GEBZE TECHNICAL UNIVERSITY

ELECTRONICS ENGINEERING

Spring 2024 - ELEC 365

Fundamentals of Digital Communications

PROJECT

NAME - SURNAME: ŞULE NUR DEMİRDAŞ

NUMBER: 210102002053

Theoretical Background:

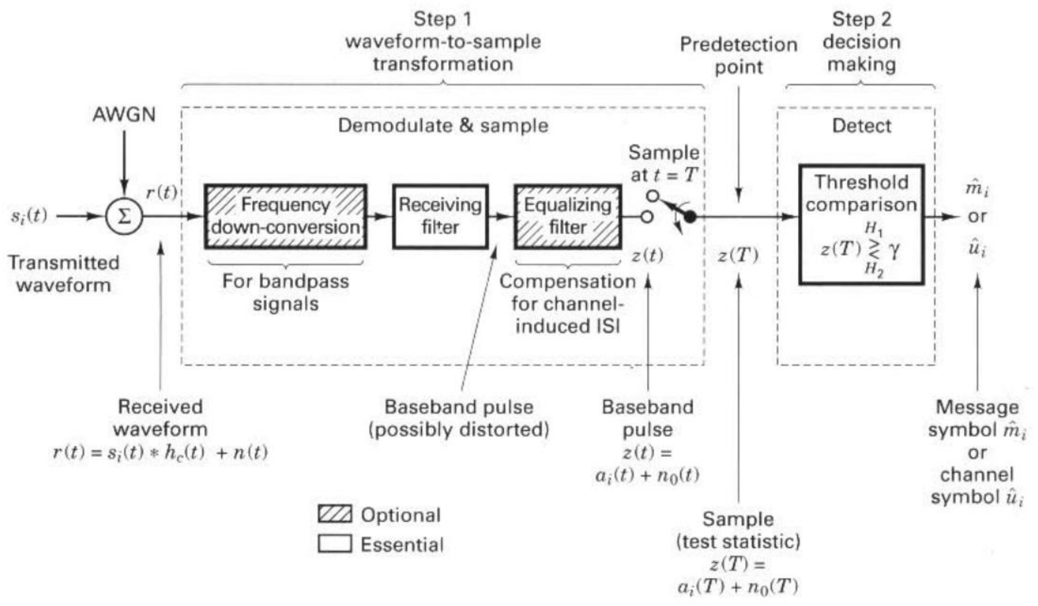


Figure 1: Two basics steps in demodulation/detection of digital signals.

Block diagram shown in the figure 1 is the demodulation of digital signals.

The received signal $r(t)$, which may be corrupted by noise during transmission (in this case it's AWGN), is passed through a receiving filter.

Following the filtering stage, the processed signal is directly sampled at the symbol rate. This sampling process converts the continuous-time signal into a discrete-time representation, capturing individual data symbols at specific points in time.

The heart of the demodulation process lies in the decision-making circuit. This circuit compares the amplitude of each sampled symbol with a predetermined threshold value(γ_0). Based on this comparison, the circuit makes a decision about the transmitted data bit. If the sampled amplitude is significantly higher than the threshold, it's interpreted as a "1," while a value significantly lower than the threshold is interpreted as a "0."

Through this sampling and decision-making process, the simulation effectively recovers the original digital data from the received baseband signal. By analyzing the recovered data and comparing it to the transmitted data, we can assess the system's performance and identify potential sources of errors introduced during transmission.

Theoretical Calculations for Simulation:

s_1 and s_2 given as;

$$s_1(t) = \begin{cases} A\sin\left(\frac{2\pi t}{T}\right), & 0 \le t \le \frac{T}{2} \\ 0, & else \end{cases}$$

$$s_2(t) = -s_1\left(t - \frac{T}{2}\right)$$

If we simplify s_2:

$$s_2(t) = \begin{cases} -A\sin\left(\frac{2\pi\left(t - \frac{T}{2}\right)}{T}\right), & 0 \le t - \frac{T}{2} \le \frac{T}{2} \\ 0, & else \end{cases}$$

$$s_2(t) = \begin{cases} -A\sin\left(\frac{2\pi t}{T} - \frac{2\pi T}{2T}\right), & \frac{T}{2} \le t \le T \\ 0, & else \end{cases}$$

$$s_2(t) = \begin{cases} -A\sin\left(\frac{2\pi t}{T} - \pi\right), & \frac{T}{2} \le t \le T \\ 0, & else \end{cases}$$

We know that,

$$\sin(x - \pi) = -\sin(x)$$

So,

$$s_2(t) = \begin{cases} A\sin\left(\frac{2\pi t}{T}\right), & \frac{T}{2} \le t \le T \\ 0, & else \end{cases}$$

Let’s calculate energies of both signals:

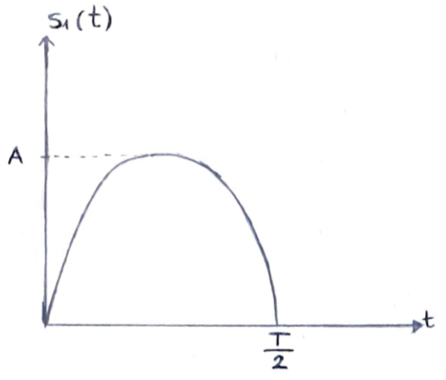


Figure 2: Graph of s1

$$E_{s_1} = \int_0^T s_1(t)^2 dt = \int_0^{\frac{T}{2}} (A\sin\left(\frac{2\pi t}{T}\right))^2\, dt = \frac{A^2 T}{4}$$

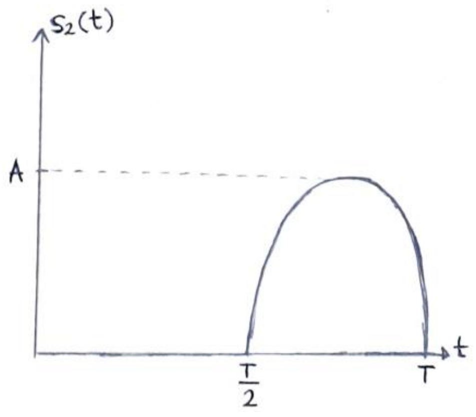
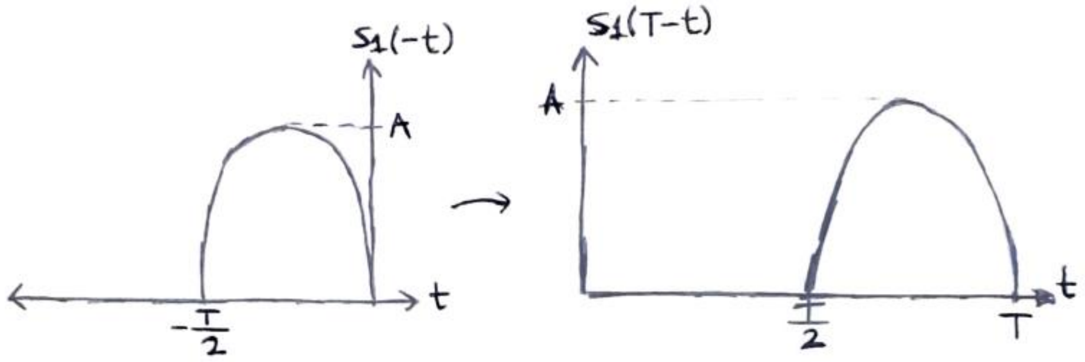


Figure 3: Graph of s2

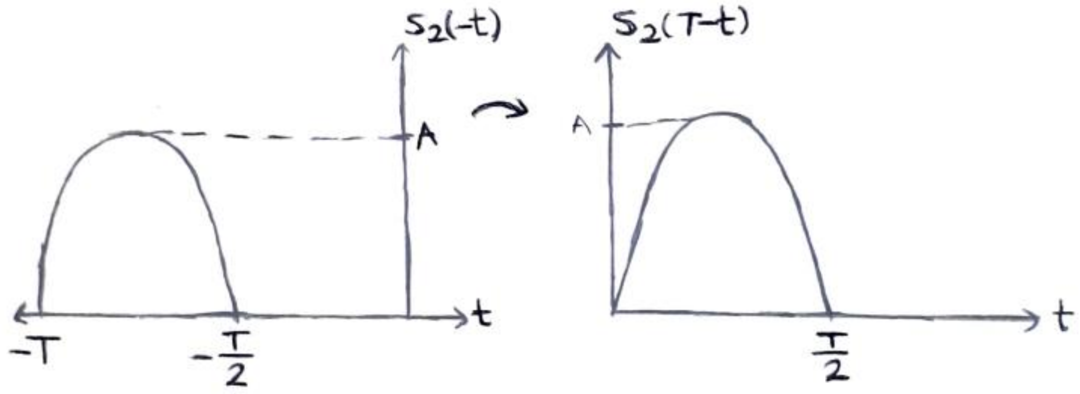
$$E_{s_2} = \int_0^T s_2(t)^2 dt = \int_{\frac{T}{2}}^{T} (Asin\left(\frac{2\pi t}{T}\right))^2\, dt = \frac{A^2 T}{4}$$

$$h(t) = s_1(T-t) - s_2(T-t)$$

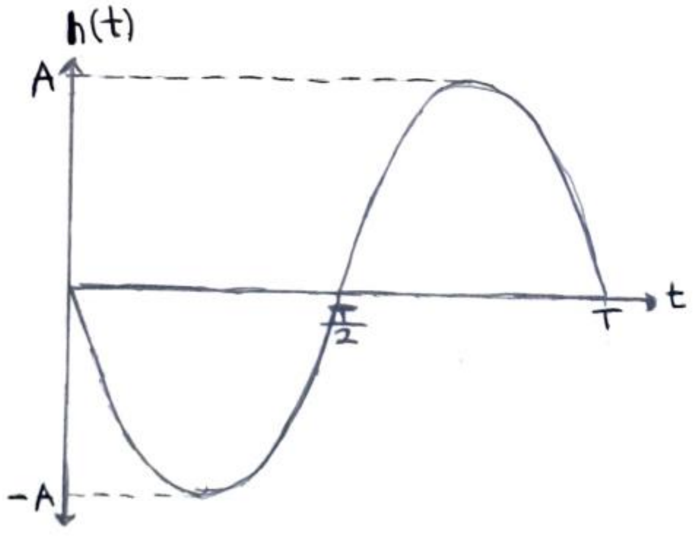
If we obtain $s_1(T-t)$ using graphical representation,



If we obtain $s_2(T-t)$ using graphical representation,



Then we subtract $s_1(T-t)$ and $s_2(T-t)$ from each other.

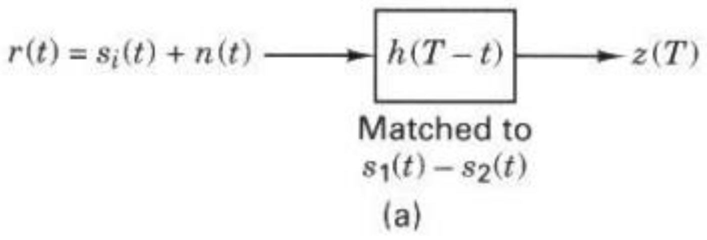


We can obtain $h(t)$ from graphical representation.

$$h(t) = \begin{cases} -Asin\left(\frac{2\pi t}{T}\right), & 0 \leq t \leq T \\ 0, & else \end{cases}$$

And energy of $h(t)$ equals to,

$$E_h = \int_0^T h(t)^2 dt = \int_0^T (-Asin\left(\frac{2\pi t}{T}\right))^2 \, dt = \frac{A^2 T}{2}$$



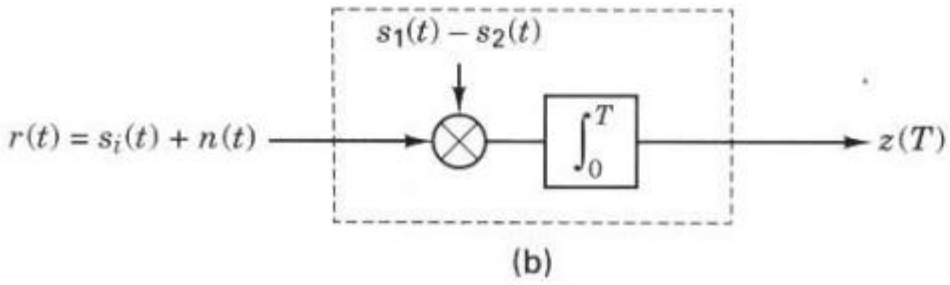


Figure 4:Equaivalence of matched filter and correlator (a) matched filter (b) correlator

To obtain $a_1$and a_2 we can you use correlator. We know that it's equivalent to the match filter shown in the figure 4.

$$a_i = \int_0^T s_i(t)(s_1(t) - s_2(t))dt$$

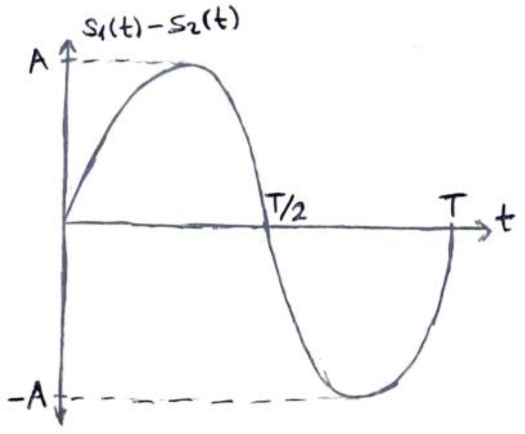


Figure 5: Graph of s1(t)-s2(t)

$$a_1 = \int_0^T s_i(t)(s_1(t) - s_2(t))dt$$

From graphs shown in the figure 5, figure 2 and figure 3 we can easily obtain a_1 and a_2.

$$a_1 = \int_0^T s_1(t)(s_1(t) - s_2(t))dt = \int_0^{\frac{T}{2}} (Asin\left(\frac{2\pi t}{T}\right))^2 dt = \frac{A^2T}{4}$$

$$a_2 = \int_0^T s_1(t)(s_1(t) - s_2(t))dt = \int_{\frac{T}{2}}^{T} -(Asin\left(\frac{2\pi t}{T}\right))^2 dt = -\frac{A^2T}{4}$$

$$\gamma_0 = \frac{a_1 + a_2}{2} = 0$$

$$\sigma_0{}^2 = \frac{N_0 E_h}{2} = \frac{A^2T}{2} N_0$$

a) For $P(1) = P(0) = \frac{1}{2}$,

$$E_b = E_{s_1}P(s_1) + E_{s_2}P(s_2) = \frac{A^2T}{4} * \frac{1}{2} + \frac{A^2T}{4} * \frac{1}{2} = \frac{A^2T}{4}$$

To obtain theoretical P_b we need ρ. From graphical representation of $s_1(t)$ and $s_2(t)$ shown in the figure 2 and figure 3 we can easily calculate ρ.

$$\rho = \frac{1}{E_b} \int_0^T s_1(t)s_2(t)dt = 0$$

$$P_b = Q\left(\sqrt{\frac{E_b(1-\rho)}{N_0}}\right) = Q\left(\sqrt{\frac{E_b}{N_0}}\right)$$

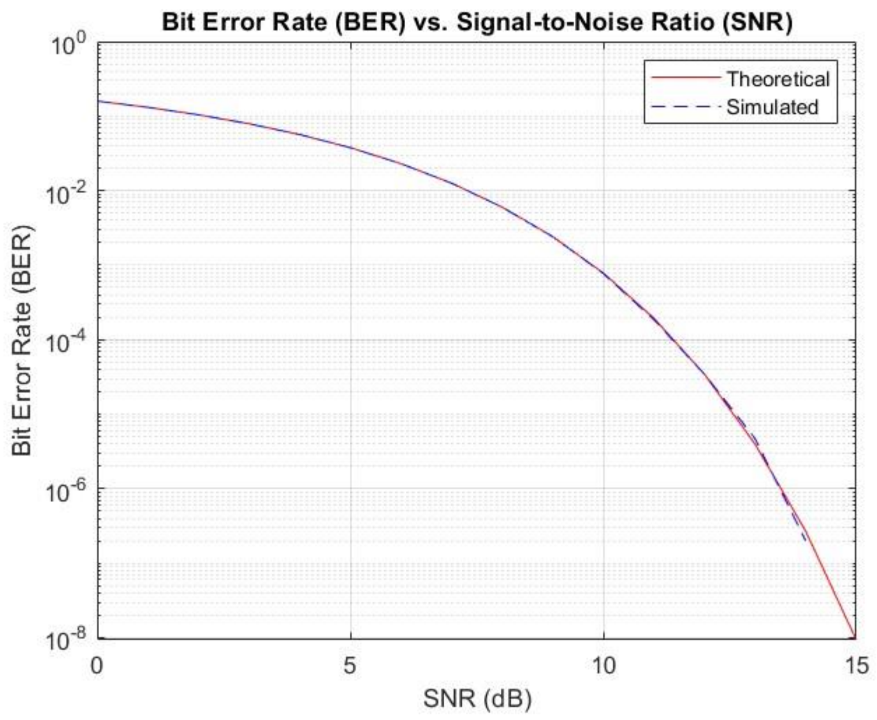


Figure 6: Theoretical and simulation result of case a.

b) For $P(1) = \frac{1}{4}$, $P(0) = \frac{3}{4}$,

$$E_b = E_{s_1}P(s_1) + E_{s_2}P(s_2) = \frac{A^2T}{4} * \frac{1}{4} + \frac{A^2T}{4} * \frac{3}{4} = \frac{A^2T}{4}$$

Theoretical P_b is equal to the one we calculated at (a) because E_b value didn't change.

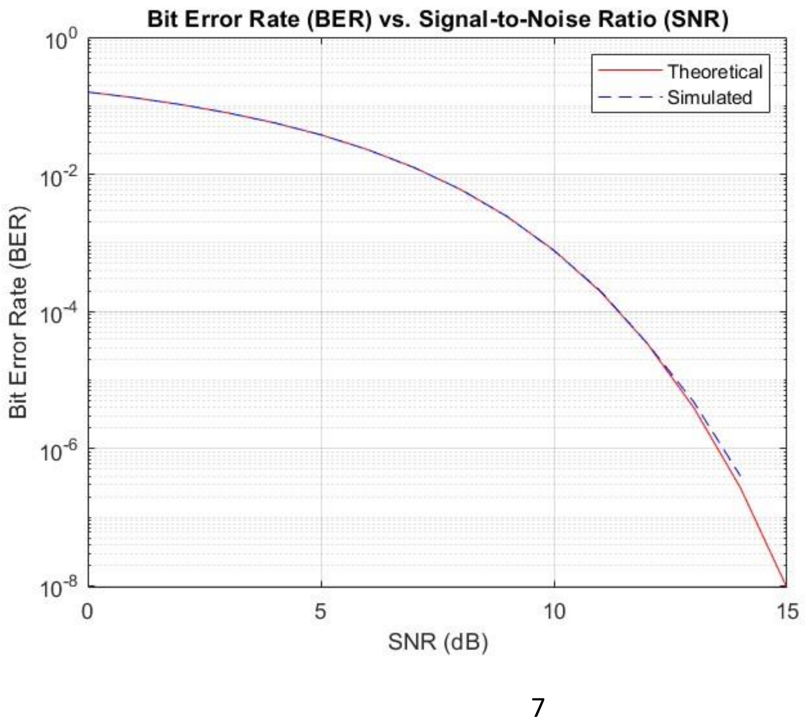


Figure 7: Theoretical and simulation results for case b.

The BER vs SNR curve illustrates how the Bit Error Rate (BER) changes with varying Signal-to-Noise Ratio (SNR) values. We have seen from figure 6 and 7, as SNR increases, meaning the signal strength compared to noise improves, the BER decreases. This indicates that higher SNR values result in better communication performance with fewer errors. Conversely, lower SNR values lead to higher BER and more errors in data transmission. Therefore, achieving higher SNR levels is crucial for reliable communication in noisy environments.

Also we noticed that even slight variations of 1 or 2 dB in higher SNR levels significantly impact the error rate.

Matlab Codes

```
close all;
clear all;

function ber_curve(P0,P1)
    % Parameters
    % we decide Eb to be 1 for both case.
    % Eb = A^2*T/4 to be 1, we choose T = 1 and A = 2.

    T = 1; % Bit duration
    A = 2; % Amplitude of the signals

    SNR_dB = 0:1:15; % Range of SNR values in dB 10log(10) (Eb/N0)

    numBits = 10*10^6; % Number of bits to simulate

    % Calculated Parameters:

    Es1 =  A^2*T/4;
    Es2 =  A^2*T/4;

    a1 =  A^2*T/4;
    a2 =  -A^2*T/4;

    gamma0 = (a1+a2)/2;

    Eh = A^2*T/2;

    BER_theoretical = zeros(size(SNR_dB));
    BER_simulated = zeros(size(SNR_dB));

    Eb = P1*Es1+P0*Es2;

    for i = 1:length(SNR_dB)

        % First we generate random databits for every SNR value
        data = randi([0, 1], 1, numBits);

        % SNR_dB = 10*log(10) (1/N0), (1/N0 because we choose Eb = 1)
        N0 = 1/(10^(SNR_dB(i)/10));

        sigma0Square = N0*Eh/2;

        % Generating AWGN Channel Noise:

        % randn generates normally distributed random numbers with variance of 1
        % but we want normally distributed values with different variance
        % value.
        % To have sdifferent variance value we multiply it with square root of
        % desired varience.
        noise = sqrt(sigma0Square) .* randn(1, numBits);

        % This line of code makes z = a1+n0 when given data bir is 1 and
        % z = a2+n0 when given data is 0.
        z = (a1 + noise) .* data + (a2 + noise) .* (1 - data);

        % This line of code is used for deciding s to be 1 or 0.
        % When z is bigger than gamma0 detected_data_a is 1 when it's 0
        % detected_data_a is 0.
        detected_data_a = (z>gamma0);

        % Calculating Number of Bit Errors
        numErrors = sum(detected_data_a ~= data);

        % Calculate BER (bit error rate)
        % This line of code calculates Pb for every single SNR value.
        BER_simulated(i) = numErrors / numBits;

        % Calculate theoretical BER
        q = qfunc(sqrt(Eb/N0)); % Q-function
        BER_theoretical(i) = q;
    end

    figure;
    semilogy(SNR_dB, BER_theoretical, SNR_dB, BER_simulated,'--');
    grid on;
    xlabel('SNR (dB)');
    ylabel('Bit Error Rate (BER)');
    legend('Theoretical', 'Simulated');
    title('Bit Error Rate (BER) vs. Signal-to-Noise Ratio (SNR)');

end

% For the option (a) (P(1) = P(0) = 1/2):
ber_curve(1/2,1/2)

% For the option (b) (P(1) = 1/4 and P(0) = 3/4):
ber_curve(1/4,3/4)
```