EN2040 Random Signals and Processes

Simulation Assignment 180631J

Rectangular pulses of ±A carry binary equiprobable data over a communication channel. Binary data D $\in$ {0, 1} is mapped to the amplitude of the rectangular pulses S as follows:

$$S = \begin{cases} +A & if \;\; D = 1 \\ -A & if \;\; D = 0 \end{cases}$$

The channel is corrupted by additive white Gaussian noise (AWGN) of zero mean and variance σ^2 . At the receiver,the received signal is sampled and compared with a threshold $\tau = 0$. A received signal sample is given by R = S + N, where N represents the random effect of noise. Decoding is done by considering the amplitude of R. To this end, the decision is taken as follows:

$$Y = \begin{cases} +A & if \;\; R > \tau \\ -A & if \;\; R \leq \tau \end{cases}$$

1. Generate a binary sequence of length L = 1000, considering D $\in$ {0, 1} and Pr (D = 0) = Pr (D = 1) = 1/2. Use the binary sequence to generate a stream of rectangular pulses of amplitude S, where S $\in$ {−A, +A} with A = 1.

```
L = 1000;   % length of the binary sequence
A = 1;      % amplitude of the pulse
bin_seq = randi([0 1], 1, L);
pulse_seq = zeros(1,L); % transmitted signal
for index = 1:L
    if bin_seq(index) == 1
        pulse_seq(index) = +A;
    else
        pulse_seq(index) = -A;
    end
end
```

2. Generate an AWGN sequence, also of length L = 1000, considering σ^2= 1.

```
mu = 0;                     % mean
variance = 1;               % variance
sigma = sqrt(variance);     % standard deviation
awgn_seq = normrnd(mu,sigma,[1, L]);
```

3. Plot the sequence of R and observe the impact of the variance of noise on R by varying σ^2 .

```
R = pulse_seq + awgn_seq;   % received signal
stairs(R);
ylabel("$ Amplitude$",'Interpreter','latex');
xlabel("$ Sample$",'Interpreter','latex');
grid on;
figure;
```

```
for sigma2 = 1:3
    subplot(3,1,sigma2);
    awgn_seq2 = normrnd(mu,sigma2,[1, L]);
    R2 = pulse_seq + awgn_seq2;
    stairs(R2);
    ylabel("$Variance = " +num2str(sigma2^2)+"$",'Interpreter','latex');
end
```

4. Sketch and compare the sequence of Y with the transmitted signal.

```
xspan = 50;             % to visualize a limited number of samples
figure;
% sketching the transmitted signal
stairs(pulse_seq, 'DisplayName', 'Transmitted Signal','MarkerSize',10);
hold on;
threshold = 0;          % thresholding the received signal
Y = zeros(1,L);
for index = 1:L
    if R(index) > threshold
        Y(index) = +A;
    else
        Y(index) = -A;
    end
end
% sketching the thresholded signal

stairs(Y,'DisplayName','Thresholded Signal','MarkerSize',10);
ylabel("$ Amplitude$",'Interpreter','latex');
xlabel("$ Sample$",'Interpreter','latex');
grid on; legend('show');
ylim([-2 2]); xlim([(L/2 -xspan) (L/2 +xspan)]);
hold off;
```

Comparison: Some of the transmitted pulses have been identified by the receiver as they are. while some of them have been identified incorrectly due to the AWGN added in the channel.

5. Repeat the above steps for a sequence of length L = 100,000. Write a code to generate and plot the histogram of the received sequence taking the no of bins as 10. Compare your result with the one generated from the built-in function hist() of MATLAB.

(a) Change the no of bins from 10 to 100 and observe the impact.

(b) By selecting a suitable value for the no of bins, sketch the conditional PDFs $f_{R|S}(r|S = A)\ \ and\ \ f_{R|S}(r|S = -A)$ with the use of the normalized histograms. Change the value of A and observe the impact.

(c) Find E [R|S = A] ;E [R|S = �A] and E [R].

(d) Similarly, sketch the PDF fR (r).

Repeating the previous steps for a sequence of length L = 100,000

```matlab
L = 100000;   % length of the binary sequence
A = 1;      % amplitude of the pulse
bin_seq = randi([0 1], 1, L);
pulse_seq = zeros(1,L); % transmitted signal
for index = 1:L
    if bin_seq(index) == 1
        pulse_seq(index) = +A;
    else
        pulse_seq(index) = -A;
    end
end

% generating AWGN noise
mu = 0;                          % mean
variance = 1;                    % variance
sigma = sqrt(variance);          % standard deviation
awgn_seq = normrnd(mu,sigma,[1, L]);

R = pulse_seq + awgn_seq;        % received signal

xspan = 50;                      % to visualize a limited number of samples
figure;
% sketching the transmitted signal
stairs(pulse_seq, 'DisplayName', 'Transmitted Signal','MarkerSize',10 );
hold on;
threshold = 0;                   % thresholding the received signal
Y = zeros(1,L);
for index = 1:L
    if R(index) > threshold
        Y(index) = +A;
    else
        Y(index) = -A;
    end
end
% sketching the thresholded signal
stairs(Y,'DisplayName','Thresholded Signal','MarkerSize',10);
ylabel("$ Amplitude$",'Interpreter','latex');
xlabel("$ Sample$",'Interpreter','latex');
grid on; legend('show');
ylim([-2 2]); xlim([(L/2 -xspan) (L/2 +xspan)]);
hold off;
```

Code to generate and plot the histogram of the received sequence

```matlab
% taking the no of bins as 10. Functions is at the end of the script.
figure; myHistogram(R,10,0);
ylabel("$ Frequency$",'Interpreter','latex');
xlabel("$ Amplitude$",'Interpreter','latex');
% histogram generated from the built-in function hist() of MATLAB.
figure; hist(R,10);
ylabel("$ Frequency$",'Interpreter','latex');
xlabel("$ Amplitude$",'Interpreter','latex');
```

Comparison: Histograms generated using the custom code and the built-in function are almost the same. The reason for the difference in frequencies can be identified as the range difference of bins considered in the two algorithms.

Impact when the number of bins is changed from 10 to 100

```
% histogram generated from the custom function.
figure; myHistogram(R, 100,0);
ylabel("$ Frequency$",'Interpreter','latex');
xlabel("$ Amplitude$",'Interpreter','latex');
% histogram generated from the built-in function hist() of MATLAB.
figure; hist(R,100);
ylabel("$ Frequency$",'Interpreter','latex');
xlabel("$ Amplitude$",'Interpreter','latex');
% By selecting a suitable value for the no of bins,
% sketch the conditional PDFs  with the use of the normalized histograms
% Change the value of A and observe the impact.
bins = 100;
Normalized = 1; % to get the probability as the dependant axis
```

Marginal PDF f_{R}(r)

```
[f_R, r, bin_width] = myHistogram(R, bins,Normalized);
ylabel("$f_{R}(r)$",'Interpreter','latex');
xlabel("$ Amplitude$",'Interpreter','latex');
hold on; plot(r, f_R);hold off;
% area of the graph should be 1 to be a PDF
disp("Area of the bar graph = " + num2str(sum(f_R.*bin_width)))
disp("Expected value E[R] = " + num2str(Expected(r, f_R, bin_width)))
```

Conditional PDF f_{R|S}(r|S = A); A = 1

```
valArray1 = getConditionedVlaues(R, pulse_seq, A);
[f_RS1, rs1, bin_width] = myHistogram(valArray1, bins,Normalized);
ylabel("$f_{R|S}(r|S=" +num2str(A)+")$",'Interpreter','latex');
xlabel("$ Amplitude$",'Interpreter','latex');
hold on; plot(rs1, f_RS1);hold off;
disp("Area of the bar graph = " + num2str(sum(f_RS1.*bin_width)))
disp("Expected value E[R|S=A] = " + num2str(Expected(rs1, f_RS1, bin_width)))
```

Conditional PDF f_{R|S}(r|S = -A); A = 1

```
valArray2 = getConditionedVlaues(R, pulse_seq, -A);
[f_RS2, rs2, bin_width] = myHistogram(valArray2, bins,Normalized);
ylabel("$f_{R|S}(r|S=" +num2str(-A)+")$",'Interpreter','latex');
xlabel("$ Amplitude$",'Interpreter','latex');
hold on; plot(rs2, f_RS2);hold off;
disp("Area of the bar graph = " + num2str(sum(f_RS2.*bin_width)))
disp("Expected value E[R|S=-A] = " + num2str(Expected(rs2, f_RS2, bin_width)))
```

Change the value of A to observe the impact

```
position = 1; % subplots position
for A = [1,3,5]
    for i = [1, -1]
        subplot(3,2,position);
        ylabel("$f_{R|S}(r|S=" +num2str(i*A)+")$",'Interpreter','latex')
        R = pulse_seq*A + awgn_seq;   % received signal
        valArray1 = getConditionedVlaues(R, pulse_seq*A, i*A);
        [counts, samples, bin_width] = myHistogram(valArray1, bins,Normalized);
        hold on;
        ylabel("$f_{R|S}(r|S=" +num2str(i*A)+")$",'Interpreter','latex')
        plot(samples, counts);hold off;
        position = position +1;
    end
end
```

6. Now, consider that there is interference I from other transmitters in addition to noise. Thus, the received sample can then be written as R = S + N + I: Assuming that I is also Gaussian distributed with zero mean and variance = 1, repeat step 5 � (b) to (d) and discuss the impact of the addition of interference.

```
% generating AWGN noise and interference
mu = 0;                     % mean
variance = 1;               % variance
sigma = sqrt(variance);     % standard deviation
awgn_seq = normrnd(mu,sigma,[1, L]);
interference = normrnd(mu,sigma,[1, L]);

R = pulse_seq + awgn_seq + interference;   % received signal
```

Marginal PDF f_{R}(r) (under Interference)

```
figure;
[f_R, r, bin_width] = myHistogram(R, bins,Normalized);
ylabel("$f_{R}(r)$",'Interpreter','latex');
xlabel("$ Amplitude$",'Interpreter','latex');
hold on; plot(r, f_R);hold off;
% area of the graph should be 1 to be a PDF
disp("Area of the bar graph = " + num2str(sum(f_R.*bin_width)))
disp("Expected value E[R] = " + num2str(Expected(r, f_R, bin_width)))
```

Conditional PDF f_{R|S}(r|S = A); A = 1 (under Interference)

```
A = 1; figure;
valArray1 = getConditionedVlaues(R, pulse_seq, A);
[f_RS1, rs1, bin_width] = myHistogram(valArray1, bins,Normalized);
ylabel("$f_{R|S}(r|S=" +num2str(A)+")$",'Interpreter','latex');
xlabel("$ Amplitude$",'Interpreter','latex');
hold on; plot(rs1, f_RS1);hold off;
disp("Area of the bar graph = " + num2str(sum(f_RS1.*bin_width)))
disp("Expected value E[R|S=A] = " + num2str(Expected(rs1, f_RS1, bin_width)))
```

Conditional PDF f_{R|S}(r|S = -A); A = 1 (under Interference)

```
figure;
valArray2 = getConditionedVlaues(R, pulse_seq, -A);
[f_RS2, rs2, bin_width] = myHistogram(valArray2, bins,Normalized);
ylabel("$f_{R|S}(r|S=" +num2str(-A)+")$",'Interpreter','latex');
xlabel("$ Amplitude$",'Interpreter','latex');
hold on; plot(rs2, f_RS2);hold off;
disp("Area of the bar graph = " + num2str(sum(f_RS2.*bin_width)))
disp("Expected value E[R|S=-A] = " + num2str(Expected(rs2, f_RS2, bin_width)))
```

Change the value of A to observe the impact (under Interference)

```
figure;
position = 1; % subplots position
for A = [1,3,5]
    for i = [1, -1]
        subplot(3,2,position);
        ylabel("$f_{R|S}(r|S=" +num2str(i*A)+")$",'Interpreter','latex')
        R = pulse_seq*A + awgn_seq + interference;   % received signal
        valArray1 = getConditionedVlaues(R, pulse_seq*A, i*A);
        [counts, samples, bin_width] = myHistogram(valArray1, bins,Normalized);
        hold on;
        ylabel("$f_{R|S}(r|S=" + num2str(i*A) + ")$",'Interpreter','latex')
        plot(samples, counts);hold off;
        position = position +1;
    end
end
```

7. Finally, consider that the received signal is amplified by a factor of α such that $R = \alpha.S + N$ Repeat step 5 � (b) to (d) and discuss the impact of scaling.

```
A = 1;
bin_seq = randi([0 1], 1, L);
pulse_seq = zeros(1,L); % transmitted signal
for index = 1:L
    if bin_seq(index) == 1
        pulse_seq(index) = +A;
    else
        pulse_seq(index) = -A;
    end
end

figure;
position = 1; % subplots position
for alpha = [1,3,5,7]
    subplot(2,2,position);
    R = alpha*pulse_seq + awgn_seq;
    [f_R, r, bin_width] = myHistogram(R, bins,Normalized);
    hold on;
    ylabel("$f_{R}(r) ~ when ~\alpha = "+num2str(alpha)+"$",'Interpreter','latex');
    plot(r, f_R);hold off;
    position = position +1;
end
```

User defined functions

1. **Function to generate the Histogram**

```
function [counts, samples, bin_width] = myHistogram(valueArray,numberOfBins, Normalized)
    bins = numberOfBins;
    minR = min(valueArray);
    maxR = max(valueArray);
    bin_width = (maxR - minR)/(bins -1);        % width of a bin
    bin_lower_bound = minR - bin_width/2;       % lower limit of a bin
    bin_upper_bound = minR + bin_width/2;       % upper limit of a bin

    % array to store the number of values in a given bin
    counts = zeros(1,bins);
    samples = minR: bin_width: maxR; % middle of every bar

    for bin = 1:bins
        counts(bin) = 0;
        for sample = valueArray
            if bin_lower_bound <= sample && sample < bin_upper_bound
                counts(bin) = counts(bin) +1;
            end
        end
        bin_lower_bound = bin_upper_bound;
        bin_upper_bound = bin_upper_bound + bin_width;
    end

    counts = counts/bin_width; % to get the height of the bar to plot

    % normalization
    if Normalized
        counts = counts/length(valueArray);
    end
    bar(samples, counts, 'BarWidth', 1) % plotting the histogram
end
```

2. Function to get the PDFs

```
function [valueArray] = getConditionedVlaues(receivedSignal, transmittedSignal, condition)
    valueArray = zeros(1, sum(transmittedSignal == condition));
    index = 1;
    for i = 1:length(transmittedSignal)
        if transmittedSignal(i) == condition
            valueArray(index) = receivedSignal(i);
            index = index +1;
        end
    end
end
```

3. Function to calculate the expected values

```
function [value] = Expected(x, px, bin_width)
    value = sum((x.*px)*bin_width)
end
```