ECE4634 - Digital Communications
November 30, 2020
Nima Mohammadi
nimamo@vt.edu

Modulation

Design Project 3
Technical Memorandum

I. Introduction

In this technical report we will further investigate the impact modulation on transmission of voice signals. As opposed to the last report however we use the same BPSK modulation scheme across our tests and make our analysis based on the impact of pulse shaping, matched filtering and modeling a Rayleigh channel on the performance of our system. Again, we are working with the 5.86 second voice clip as our input signal.

We start by converting our input "analog" signal to a digital one through sampling and quantization. Our sampling rate of $f_s = 8192Hz$ was shown previously to be adequate for the voice recording based on Nyquist sampling rate and the fact that a bandwidth of 3.685kHz holds 99.9% energy of this signal. We are also using $\mu-$law to quantize the signal value into 64 levels represented by 6 bits. We have shown previously that this nonlinear quantization scheme outperform a linear method and both qualitatively and quantitatively confirms its efficiency with the number of required bits compared to a naïve linear approach. Upon this process we transform the signal to a bitstream of 0s and 1s.

In the next section we study various pulsing shaping in transmission of BPSK modulated signals and then continue our discussion with matched filtering and then with Rayleigh and Ricean fadings.

II. Pulse Shaping

We start by observing the impact of pulse shaping on the modulated signal. Pulse shaping is the process of changing the waveform of transmitted pulses in order to increase the efficiency of the required bandwidth of the transmission. We perform all of our tests on a BPSK-modulated signal of the voice recording. The four pulse shapes we see in this report are:

- Square Pulse (SQAR)
- Sinc Pulse (SINC)
- Raised Cosine (RaCo)
- Square-Root Raised Cosine (SRRC)

The plots of resulting BPSK-modulated signal with these different pulse shapes are depicted in Fig. 1. The pulse duration for all these cases is equal to 10 and with truncation length of N=10 and roll-off factor of 0.25. The generated signal for SINC, RaCo and SRRC in fact look a lot like each other to the naked eye.

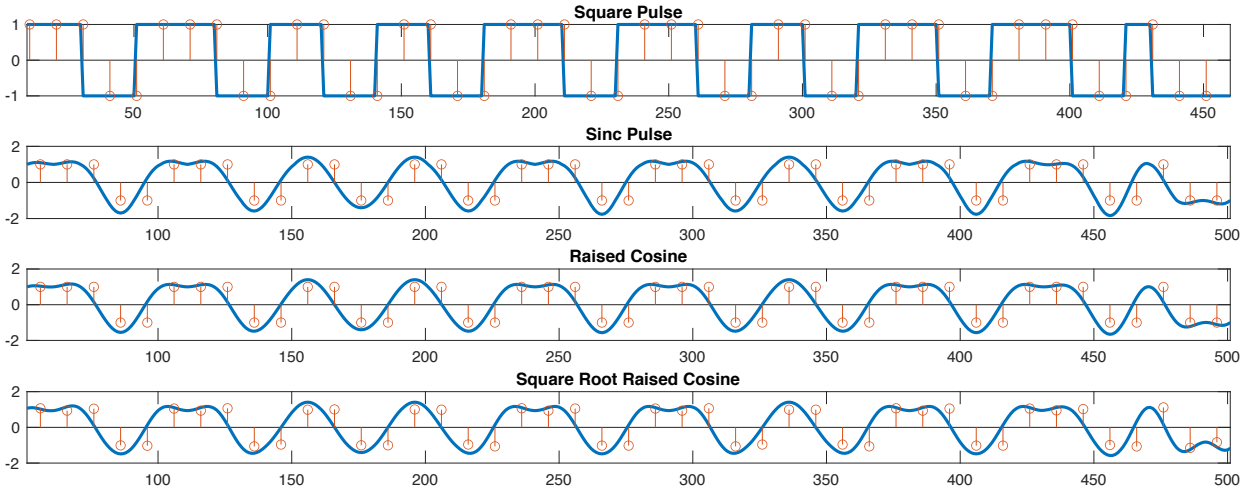

Fig. 1) Time plots of resulting BPSK-modulated signal with different pulse shapes

The energy spectral density for each of these pulse shapes is shown in Fig. 2. The right column, showing the ESD in decibels indicates the location of the first null and the peak of the first side lobe with red and blue asterisks, respectively. We are using this to find the first-null bandwidth

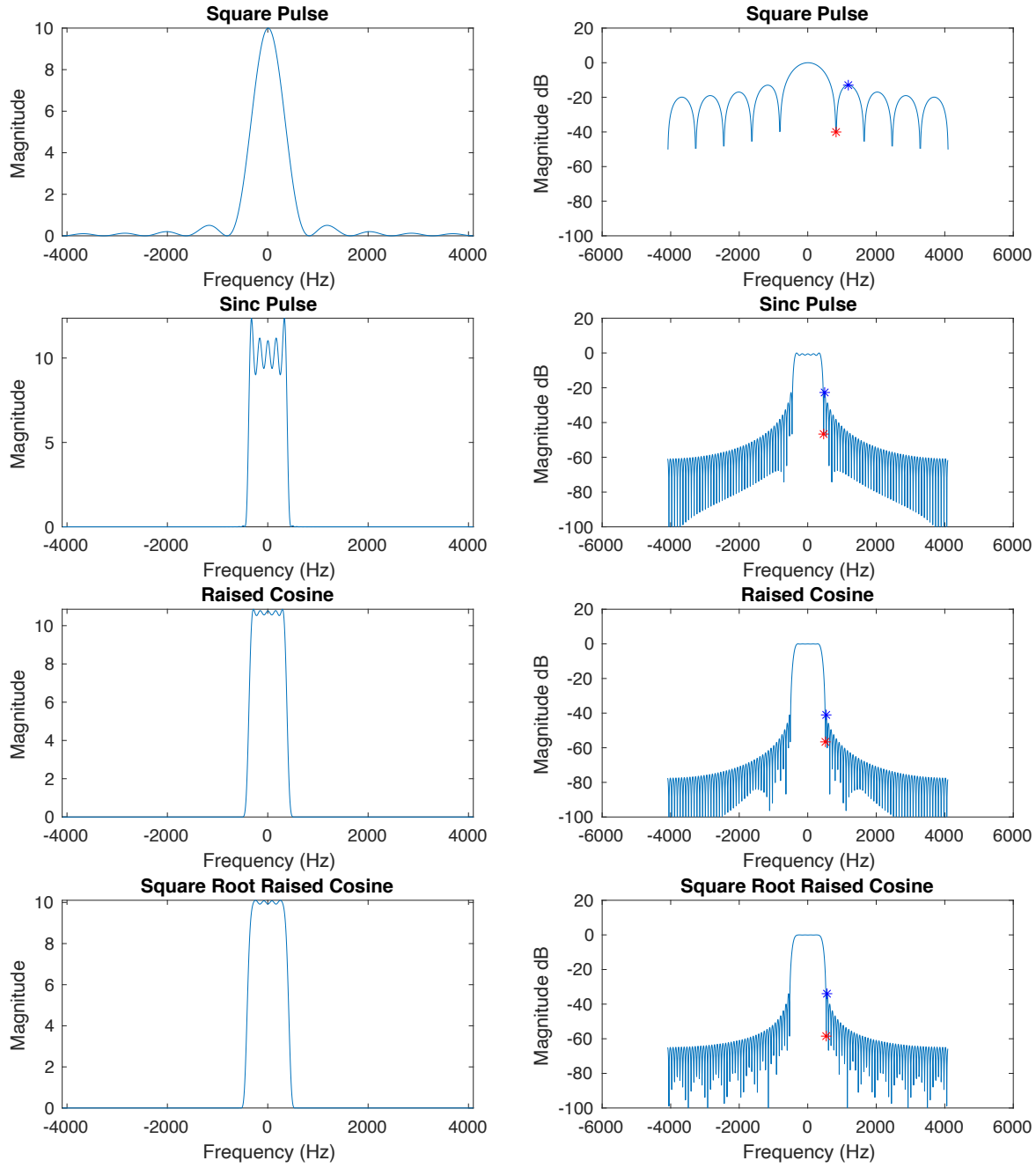

Fig. 2. Energy spectral density for different pulse shapes. Right column is in dB with first null and peak of first side lobe indicated in red and blue asterisks, respectively.

and the size of the first side lobe, that is the stop band attenuation. The 50dB bandwidth corresponding to the 99.999% energy bandwidth is also calculated. The calculated values are reported in Table 1.

The resulting ESD and the values reported in Table 1 let us compared different pulse shapes. As we expected SQAR has the highest bandwidth requirement and the size of the side lobe is considerably of larger magnitude. RaCo achieves the lowest 50dB bandwidth among these pulse shapes. Although its first null bandwidth is a bit wider, the first lobe is down by -41.03 dB which outperforms the other pulse shapes. Based on first-null bandwidth SINC is the winner here, however the stop band attenuation seems to be a more important criteria and suggests the superiority of RaCo.

	50dB BW	1st Null BW	dB down
SQAR	4055.45	1184.19	-12.97
SINC	1482.01	494.79	-22.70
RaCo	528.76	539.44	-41.03
SRRC	722.39	561.76	-33.97

Table 1. 50dB BW, first null bandwidth and the amount energy declines for the first side lobe for four different pulse shapes.

Bandwith is not right -5

Next, we focus on RaCo of different truncations *N* and roll-off factors *r* and compare them with SINC. The ESD for each configuration is depicted Fig. 3 and the results are reported in Table 2. The left column of the Fig. 3 shows the shape of the pulses with the two right columns being the ESD. We can see that by increasing the roll-off factor the envelope will decay faster and the first null bandwidth becomes larger. In fact, roll-off factor of zero is equivalent to the SINC pulse. Increasing N has also positively impacted the bandwidth requirement.

	N	Roll-off	50dB BW	1st Null BW	dB down
SINC	5		4064.79	510.54	-20.05
RaCo		0.25	1414.28	521.17	-28.82
RaCo		0.75	680.09	721.92	-51.62
SINC	10		1482.01	465.03	-22.70
RaCo		0.25	528.76	517.12	-41.03
RaCo		0.75	670.25	725.45	-59.41

Table 2.

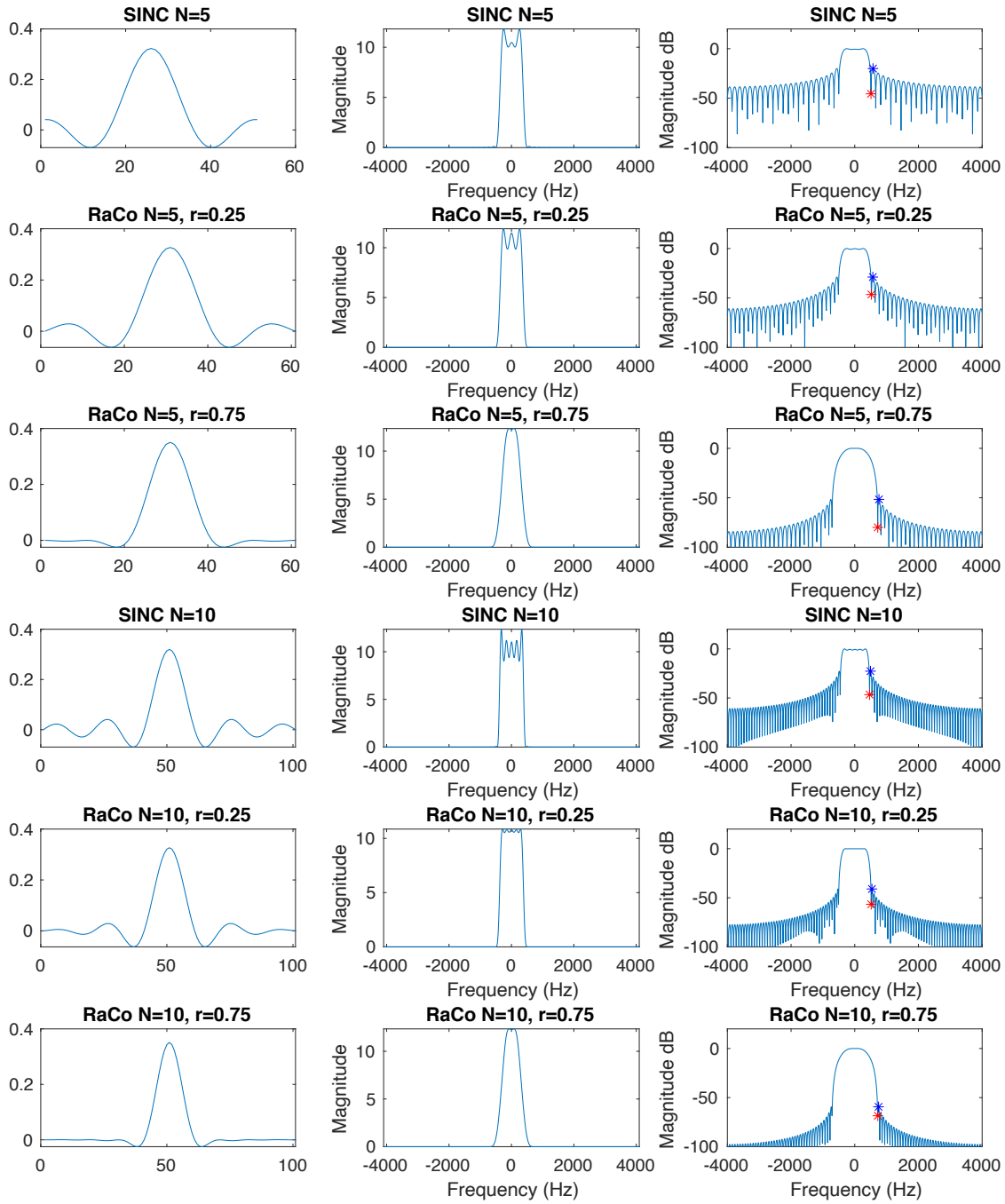

Fig 3. Comparison of ESD for SINC and RaCo of different N and roll-off factors.

In Fig. 3, the spectra and corresponding time series of various values of N and r are plotted. Larger values of r are characterized by a time-domain signal that has faster sideline decay rates. The good thing about SRRC is that it has no ISI, but the problem is that it has infinite support in time, which makes it difficult for a practical system as the pulses can not last indefinitely. This mandates the requirement for truncation of the pulse shape. We observe that with increasing N,

the passband ripples have been eliminated and the out-of-band sidelines are down by a larger amount. As the time span of the pulse is increased, the spectrum approaches the ideal spectrum. In general, the smaller the roll-off factor, the longer the pulse shape needs to be in order to achieve a desired stop-band attenuation.

In the next section, we study the impact of presence or absence of matched filtering and sampling at the optimal instances, using our usual performance evaluation metrics.

III. Matched Filtering

In this part we investigate matched filtering, using identical filter to what has been used in the transmitting side before sampling in the receiving side. Matched filtering is a process for detecting a known piece of signal or wavelet that is embedded in noise. The filter will maximize the signal to noise ratio (SNR) of the signal being detected with respect to the noise. We implement the integrator by convolving the received signal with the pulse shape.

In Fig. 4, we have the time plots of the BPSK-modulated signal with different pulse shapes. The blue and red curves show the signal before and after convolution for matched filtering, respectively. Also the the vertical black lines at discrete times indicate the demodulated bitstream at the optimal sampling time. The right column correspond to the case where a Gaussian noise with $E_b/N_0 = 7.5\text{dB}$ is added to the signal, whereas the left column shows the signal and demodulation in absence of noise. Clearly, adding noise will increase the BER, as shown in the figure.

Table 3 shows the performance metrics, BER and SNR for different settings. For the case where we have matched filtering and sampling is done at the correct optimal instances we observe BERs in the order of 10^{-4} with SRRC achieving the best BER. Interestingly, SINC achieves a better SNR despite worse BER. If the sampling distance is off by one timestep, the performance deteriorates. For all pulse shapes we would observe an order of magnitude increase in calculated BERs. The impact can also be seen in the corresponding SNRs. If we discard the integrator and perform sampling without matched filtering (convolution), we would vividly observe a sharp decrease in the performance. The BERs then become on the order of 10^{-1}, approximately misidentifying one bit out of every 10 bits. This clearly shows the importance of matched filtering, as the SNRs with negative dB values indicate.

	W/O Matched Filtering		Nonoptimal Sampling		Optimal Sampling		
	BER	SNR	BER	SNR	BER	SNR	Quality
SQAR	0.14422	-6.264	0.00161	11.430	0.00044	15.783	Good
SINC	0.14310	-6.226	0.00237	12.608	0.00077	17.110	Good
RaCo	0.13633	-6.055	0.00207	13.718	0.00086	16.186	Good
SRRC	0.13470	-6.119	0.00120	13.567	0.00038	16.705	Good

Table 3. BERs and SNRs for three cases 1) without matched filtering 2) With matched filtering but sampling performed non optimally 3) With matched filtering and optimal sampling

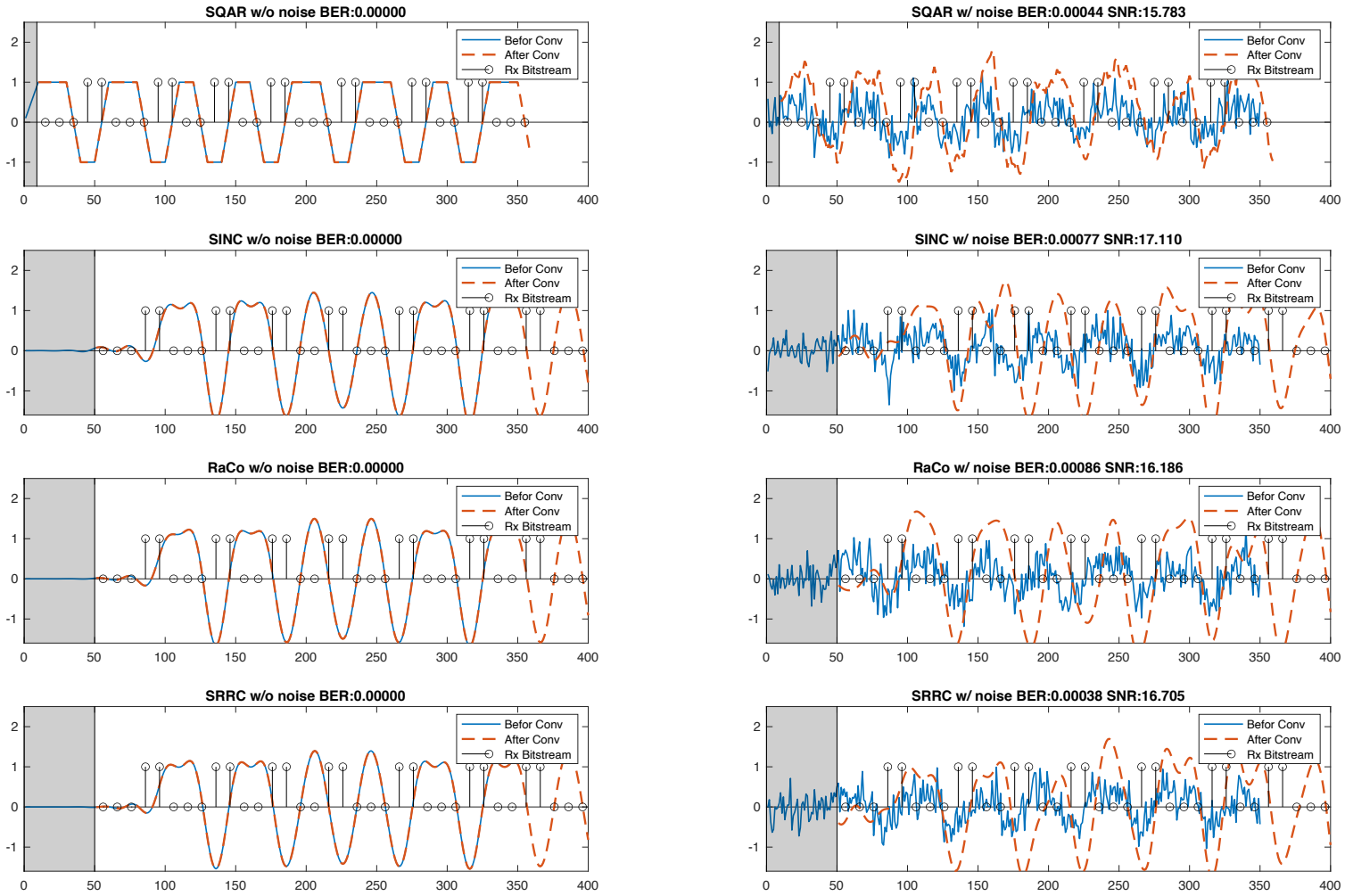

Fig. 4. Time plot of different pulse shapings on BPSK-modulated signal. Left shows the case without noise and right is the case where Gaussian noise has been added to the signal. The grayed area indicates the delay. The blue curve is the signal before matched filtering and the red curve is after performing convolution which are moved to overlay each other for better clarity. The demodulate bitstream are shown at discrete times with black vertical lines at the instances of optimal sampling. BER and SNR metrics are reported above the plots.

The fact that all BERs when we are using matched filtering at optimal sampling times across different pulse shapings confirm the theory that pulse shaping does not effect the P_e calculation (if we assume no ISI) for the matched filter, but it does change the integrator or matched filter implementation.

Using the theoretical results, we know that $P_e = Q\left(\sqrt{\frac{A^2T}{2N_o}}\right) = Q\left(\sqrt{\frac{E_b}{N_o}}\right)$, irrespective of the pulse shape used which is calculated to be around 0.0088 for $E_b/N_0 = 7.5$dB in our case. Although our empirical results although are closely approximated by this value (in an upper bound sense), they show some negligible variance that might have been caused mainly by the stochastic nature of the additive noise. We might get a better sense if we repeated the procedure many time and averaged the BERs for each case to cancel out the variations caused by a particular randomized noise.

We also demodulate and then converted the received signal to analog and calculated the SNR and listened to the resulting voice clip. In case of matched filtering, all pulse shapes resulted in an acceptable quality, however without matched filtering the analog signal suffers a lot from the noise, as evident by the negative dB SNRs reported in Table 3.

In the next section, we see the impact of fading in the quality of transmitted signal.

IV. Fading

At last we will investigate the impact of fading in our telecommunication system. In Rayleigh fading we have amplitude and phase distortion of the signal after being transmitted over the transmission channel. The fading phenomenon points to this situation where the signal power gets reduced significantly, causing deterioration of our performance metrics, BER and SNR.

In Table 4, the performance metrics for different fading channels are reported and compared with the case of an AWGN channel without fading. Our observations are compatible with the theory. We observe that increasing the factor *K* for Ricean fading, less fading occurs, as it is evident from the decline in BER. However, we see that changing the doppler shift from 200Hz to 10Hz does not have a pronounced impact on the BER, in case of Rayleigh fading. The theory states that for Rayleigh fading, we would observe lower BER for higher doppler shift, which is not the case here. But one may need to take into the account that the difference is low and might not be meaningful enough to draw conclusions. Ricean channels are characterized by a K factor that quantifies the amount of power in the LOS component relative to the multi path terms, where a larger K causes less fading. We also observe that Ricean fading causes less distortion compared to Rayleigh fading.

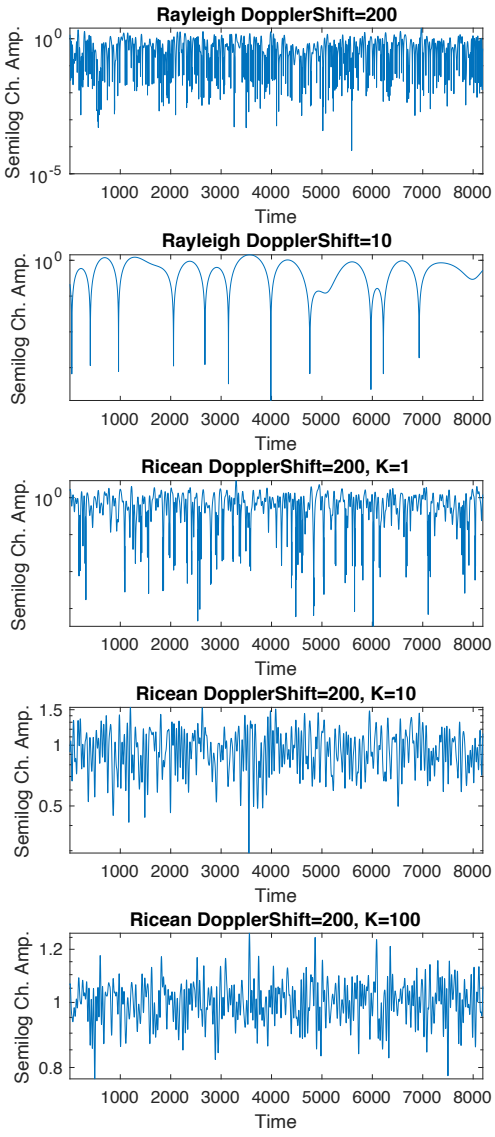

Fig. 5. Semilog plots of channel amplitudes for different fading schemes

Fading	Doppler Shift	K	BER	SNR	Quality
-	-	-	0.0004	15.7723	High
Rayleigh	200	-	0.0388	-1.1712	Low
Rayleigh	10	-	0.0371	-0.8503	Low
Ricean	200	1	0.0315	-0.2379	Low
Ricean	200	10	0.0038	8.9460	Good
Ricean	200	100	0.0006	15.5654	High

Table 4. Performance evaluation across different types of fading

V. Conclusion

In this report, we studied and investigated the impact of pulse shaping, matched filtering and fading by carrying out a set of simulations on transmission of a BPSK-modulated signal. We visualized the time plots and energy spectral density plots for four different pulse shaping techniques and used 50dB bandwidth, first-null bandwidth and stop band attenuation to compare them. We followed our discussion with the impact of matched filtering and performing integration and sampling at optimal sampling instances on the performance. We observed that type of pulse shaping does not impact the BER of received signal as long as matched filtering is properly performed, but that pulse shaping is needed due to its impact on spectral properties of transmitted signal. At last, we investigated Rayleigh and Ricean fading (without pulse shaping) with the case of a simple AWGN channel and reported quantitative and qualitative evaluations.

VI. Appendix (MATLAB Code)

```
% Nima Mohammadi
% Design Project 3

clc;
clear;

%% Load dataset
load DesignProject1
duration = length(Original) / 65536;

%% Sampling and Quantization -> bitstream
bits = 6;
fs = 8192;
inpsig = Analog2Digital(Original, fs, bits, 1, 255, 65536);
outsig = Digital2Analog(inpsig, bits, 1, 255);
% sound(outsig, fs);
sig_bpsk = PhaseMod(inpsig, 1, 0); %BPSK
```

```
% 1st Part) Pulse Shaping
%% Pulse Shaping
NS = 10;
N = 10;
roll_off = .25;

pulse_shapes = {'SQAR',... % Square Pulse
                'SINC',... % Sinc Pulse
                'RaCo',... % Raised Cosine
                'SRRC',... % Square Root Raised Cosine
                };
pulse_captions = {'Square Pulse',...
                  'Sinc Pulse',...
                  'Raised Cosine',...
                  'Square Root Raised Cosine'...
                  };

sigs_ps = cell(1, length(pulse_shapes));
pulse_ps = cell(1, length(pulse_shapes));
Es_ps = cell(1, length(pulse_shapes));

for i = 1:length(pulse_shapes)
    [sig_pulseshape, pulse, Es] = PulseShape(sig_bpsk, pulse_shapes{i},...
```

```
        NS, N, roll_off);
    sigs_ps{i} = sig_pulseshape;
    pulse_ps{i} = pulse;
    Es_ps{i} = Es;
end

%% Plot Signals

set(gcf,'PaperSize',[12 5]);
set(gcf,'position',[0 0 1000 360]);

for i = 1:length(pulse_shapes)
    subplot(length(pulse_shapes), 1, i);
    if i == 1
        dd = N-1;
        ddd = 1;
    else
        dd = N * NS / 2;
        ddd = 5;
    end
    plot(1+dd:451+dd, sqrt(Es_ps{i})*sigs_ps{i}(1+dd:451+dd),'LineWidth',2);
    hold on;
    stem(1+dd+ddd:10:451+dd+ddd, sqrt(Es_ps{i})*sigs_ps{i}(1+dd:10:451+dd))
    title(pulse_captions{i})
    xlim([1+dd 451+dd])
end
saveas(gcf, 'plot1.pdf')

%% Plot ESDs and calculate first-null BW
clc;
set(gcf,'PaperSize',[9 10]);
set(gcf,'position',[0 0 700 1000]);
for i = 1:length(pulse_shapes)
    subplot(length(pulse_shapes), 2, (i-1)*2+1);
    [Y, f] = EnergySpectralDensity([pulse_ps{i}, zeros(1,1000)], 8192);
    Y1 = abs(Y).^2;
    maximas = islocalmax(Y1);
    ylabel('Magnitude')
    title(pulse_captions{i})
    bw = CalculateBandwidth([pulse_ps{i}, zeros(1, 1000)], 8192, .99999);
    ind50dB_start = find(f > -bw, 1);
    ind50dB_end   = find(f > bw, 1);
```

```
    subplot(length(pulse_shapes), 2, (i-1)*2+2);
    EnergySpectralDensity([pulse_ps{i}, zeros(1,1000)], 8192, [-6000 6000 -100
20],1 );
    hold on;
    Y2 = 20*log10(abs(Y)/max(abs(Y)));

    maximas2 = Y2(maximas) < -5;
    ind1stLobe = find(maximas2(1:end-1) - maximas2(2:end) == -1)+1;
    tmp1 = find(maximas==1, ind1stLobe, 'first');
    ind1stLobe = tmp1(end);

    minimas = islocalmin(Y2);
    ind1stNull = find(minimas(1:ind1stLobe), 1, 'last');

    if i == 1
        plot(f(ind1stNull), Y2(ind1stNull+1), 'r*');
    else
        plot(f(ind1stNull), Y2(ind1stNull), 'r*');
    end
    plot(f(ind1stLobe), Y2(ind1stLobe), 'b*');
    ylabel('Magnitude dB')
    title(pulse_captions{i})

    fprintf("%s 50db-BW: %.2f \t null-bw:%.2f \t dB-down:%.2f\n",...
        pulse_shapes{i}, bw, f(ind1stNull), Y2(ind1stNull));
end
saveas(gcf, 'plot2.pdf')

%% Raised Cosine - Impact of N and r
NS = 10;
ps_raco = {'SINC', 'RaCo', 'RaCo', 'SINC', 'RaCo', 'RaCo'};
sigs_raco_N = [5, 5, 5, 10, 10, 10];
sigs_raco_r = [0 .25, .75, 0 .25, .75];

sigs_raco = cell(1, length(ps_raco));
pulse_raco = cell(1, length(ps_raco));
Es_raco = cell(1, length(ps_raco));
pulse_raco_cap = cell(1, length(ps_raco));

for i = 1:length(ps_raco)
    [sig_pulseshape, pulse, Es] = PulseShape(sig_bpsk, ps_raco{i},...
        NS, sigs_raco_N(i), sigs_raco_r(i));
    sigs_raco{i} = sig_pulseshape;
```

```
    pulse_raco{i} = pulse;
    Es_raco{i} = Es;
end

%% Calculate first-null BW and 50dB BW for SINC and RaCo w/ different N &r
clc;
set(gcf,'PaperSize',[9 10]);
set(gcf,'position',[0 0 700 1000]);
for i=1:length(ps_raco)
    if ps_raco{i}=='SINC'
        caption = sprintf('%s N=%d', ps_raco{i}, sigs_raco_N(i));
    else
        caption = sprintf('%s N=%d, r=%.2f', ps_raco{i}, sigs_raco_N(i),
sigs_raco_r(i));
    end
    subplot(length(ps_raco), 3, (i-1)*3+1);
    plot(1:length(pulse_raco{i}), pulse_raco{i});
    title(caption);

    subplot(length(ps_raco), 3, (i-1)*3+2);
    [Y, f] = EnergySpectralDensity([pulse_raco{i}, zeros(1,1000)], 8192);
    Y1 = abs(Y).^2;
    maximas = islocalmax(Y1);
    ylabel('Magnitude')
    title(pulse_raco_cap{i})
    bw = CalculateBandwidth([pulse_raco{i}, zeros(1, 1000)], 8192, .99999);
    ind50dB_start = find(f > -bw, 1);
    ind50dB_end   = find(f > bw, 1);
    title(caption);

    subplot(length(ps_raco), 3, (i-1)*3+3);
    EnergySpectralDensity([pulse_raco{i}, zeros(1,1000)], 8192, [-4000 4000 -100 20],1
);
    hold on;
    Y2 = 20*log10(abs(Y)/max(abs(Y)));

    maximas2 = Y2(maximas) < -5;
    ind1stLobe = find(maximas2(1:end-1) - maximas2(2:end) == -1)+1;
    tmp1 = find(maximas==1, ind1stLobe, 'first');
    ind1stLobe = tmp1(end);

    minimas = islocalmin(Y2);
    ind1stNull = find(minimas(1:ind1stLobe), 1, 'last');
```

```
    plot(f(ind1stNull), Y2(ind1stNull), 'r*');
    plot(f(ind1stLobe), Y2(ind1stLobe), 'b*');

    ylabel('Magnitude dB')
    title(caption);

    fprintf("%s 50db-BW: %.2f \t null-bw:%.2f \t dB-down:%.2f\n",...
        ps_raco{i}, bw, f(ind1stNull), Y2(ind1stLobe));
end
saveas(gcf, 'plot3.pdf')
```

```
% 2nd Part) Matched Filtering
%% Adding noise - Matched Filtering

clc;
set(gcf,'PaperSize',[18 12]);
set(gcf,'position',[0 0 1300 900]);

% sigs_MF = cell(1, length(pulse_shapes));
sig_bpsk_noisy = AddNoise(sig_bpsk, 5.62, 1);
sig_bpsk_demod_noisy = PhaseDemod(sig_bpsk_noisy, 1, 0);
[~, berr] = biterr(inpsig, sig_bpsk_demod_noisy);
noisy_analog_sig = cell(1, length(pulse_shapes));
noisy_analog_sig_wo_mf = cell(1, length(pulse_shapes));

for i = 1:length(pulse_shapes)
    if i == 1
        dd = N-1;
    else
        dd = N * NS / 2;
    end

    [sig_pulseshape, pulse, Es] = PulseShape(sig_bpsk, pulse_shapes{i},...
        NS, N, roll_off);
    orig_sig = sig_pulseshape;
    noisy_sig = AddNoise(sig_pulseshape, 5.62, 1);

    orig_mf = conv(orig_sig, pulse);
    noisy_mf = conv(noisy_sig, pulse);

    % Orig signal
    subplot(length(pulse_shapes), 2, (i-1)*2+1);
```

```
    h1 = plot(1:350, orig_mf(1:350), 'DisplayName', 'Befor Conv',...
        'LineWidth', 1.1);
    hold on;
    h2 = plot(1+dd:350+dd, orig_mf(1+dd:350+dd), '--', 'DisplayName', 'After Conv',...
        'LineWidth', 1.6);

    hold on;
    demod_mf = PhaseDemod(orig_mf(1+dd:10:end), 1, 0);
    if i > 1
        demod_mf = demod_mf(6:length(inpsig)+6);
    end
    [~, berr] = biterr(inpsig, demod_mf(1:length(inpsig)));
    title(sprintf('%s w/o noise BER:%.5f', pulse_shapes{i}, berr));

    h3 = stem((1+dd+5:10:355+dd), demod_mf(1:35), 'k', 'DisplayName', 'Rx Bitstream');
    h4 = patch([0 dd dd 0], [-5 -5 5 5], [17 17 17]/255);
    alpha(.2);
    legend([h1 h2 h3]);
    ylim([-1.6 2.5])

    % Noisy signal
    subplot(length(pulse_shapes), 2, (i-1)*2+2);
    h1 = plot(1:350, noisy_sig(1:350), 'DisplayName', 'Befor Conv',...
        'LineWidth', 1.1);
    hold on;
    h2 = plot(1+dd:350+dd, noisy_mf(1+dd:350+dd), '--', 'DisplayName', 'After
Conv',...
        'LineWidth', 1.6);
    hold on;

    % Optimal Sampling
    demod_mf = PhaseDemod(noisy_mf(1+dd:10:end), 1, 0);
    if i > 1
        demod_mf = demod_mf(6:end);
    end
    demod_mf = demod_mf(1:length(inpsig));
    [~, berr] = biterr(inpsig, demod_mf);
    noisy_analog_sig{i} = Digital2Analog(demod_mf, bits, 1, 255);
    analog_snr = snr(Original', Original'-interp(noisy_analog_sig{i}, 65536/fs));

    title(sprintf('%s w/ noise BER:%.5f SNR:%.3f', pulse_shapes{i}, berr,
analog_snr));
    h3 = stem((1+dd+5:10:355+dd), demod_mf(1:35), 'k', 'DisplayName', 'Rx Bitstream');
```

```
    h4 = patch([0 dd dd 0], [-5 -5 5 5], [17 17 17]/255);
    alpha(.2);
    legend([h1 h2 h3]);
    ylim([-1.6 2.5])

    fprintf('%s (opt) w/ noise\t BER:%.5f SNR:%.3f \n', pulse_shapes{i}, berr,
analog_snr);

    % Without matched filtering
    demod_wo_mf = PhaseDemod(noisy_sig(1+dd:10:end), 1, 0);
    demod_wo_mf = demod_wo_mf(1:length(inpsig));
    [~, berr] = biterr(inpsig, demod_wo_mf);
    noisy_analog_sig_wo_mf{i} = Digital2Analog(demod_wo_mf, bits, 1, 255);
    analog_snr = snr(Original', Original'-interp(noisy_analog_sig_wo_mf{i}, 65536/
fs));

    fprintf('%s (opt) w/o MF\t BER:%.5f SNR:%.3f \n', pulse_shapes{i}, berr,
analog_snr);

    % Nonoptimal sampling
    demod_mf = PhaseDemod(noisy_mf(1+dd-1:10:end-1), 1, 0);
    if i > 1
        demod_mf = demod_mf(6:end);
    end
    demod_mf = demod_mf(1:length(inpsig));
    [~, berr] = biterr(inpsig, demod_mf);
    noisy_analog_nonopt = Digital2Analog(demod_mf, bits, 1, 255);
    analog_snr = snr(Original', Original'-interp(noisy_analog_nonopt, 65536/fs));

    fprintf('%s (nonopt) w/ noise\t BER:%.5f SNR:%.3f \n', pulse_shapes{i}, berr,
analog_snr);

end
saveas(gcf, 'plot4.pdf')

%% Listen to received analog signals
sound(noisy_analog_sig{4}, fs);
% sound(noisy_analog_sig_wo_mf{1}, fs);

% 3rd Part) Fading
%% Without Fading
sig_bpsk_noisy = AddNoise(sig_bpsk, 5.62, 1);
b_hat_bpsk_noisy = PhaseDemod(sig_bpsk_noisy, 1);
```

```
[~, berr] = biterr(inpsig, b_hat_bpsk_noisy);
analog_bpsk_noisy = Digital2Analog(b_hat_bpsk_noisy, bits, 1, 255);
analog_snr_bpsk = snr(Original', Original'-interp(analog_bpsk_noisy, 65536/fs));
sound(analog_bpsk_noisy, fs);

%% With fading
clc;
fading_types = {'RAYL', 'RAYL', 'RICE', 'RICE', 'RICE'};
analog_faded = cell(1, length(fading_types));
dopper_shifts = [200 10 200 200 200];
k_factors = [0 0 1 10 100];
for i=1:length(fading_types)
    doppler_rate = dopper_shifts(i)/fs;
    [faded_sig, channel] = FadingChannel(sig_bpsk, fading_types{i},...
        doppler_rate, k_factors(i));
    rx_sig = AddNoise(faded_sig, 5.62, 1);
    z = conj(channel) .* rx_sig;
    b_hat_faded = PhaseDemod(z, 1);
    [~, berr] = biterr(inpsig, b_hat_faded);
    analog_faded{i} = Digital2Analog(b_hat_faded, bits, 1, 255);
    analog_snr_faded = snr(Original', Original'-interp(analog_faded{i}, 65536/fs));

    subplot(length(fading_types), 1, i);
    semilogy(1:fs, sqrt(channel(1:fs).^2));
    ylabel('Semilog Ch. Amp.');
    xlabel('Time');
    xlim([1 fs]);

    if fading_types{i} == 'RAYL'
        fprintf('%s DopplerShift=%d    \tBER: %.4f, SNR: %.4f\n',...
            'Rayleigh', dopper_shifts(i), berr, analog_snr_faded);
        title(sprintf('%s DopplerShift=%d',...
            'Rayleigh', dopper_shifts(i)));
    else
        fprintf('%s DopplerShift=%d, K=%d\tBER: %.4f, SNR: %.4f\n',...
            'Ricean', dopper_shifts(i), k_factors(i), berr, analog_snr_faded);
        title(sprintf('%s DopplerShift=%d, K=%d',...
            'Ricean', dopper_shifts(i), k_factors(i)));
    end
end

%% Listen to received signals
sound(analog_faded{1}, fs);
```