequencies, however there is an issue with this approach. When we start chopping

off frequencies from the sinc function we can achieve our desired bandwidth, but when we transform back to the time domain our rectangular pulse functions can begin to overlap. The more we band-limit our signal, the more overlap we are likely to get. Since the rectangular pulses represent our data, overlap in the pulses means data will start to overlap; this type of interference is called ISI (Inter-Symbol Interference), and it is akin to 'corrupting' the data. A way to visualize ISI is to imagine as the frequency domain gets more band-limited by our lowpass and highness filters, the rectangular pulse starts smoothen out, almost like a block of ice melting. If we put two blocks of ice next to each other and melted them at the same time, eventually we would get a single puddle of water. There would be no way to recreate exactly both blocks of ice. Although crude, the ice example illustrates how the increasing overlap or ISI from the rectangular pulses makes it increasingly hard to recover the data at the received end.

Raised Cosine and SRRC Filters

We must refine our previous question to, "How do we band-limit our signal while incurring minimal ISI?". This is where the Nyquist pulse and raised cosine take the stage. The Nyquist pulse is a theoretical sinc function in the time domain which is set up so that for a given data rate, each symbol of data does not interfere with each other. E.g. for every point that we sample our signal in order to recover one piece of data, every other piece of data is at 0, resulting in 0 ISI. The transform of a sinc in the time domain is a rectangular pulse in the frequency domain, thereby providing both the 0 ISI and the band-limiting functions that we require. However, Nyquist pulses have been historically very complex to implement (although some newer methods have been found see: https://www.nature.com/articles/ncomms3898), therefore we turn to the raised cosine. The raised cosine filter is much easier to implement, and while by itself it does not reach the ideal 0 ISI for most practical applications, it can come very close. The raised cosine filter looks extremely similar to the Nyquist pulse sinc function and functions essentially the same. In the frequency domain the raised cosine takes a rectangular shape with a given roll-off. At 0 the frequency pulse is rectangular, and as the roll-off factor alpha increases the smoother or more 'melted' the shape becomes, yielding the raised cosine a higher degree of flexibility. To get a higher probability of 0 ISI while still using raised cosine filters we turn to the square root raised cosine filter (SRRC filter) sometimes called just the root raised cosine filter (RRC filter). The SRRC filter can be used in matched filtering. Matched filtering is when both the transmitter and receiver use the same known filter and the same sample rate on the signal. This serves to allow the receiver to know exactly where the data has 0 ISI and in theory completely recover the data. In practice, it is harder to achieve exactly 0 ISI, but a matched filter like the SRRC filter is good enough for many circumstances.

Applications to Our Project

To filter our data we convolve it with the SRRC filter which yields a signal which we can then up-convert and transmit. We must find a balance between the roll-off factor and the ISI as decreasing the roll-off factor increases ISI, but increasing the roll-off factor increases the amount of bandwidth used. Once we create the SRRC filter and transmit our data, we will utilize and SRRC filter with the same properties at the receiver to recover our data symbols e.g. matched filtering.