Orkun İbrahim Kök

22003656

EEE431 Assignment #3

1) Frequency Shift Keying

Part a)

Using the given information, 5 random symbols are generated, and corresponding x(t) is plotted. The generated symbol sequence is as follows:

Generated 2-bit symbols:

00 11 01 00 10

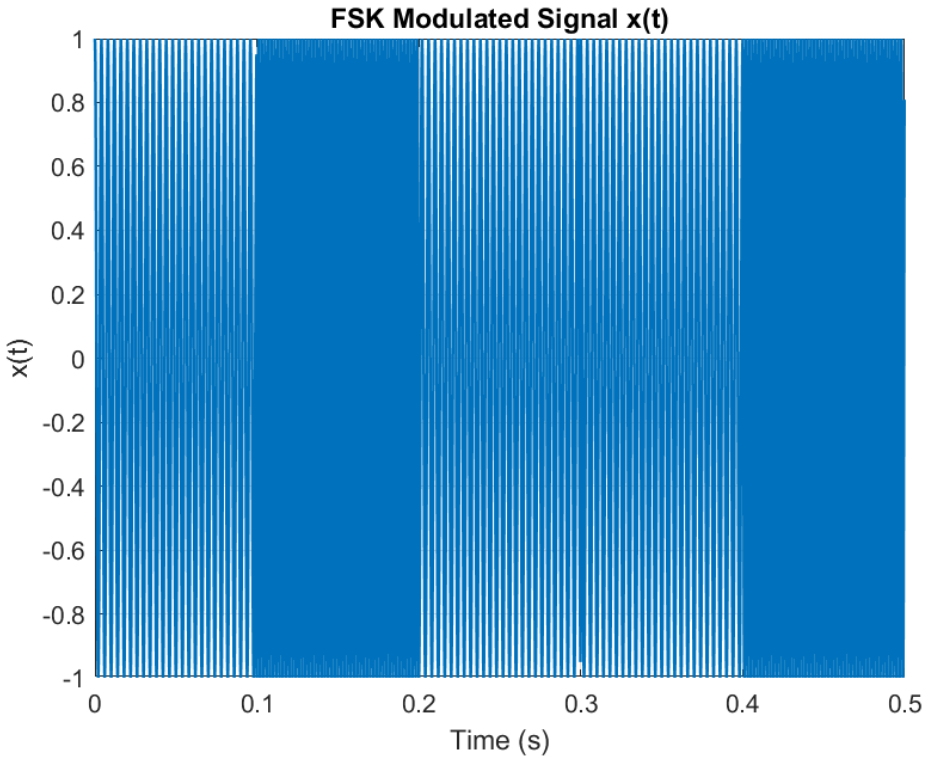


Fig. 1: x(t) for 5 randomly generated symbols

Part b)

$\psi_1(t)$ formula is given below. Because it is FSK, the corresponding basis functions will be formed by cosine and sine, with a constant amplitude.

$$\psi_1(t) = \frac{s_1(t)}{\sqrt{E_p}}$$

From the calculations, the corresonding orthogonal basis functions are as follows:

$$\psi_1(t) = \sqrt{\frac{2}{T}}\cos(2\pi 250t)\, h(t)$$

$$\psi_2(t) = \sqrt{\frac{2}{T}}\cos(2\pi 500t)\, h(t)$$

To be orthogonal, inner product of a basis function with itself has to be equal to 1, and inner product of a basis function with another basis function has to be equal to zero. The process is run through matlab and here is the result:

⟨ψ1, ψ1⟩ = 0.999802

⟨ψ2, ψ2⟩ = 0.999810

⟨ψ1, ψ2⟩ = -0.000194

The basis functions are orthogonal.

The vectors for $s_1(t)$ and $s_2(t)$ will be as follows:

$$s_1 = \begin{bmatrix} \sqrt{\frac{T}{2}} \\ 0 \end{bmatrix} \qquad s_2 = \begin{bmatrix} 0 \\ \sqrt{\frac{T}{2}} \end{bmatrix} \qquad -s_1 = \begin{bmatrix} -\sqrt{\frac{T}{2}} \\ 0 \end{bmatrix} \qquad -s_2 = \begin{bmatrix} 0 \\ -\sqrt{\frac{T}{2}} \end{bmatrix}$$

The dimension of the signal space is as shown 2. The mappings of the signal space is given in below figure.

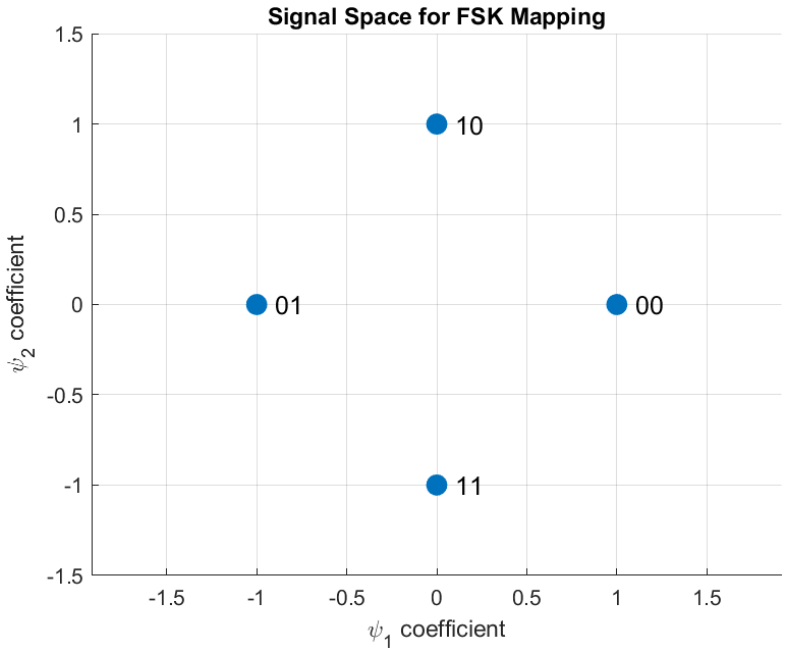


Fig. 2: Signal Space

The plots of the basis functions are given in Fig. 3 and Fig. 4.

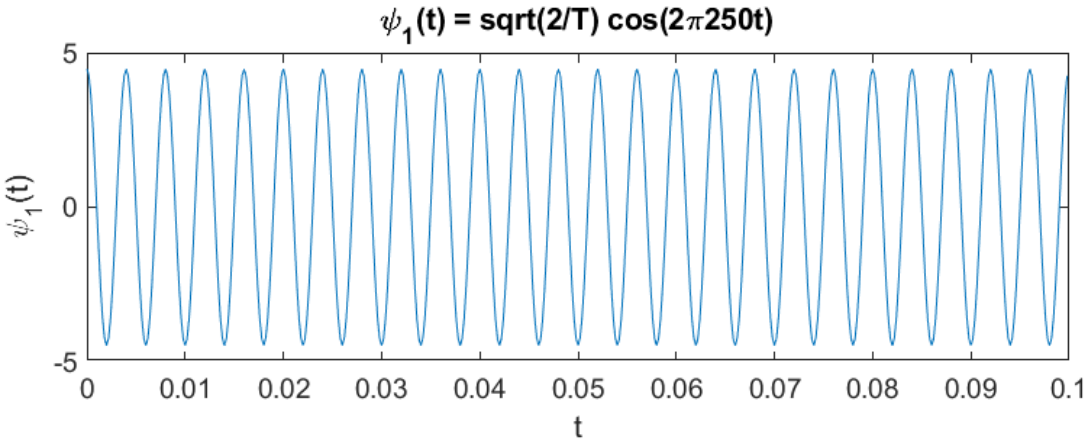


Fig. 3: Plot of first basis function

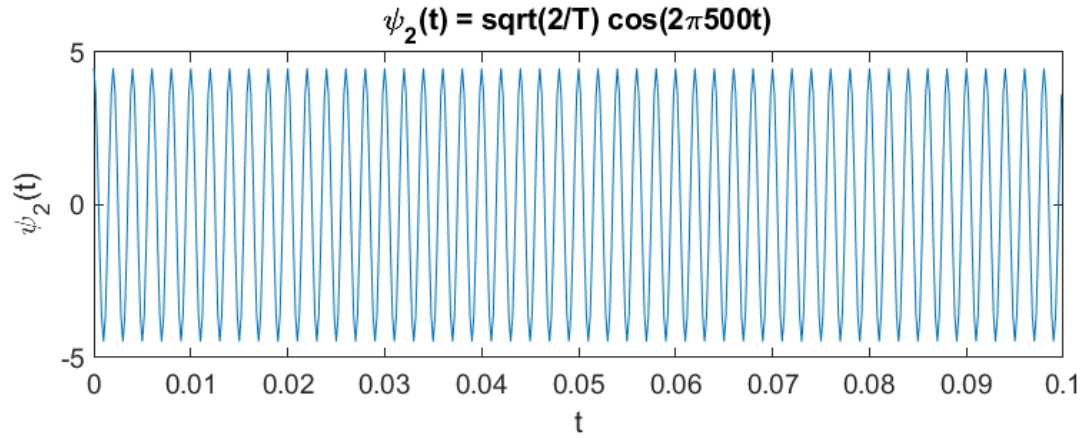


Fig. 4: Plot of second basis function

Part c)

In this part, different variance valued noises were added to the signals. The SNR formula is given below:

$$SNR = \frac{E_s}{N_0}$$

Where E_s has the following formula:

$$E_s = \frac{1}{4}\sum_{k=1}^{4} \|s_i\|^2$$

Because there is 4 signals in total, $s_1(t)$, $s_2(t)$, $-s_1(t)$, $-s_2(t)$; the average is taken by diving to 4.

Regarding the SNR formula, different variance values for noise will indeed give different SNR values, however, the E_s values will remain unchanged.

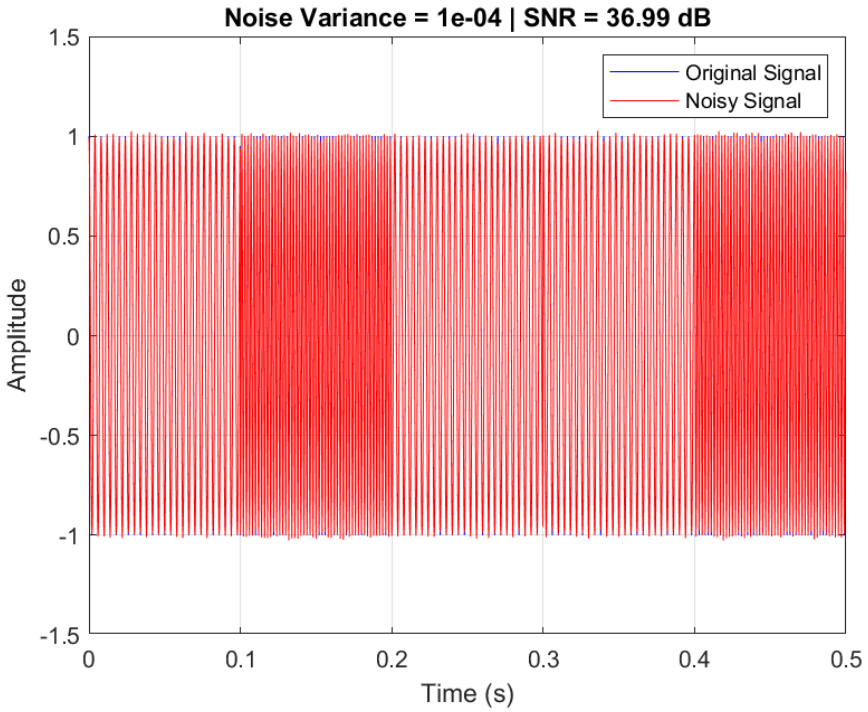


Fig. 5: SNR with var = 10^{-4}

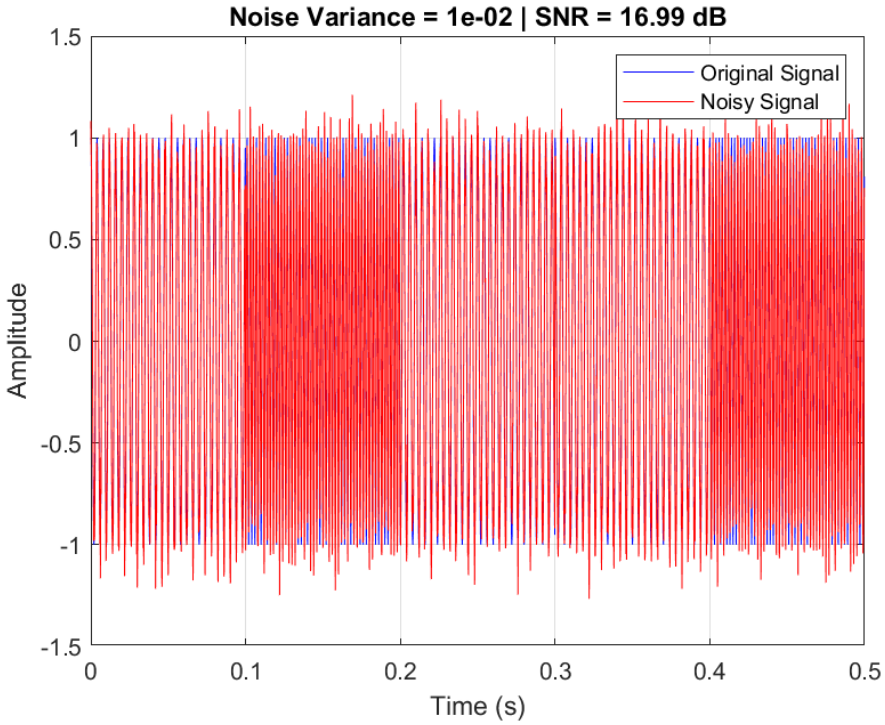


Fig. 6: SNR with var = 10^{-2}

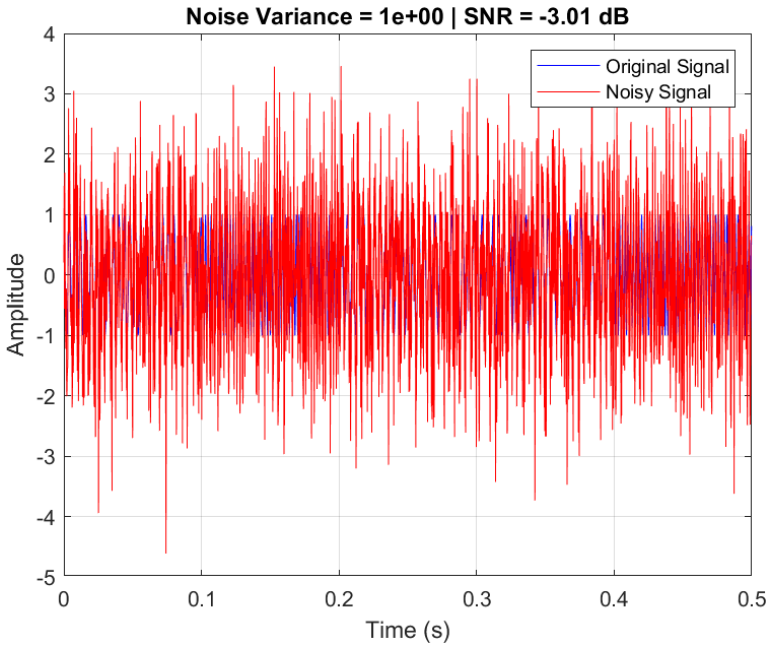


Fig. 7: SNR with var = 1

Part d)

This part can be seen in Appendix b.

Part e)

In this part theoretical and simulation values of the bit error and SNR is analyzed. The analyzed signals form 4-ary FSK. The theoretical and simulated values are close to each other as seen. As SNR increases, probability of bit error decreases as expected.

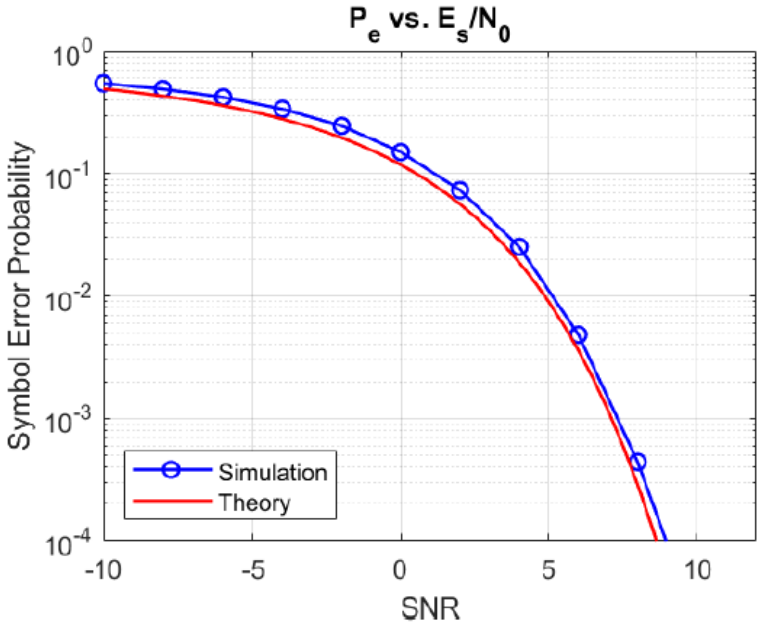


Fig. 8: BER vs. SNR for 4-ary FSK

2) Binary Modulation and MAP vs. MLE

Part a)

Using the given information, again 5 random bits are generated, and corresponding x(t) is plotted. Here is the generated bit sequence:

Random bits:

0 1 0 0 0

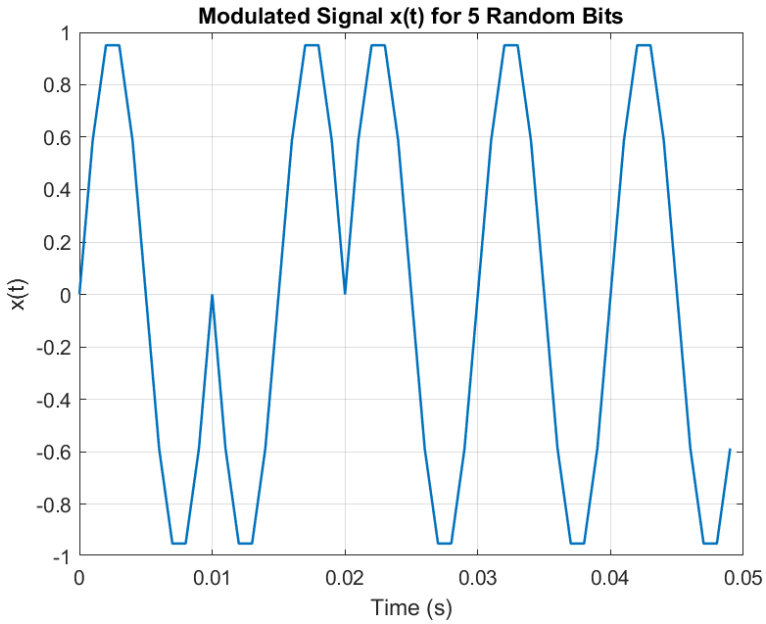


Fig. 9: Randomly generated 5 bits

Part b)

In this part, the basis functions are derived from the following signal, $s_1(t)$.

$$s_1(t) = \sin(2\pi 100t)\, h(t)$$

$$\psi_1(t) = \sqrt{\frac{2}{T}} \sin(2\pi 100t)\, h(t)$$

The constellation points for T=1 turned out to be appear at $-\sqrt{2}$ and $\sqrt{2}$. Fig. 9 shows the plot of the basis function, Fig. 10 shows the constellation points plotted on the signal space.

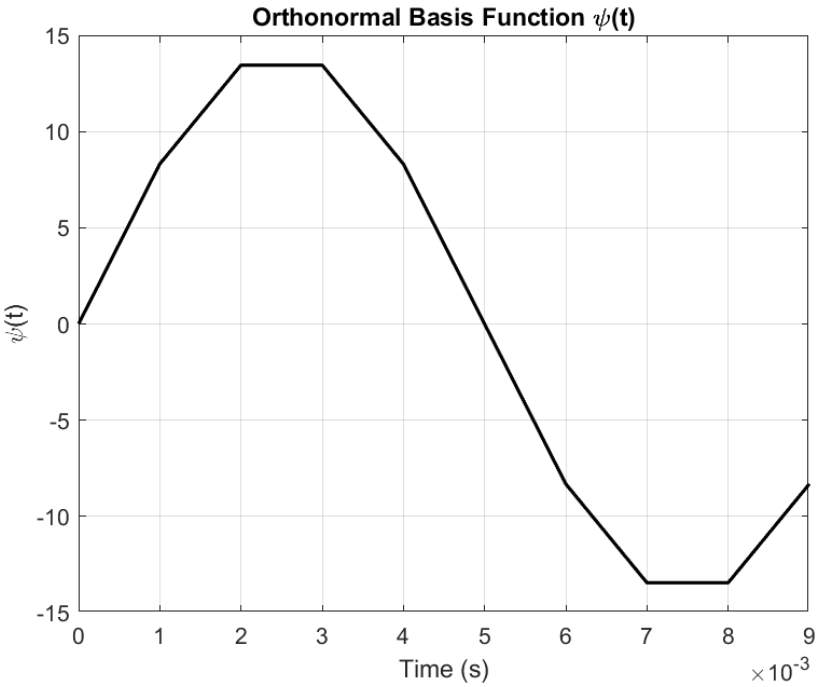


Fig. 10: Plot of basis function

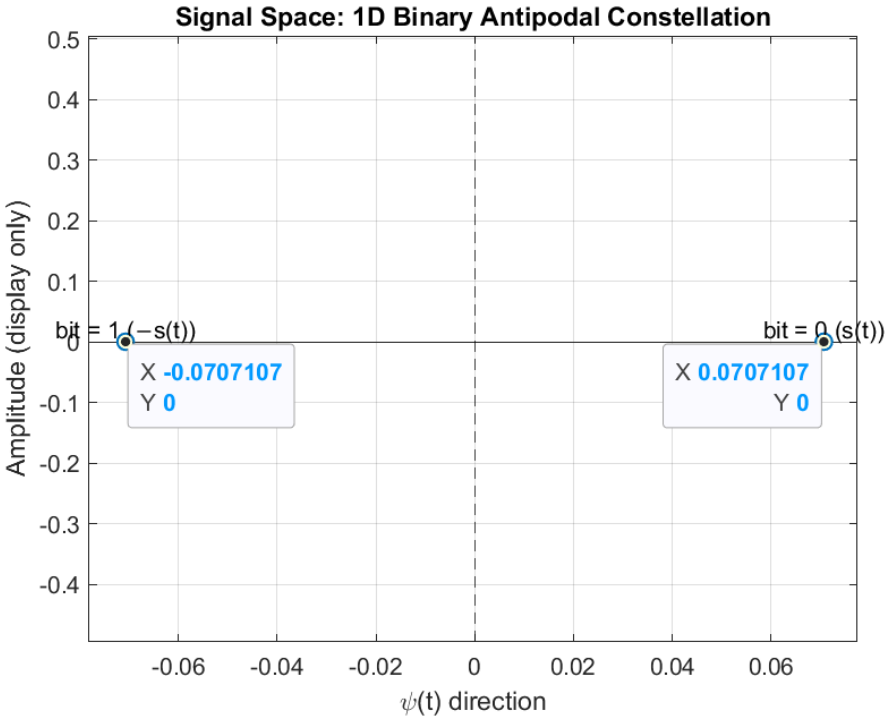


Fig. 11: Constellation points

Part c)

Three different noise signals are added to the signal with different variance values, as done in previous part.

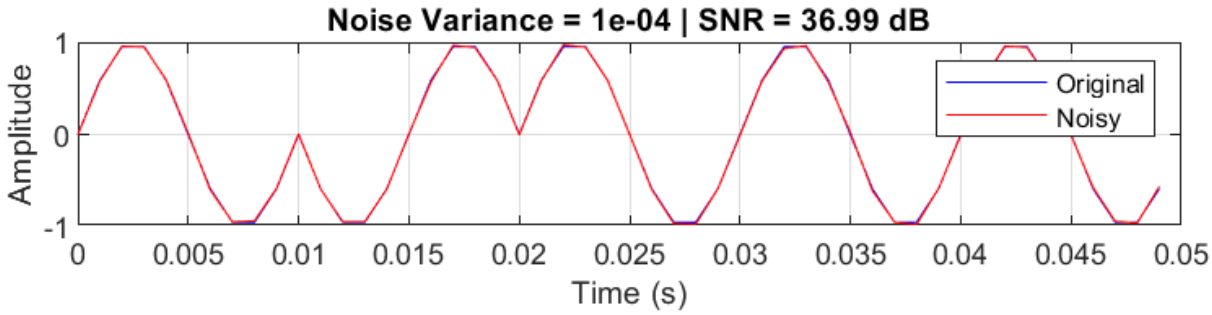


Fig. 12: Noisy plot vs original plot with variance = 10^{-4}

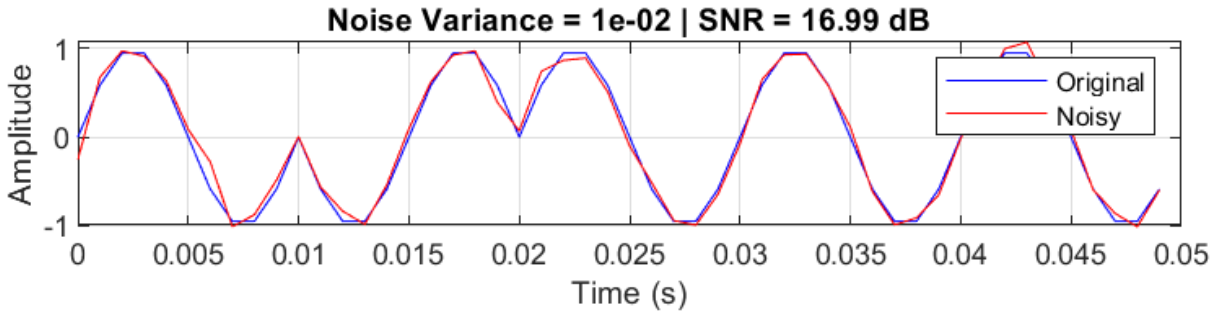


Fig. 13: Noisy plot vs original plot with variance = 10^{-2}

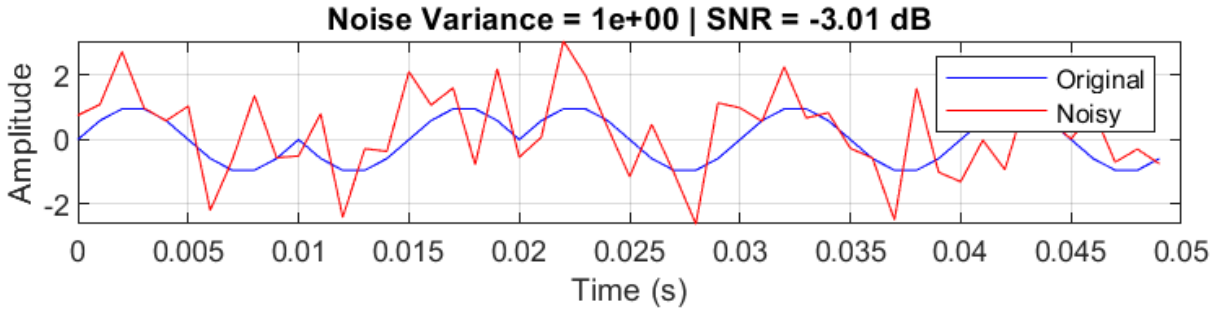


Fig. 14: Noisy plot vs original plot with variance = 1

As seen from the plots, higher the noise, better the signal. The SNR equation is given above, which is signal energy/noise variance. So that, higher the variance, lower the SNR. Therefore, the sampling rate does not affect the SNR values.

Part d)

This part can be seen in Appendix b.

Part e)

In this part, the theoretical value of probability of error is compared to the SNR. The theoretical value is obtained from the below equation. As expected, the curves resemble to each other. Also higher the SNR lower the probability of bit error as expected.

$$P_e = Q\left(\sqrt{\frac{2E_s}{N_0}}\right)$$

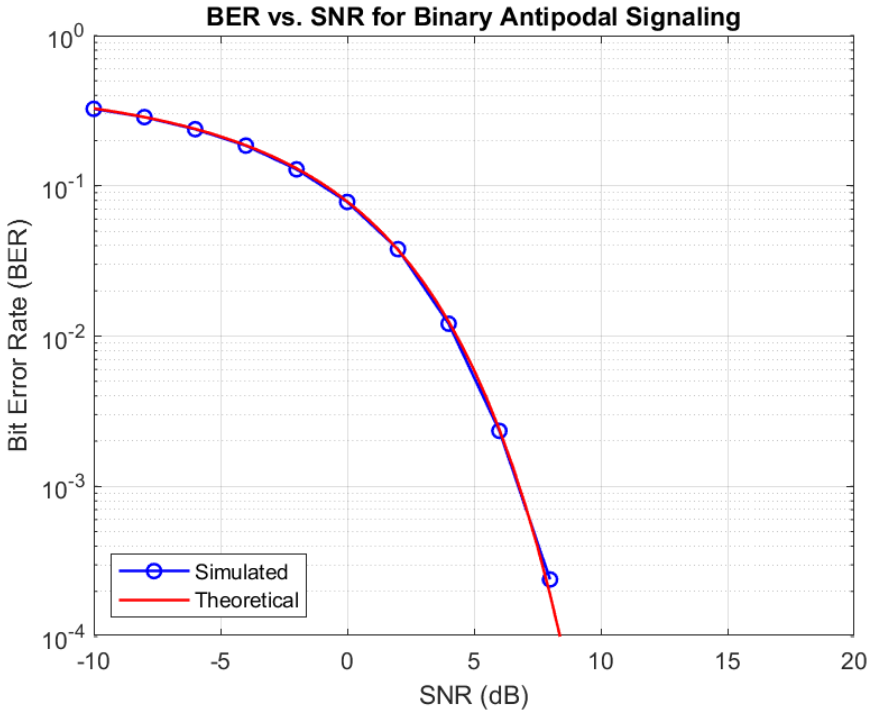


Fig. 15: BER vs SNR plot

Part f)

Because transmitted signals' probabilities are not equal to each other, we have to use MAP rule rather than ML rule. Therefore the receiver found in part d is no longer optimal for this structure. Below is the decision regions regarding the MAP rule.

$$\frac{p(r|0)}{p(r|1)} > \frac{P(1)}{P(0)}$$

$$\exp\left(-\frac{\left(r-\sqrt{E}\right)^2}{2\sigma^2} + \frac{\left(r+\sqrt{E}\right)^2}{2\sigma^2}\right) > \frac{P(1)}{P(0)}$$

$$r > \frac{\sigma^2}{2\sqrt{E}} ln\left(\frac{P(1)}{P(0)}\right)$$

Part g)

As stated, due to the inequality of the probabilities, using MAP rule rather than ML is more optimal, because ML rule only looks at the minimum distance (L2 norm), whereas MAP rule finds the decision boundaries regarding transmission bit error and transmission probabilities. Also, when alpha = 0.5, which is the case where transmission probabilities are equal, both ML rule and MAP rule give the same results as expected.

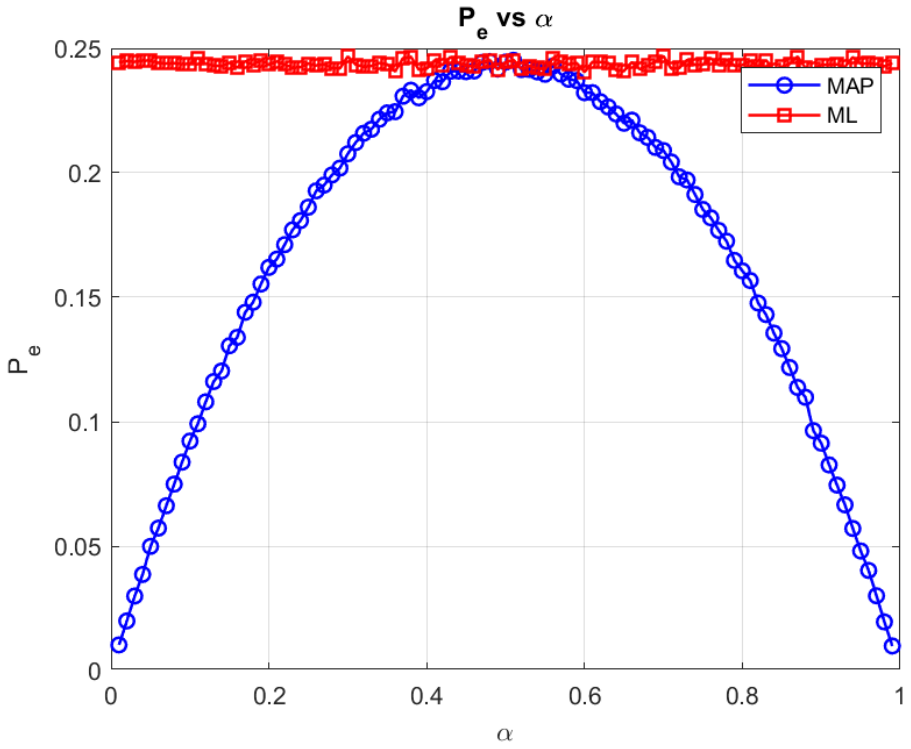


Fig. 16: ML vs. MAP for different bit transmission prababilities

Part h)

When the probabilities are unknown or known to be equal, I will prefer to usse ML rule in order to get vanish any unexpected behavior. However, when the probabilities are known and inequal, I will prefer to use MAP rule in order to get more accurate results, and to minimize the error.

Question 3)

Results are in Appendix b.

Appendix a

```
%% Part a - FKS
rng(55);
% Parameters
T = 0.1;                % symbol duration (s)
Fs = 5000;              % sampling frequency (Hz)
t = 0:1/Fs:T-1/Fs;      % time vector for one symbol
N = 5;                  % number of symbols

s1 = cos(2*pi*250*t);   % corresponds to '00' and '01'
s2 = cos(2*pi*500*t);   % corresponds to '10' and '11'

% Define symbols and mapping
symbols = ['00'; '01'; '10'; '11'];

% Generate 5 random symbols
rand_idx = randi(4, 1, N);
message = symbols(rand_idx, :);


x = [];


for i = 1:N
    symbol = message(i, :);
    switch symbol
        case '00'
            x_segment = s1;
        case '01'
            x_segment = -s1;
        case '10'
            x_segment = s2;
        case '11'
            x_segment = -s2;
    end
    x = [x, x_segment];
end

total_time = 0:1/Fs:T*N - 1/Fs;
figure;
plot(total_time, x, 'LineWidth', 1.2);
xlabel('Time (s)');
ylabel('x(t)');
title('FSK Modulated Signal x(t)');
grid on;

disp('Generated 2-bit symbols:');
joined_message = join(string(message), ' ');
disp(joined_message);
%% Part b orthogonality check

T = 1;              % Duration in seconds
fs = 10000;         % Sampling frequency
t = 0:1/fs:T-1/fs;  % Time vector

% Define h(t): rectangular pulse from 0 to T
h = ones(size(t));
```

```
% Define basis functions
psi1 = sqrt(2/T) * cos(2*pi*250*t) .* h;
psi2 = sqrt(2/T) * cos(2*pi*500*t) .* h;

% Inner products
inner_psi1_psi1 = trapz(t, psi1 .* psi1); % Should be 1
inner_psi2_psi2 = trapz(t, psi2 .* psi2); % Should be 1
inner_psi1_psi2 = trapz(t, psi1 .* psi2); % Should be 0


fprintf('⟨ψ1, ψ1⟩ = %.6f\n', inner_psi1_psi1);
fprintf('⟨ψ2, ψ2⟩ = %.6f\n', inner_psi2_psi2);
fprintf('⟨ψ1, ψ2⟩ = %.6f\n', inner_psi1_psi2);
%% Part b

T = 0.1;                % Symbol duration in seconds
Fs = 5000;              % Sampling frequency (Hz)
t = 0:1/Fs:T-1/Fs;      % Time vector
h = ones(size(t));      % Rectangular window

% Orthonormal basis functions
psi1 = sqrt(2/T) * cos(2*pi*250*t) .* h;
psi2 = sqrt(2/T) * cos(2*pi*500*t) .* h;

% Plot basis functions
figure;
subplot(2,1,1);
plot(t, psi1);
title('\psi_1(t) = sqrt(2/T) cos(2\pi250t)');
xlabel('t'); ylabel('\psi_1(t)');

subplot(2,1,2);
plot(t, psi2);
title('\psi_2(t) = sqrt(2/T) cos(2\pi500t)');
xlabel('t'); ylabel('\psi_2(t)');


% Signal space mapping
symbols = ["00", "01", "10", "11"];
coords = [ 1,  0;    % 00 → +psi1
          -1,  0;    % 01 → -psi1
           0,  1;    % 10 → +psi2
           0, -1];   % 11 → -psi2

% Plot signal space
figure;
scatter(coords(:,1), coords(:,2), 100, 'filled');
text(coords(:,1)+0.1, coords(:,2), symbols, 'FontSize', 12);
xlim([-1.5 1.5]); ylim([-1.5 1.5]);
xlabel('\psi_1 coefficient'); ylabel('\psi_2 coefficient');
title('Signal Space for FSK Mapping');
grid on; axis equal;


%% Part c
% Part (c) - Add Gaussian noise and plot
noise_vars = [1e-4, 1e-2, 1e0];  % Variances
num_cases = length(noise_vars);
signal_power = mean(x.^2);       % Average signal power
```

```
figure;
for i = 1:num_cases
    noise = sqrt(noise_vars(i)) * randn(size(x));  % Generate noise
    x_noisy = x + noise;                           % Add noise
    snr = signal_power / noise_vars(i);            % Linear SNR
    snr_db = 10 * log10(snr);                      % dB scale

    subplot(num_cases, 1, i);
    plot(total_time, x, 'b', 'DisplayName', 'Original Signal'); hold on;
    plot(total_time, x_noisy, 'r', 'DisplayName', 'Noisy Signal');
    title(sprintf('Noise Variance = %.0e | SNR = %.2f dB', noise_vars(i),
snr_db));
    xlabel('Time (s)'); ylabel('Amplitude');
    legend; grid on;
end
%% Part e
clc; clear;


Fs = 5000;                 % Sampling frequency (Hz)
T = 0.1;                   % Bit duration (s)
t = 0:1/Fs:T-1/Fs;         % Time vector
symbol_num = 1e5;          % Number of bits


s = sqrt(2/T) * cos(2*pi*250*t);   % Orthonormal basis function
phi = s / norm(s);                 % (Optional: normalize again)
E = trapz(t, s.^2);                % Energy of one symbol (should be 1)

% SNR values (dB and linear)
SNRdb_sim = -10:2:20;
SNR_sim = 10.^(SNRdb_sim/10);
SNRdb_th = -10:0.5:20;
SNR_th = 10.^(SNRdb_th/10);

% Theoretical BER for antipodal: Q(sqrt(2 * Eb/N0))
theoretical_error = 0.5 * erfc(sqrt(SNR_th));


simulation_error = zeros(size(SNR_sim));


for ii = 1:length(SNR_sim)
    N0 = E / SNR_sim(ii);
    sigma = sqrt(N0 / 2);

    % Random bits → antipodal mapping: 0 → +s(t), 1 → -s(t)
    bits = randi([0 1], 1, symbol_num);
    tx = zeros(symbol_num, length(t));

    for k = 1:symbol_num
        if bits(k) == 0
            tx(k, :) = s;
        else
            tx(k, :) = -s;
        end
    end
```

```
    % Add AWGN to each signal
    rx = tx + sigma * randn(size(tx));

    % Matched filter: projection onto phi
    projections = trapz(t, rx .* phi, 2);  % Each row projected

    % Detection: if < 0 → decide 1, else 0
    detected = projections < 0;
    simulation_error(ii) = mean(detected' ~= bits);
end

% Plot results
figure;
semilogy(SNRdb_sim, simulation_error, 'bo-', 'LineWidth', 1.2); hold on;
semilogy(SNRdb_th, theoretical_error, 'r-', 'LineWidth', 1.2);
xlabel('SNR (dB)');
ylabel('Bit Error Probability');
legend('Simulation', 'Theory', 'Location', 'southwest');
grid on;
title('BER vs E_b/N_0 for Antipodal Signaling');
ylim([1e-4 1]);
xlim([SNRdb_sim(1), SNRdb_sim(end)]);

%% Part a - Binary modulation

T = 0.01;               % symbol duration (s)
Fs = 1000;              % sampling frequency (Hz)
t = 0:1/Fs:T-1/Fs;      % time vector for one symbol
N = 5;                  % number of bits

% Define s(t)
s = sin(2*pi*100*t);

% Generate 5 random bits
rng(1);                 % fixed seed for reproducibility
bits = randi([0 1], 1, N);

% Initialize x(t)
x = [];

% Modulate each bit
for i = 1:N
    if bits(i) == 0
        x = [x s];      % 0 → s(t)
    else
        x = [x -s];     % 1 → -s(t)
    end
end

% Time vector for full signal
t_total = 0:1/Fs:N*T - 1/Fs;

% Plot
figure;
plot(t_total, x, 'LineWidth', 1.2);
xlabel('Time (s)');
ylabel('x(t)');
title('Modulated Signal x(t) for 5 Random Bits');
grid on;
```

```
% Display bits
disp('Random bits:');
disp(bits);

%% Part b

% Given parameters (should match Part a)
T = 0.01;
Fs = 1000;
t = 0:1/Fs:T-1/Fs;

% Orthonormal basis function
psi = sqrt(2/T) * sin(2*pi*100*t);

% Plot the basis function
figure;
plot(t, psi, 'k', 'LineWidth', 1.5);
xlabel('Time (s)');
ylabel('\psi(t)');
title('Orthonormal Basis Function \psi(t)');
grid on;

% Signal energy of s(t) is E_s = T/2
E_s = T / 2;
projection_value = sqrt(E_s);  % = sqrt(T/2)

% Constellation points: symmetric on 1D axis
x_vals = [-projection_value, projection_value];  % -s(t) and s(t)
bit_labels = {'bit = 1 (-s(t))', 'bit = 0 (s(t))'};

% Plot 1D constellation
figure;
stem(x_vals, [0 0], 'filled', 'LineWidth', 2);
text(x_vals(1), 0.02, bit_labels{1}, 'HorizontalAlignment', 'center');
text(x_vals(2), 0.02, bit_labels{2}, 'HorizontalAlignment', 'center');
xline(0, '--k');
ylim([-0.5 0.5]);
xlim([-1.1 1.1]*projection_value);
xlabel('\psi(t) direction');
ylabel('Amplitude (display only)');
title('Signal Space: 1D Binary Antipodal Constellation');
grid on;

%% Part c

% From Part (a): reuse x and t_total


% Noise variances
noise_vars = [1e-4, 1e-2, 1e0];

% Signal power (empirical average)
signal_power = mean(x.^2);

figure;

for i = 1:length(noise_vars)
    sigma2 = noise_vars(i);
```

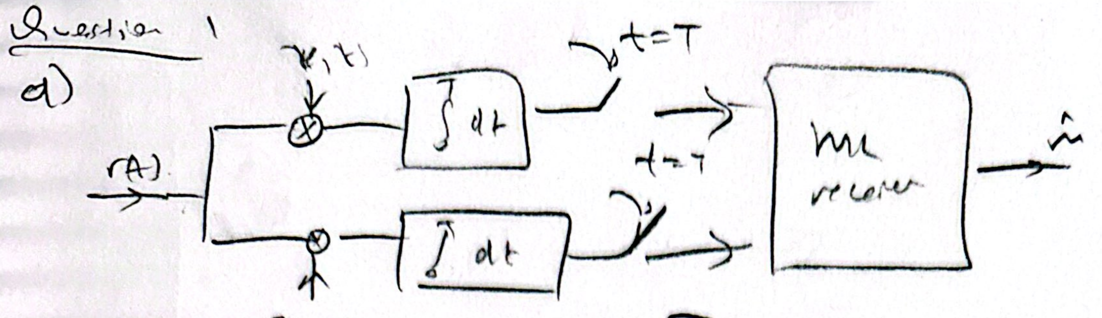
```
    noise = sqrt(sigma2) * randn(size(x));
    x_noisy = x + noise;

    % Compute SNR
    snr = signal_power / sigma2;
    snr_db = 10 * log10(snr);

    % Plot original and noisy signals
    subplot(length(noise_vars), 1, i);
    plot(t_total, x, 'b', 'DisplayName', 'Original'); hold on;
    plot(t_total, x_noisy, 'r', 'DisplayName', 'Noisy');
    title(sprintf('Noise Variance = %.0e | SNR = %.2f dB', sigma2, snr_db));
    xlabel('Time (s)');
    ylabel('Amplitude');
    legend; grid on;
end

%% Part e and g

Fs = 1000;                  % Sampling frequency
T = 0.01;                   % Bit duration
t = 0:1/Fs:T-1/Fs;          % Time vector
s = sin(2*pi*100*t);        % Base signal
phi = s / norm(s);          % Normalized basis function
E = trapz(t, s.^2);         % Signal energy
symbol_num = 1e5;           % Number of bits

% SNR values
SNRdb_sim = -10:2:20;
SNR_sim = 10.^(SNRdb_sim/10);
SNRdb_th = -10:0.5:20;
SNR_th = 10.^(SNRdb_th/10);

% Initialize arrays
simulation_error = zeros(size(SNR_sim));
theoretical_error = 0.5 * erfc(sqrt(SNR_th));  % Theoretical BER

% Simulation loop
for ii = 1:length(SNR_sim)
    N0 = E / SNR_sim(ii);
    sigma = sqrt(N0 / 2);

    bits = randi([0 1], 1, symbol_num);
    tx = (1 - 2 * bits) * sqrt(E);              % Antipodal mapping
    rx = tx + sigma * randn(1, symbol_num);     % AWGN channel

    detected = rx < 0;
    simulation_error(ii) = mean(detected ~= bits);
end

% Plot BER vs SNR
figure;
semilogy(SNRdb_sim, simulation_error, 'bo-', 'LineWidth', 1.2); hold on;
semilogy(SNRdb_th, theoretical_error, 'r-', 'LineWidth', 1.2);
xlabel('SNR (dB)');
ylabel('Symbol Error Probability');
legend('Simulation', 'Theory', 'Location', 'southwest');
grid on;
title('P_e vs E_s/N_0');
```

```
ylim([1e-4 1]);
xlim([SNRdb_sim(1), SNRdb_sim(end)]);

var_n = 1e-2;                          % Noise variance
std = sqrt(var_n);                     % Noise std dev
alpha = 0.01:0.01:0.99;                % Prior probability P(1)
map_rule = zeros(size(alpha));
ml_rule = zeros(size(alpha));

for i = 1:length(alpha)
    prob = rand(1, symbol_num) < alpha(i);          % bits with P(1) = α
    tx = (1 - 2 * prob) * sqrt(E);                  % 0 → +sqrt(E), 1 → -sqrt(E)
    received_vector = tx + std * randn(1, symbol_num); % AWGN

    % MAP decision threshold
    boundry = (var_n / (2 * sqrt(E))) * log(alpha(i) / (1 - alpha(i)));
    map_received = received_vector <= boundry;

    % ML decision: threshold at 0
    ml_received = received_vector < 0;

    % Error rates
    map_rule(i) = mean(map_received ~= prob);
    ml_rule(i) = mean(ml_received ~= prob);
end

% Plot MAP vs ML error
figure;
plot(alpha, map_rule, 'b-o', 'LineWidth', 1.2); hold on;
plot(alpha, ml_rule, 'r-s', 'LineWidth', 1.2);
xlabel('\alpha');
ylabel('P_e');
legend('MAP', 'ML');
title('P_e vs \alpha');
grid on;
```

Appendix b

Question 1

a)

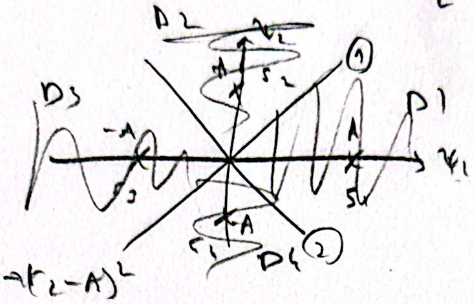


Let $A = \sqrt{\frac{E}{2}}$

ML rule

$\arg\min_{i \in \{1,2,3,4\}} \|r - s_i\|^2$

$s_1 = \begin{bmatrix} A \\ 0 \end{bmatrix}$ $s_2 = \begin{bmatrix} 0 \\ A \end{bmatrix}$ $s_3 = \begin{bmatrix} -A \\ 0 \end{bmatrix}$ $s_4 = \begin{bmatrix} 0 \\ -A \end{bmatrix}$



$\hat{m} = 1$

$(r_1 - A)^2 + r_2^2 < r_1^2 + (r_2 - A)^2$

$-2r_1 A + A^2 < -2r_2 A + A^2 \Rightarrow \boxed{r_1 > r_2}$ ①

$(r_1 - A)^2 + r_2^2 < r_1^2 + (r_2 + A)^2$

$-2r_1 A + A^2 < 2r_2 A + A^2$ $\boxed{r_1 > -r_2}$ ②

Because constellation points are symmetric, $\boxed{P_e = \frac{1}{4} \cdot 4P_{e,1,2,3,4} = P_{e,1,2,3,4}}$

$P_{e,1} = 1 - P(r_1 > r_2,\ r_1 > -r_2)$

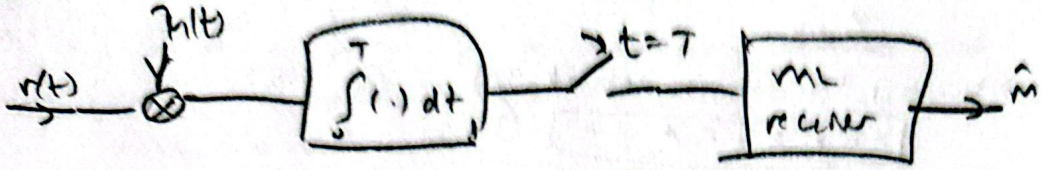
$1 - P(A + n_1 > n_2)\, P(A + n_1 > -n_2)$

$1 - P(n_1 - n_2 > -A)\, P(n_1 + n_2 > -A)$

N_0 still

$P_{e,1} = 1 - \left(1 - Q^2\left(\frac{A}{\sqrt{2N_0}}\right)\right) = \boxed{Q^2\left(\frac{A}{\sqrt{2N_0}}\right)}$

d) Because 1-D, only one demodulator is enough. Question 2

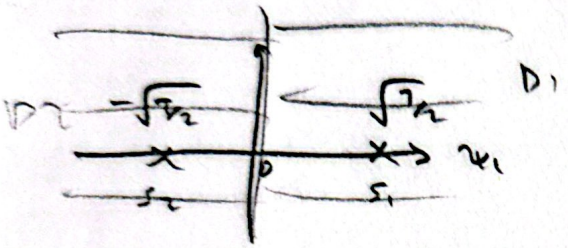


ML rule

$\arg\min_i \|r - s_i\|^2$

$i \in \{1, 2\}$

$\underline{s}_1 = \begin{bmatrix} \sqrt{\frac{T}{2}} \\ 0 \end{bmatrix}$ $\underline{s}_2 = \begin{bmatrix} -\sqrt{\frac{T}{2}} \\ 0 \end{bmatrix}$



$$\hat{m} = \begin{cases} 1, & r > 0 \\ 2, & r < 0 \end{cases}$$

$$P_{e,1} = 1 - P(r > 0) = 1 - P\left(\sqrt{\frac{T}{2}} + n > 0\right)$$

$$= 1 - P\left(n > -\sqrt{\frac{T}{2}}\right)$$

$$= 1 - 1 + Q\left(\frac{\sqrt{\frac{T}{2}}}{\sqrt{\frac{N_0}{2}}}\right) = \boxed{Q\left(\sqrt{\frac{T}{N_0}}\right)}$$

$$P_{e,b} = \frac{1}{2} \cdot 2 \cdot Q\left(\sqrt{\frac{T}{N_0}}\right) = \boxed{Q\left(\sqrt{\frac{T}{N_0}}\right)}$$

3- Asymmetric Modulation

Inner product with itself should be equal to 1.

$2\sin x\cos x = \sin 2x$

$\cos^2 x - \sin^2 x = \cos 2x$

a) $\int_0^{T_s} \psi_1^2(t)\,dt = 1 \Rightarrow \int_0^{T_s} c^2(\cos(2\pi f_0 t) + \sin(2\pi f_0 t))^2\,dt$

$= c^2 \int_0^{T_s} [\underbrace{\cos^2(2\pi f_0 t) + \sin^2(2\pi f_0 t)}_{=1} + \underbrace{2\sin(2\pi f_0 t)\cos(2\pi f_0 t)}_{=\sin(4\pi f_0 t)}]\,dt$

$\sin(4\pi f_0 t)$ → integral of $\sin 2x$ over a full period is 0.

$\Rightarrow c^2 \cdot T_s = 1 \Rightarrow \boxed{c = \frac{1}{\sqrt{T_s}}}$

As second basis function, $\psi_2(t)$, we can define it as follows in order to form a orthogonal basis function

$$2\psi_2(t) \rightarrow \frac{1}{\sqrt{T_s}}(\cos 2\pi f_0 t - \sin(2\pi f_0 t))$$

check $\langle \psi_1(t), \psi_2(t) \rangle \stackrel{?}{=} 0$

$$\frac{1}{T_s}\int_0^{T_s} (\cos(2\pi f_0 t) + \sin(2\pi f_0 t))(\cos(2\pi f_0 t) - \sin(2\pi f_0 t))\,dt$$

$$\Rightarrow \frac{1}{T_s}\int_0^{T_s} (\cos^2(2\pi f_0 t) - \sin^2(2\pi f_0 t))\,dt = 0$$

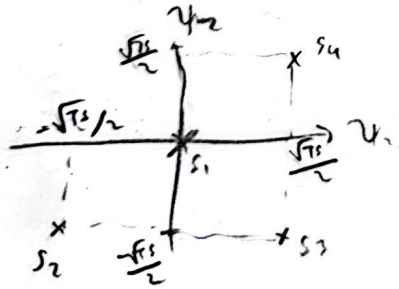
$\cos(4\pi f_0 t)$ → integral of $\cos 2x$ over a full period is zero.

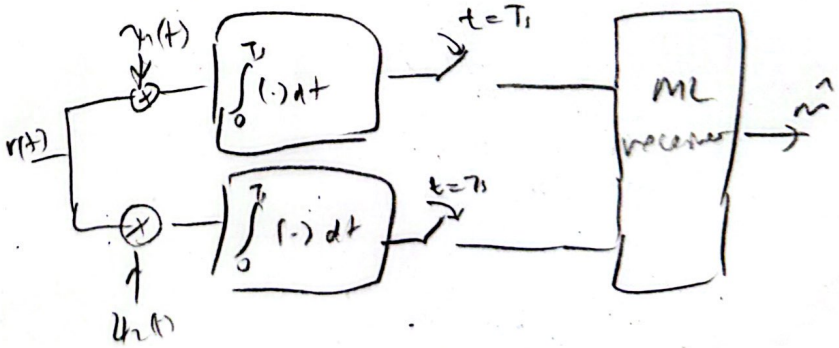
$\Rightarrow \psi_1(t)$ and $\psi_2(t)$ are orthogonal basis functions.

b) $\psi_1(t) = \frac{1}{\sqrt{T_s}}(\cos(2\pi f_0 t) + \sin(2\pi f_0 t))$ $\quad \psi_2(t) = \frac{1}{\sqrt{T_s}}(\cos(2\pi f_0 t) - \sin(2\pi f_0 t))$

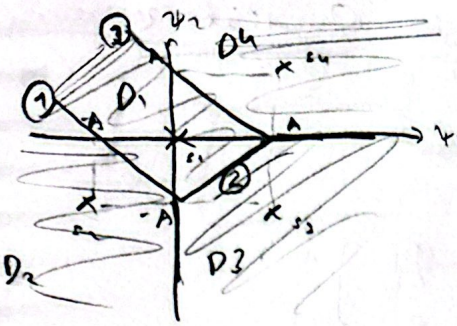
$s_1 = 0$ $\quad s_2 = -\cos(2\pi f_0 t)$ $\quad s_3 = \sin(2\pi f_0 t)$ $\quad s_4 = \cos(2\pi f_0 t)$

$$\underline{s}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \underline{s}_2 = \begin{bmatrix} -\frac{\sqrt{T_s}}{2} \\ -\frac{\sqrt{T_s}}{2} \end{bmatrix} \quad \underline{s}_3 = \begin{bmatrix} \frac{\sqrt{T_s}}{2} \\ -\frac{\sqrt{T_s}}{2} \end{bmatrix} \quad \underline{s}_4 = \begin{bmatrix} \frac{\sqrt{T_s}}{2} \\ \frac{\sqrt{T_s}}{2} \end{bmatrix}$$





Because probabilities are equal, the expressions simply to L2 Norm $(\|r - s_i\|^2)$, and therefore we identify the decision regions by ML rule



$A = \sqrt{\frac{T_s}{2}}$

$\hat{m} = 1$

$-r_1^2 + r_1^2 < (r_1 + A)^2 + (r_2 + A)^2$

$0 < r_1 A + r_2 A + A^2$ (1)

$-r_1^2 + r_2^2 < (r_1 - A)^2 + (r_2 + A)^2$

$0 < -r_1 A + r_2 A + A^2$ (2)

$-r_1^2 + r_2^2 < (r_1 - A)^2 + (r_2 - A)^2$

$0 < -r_1 A - r_2 A + A^2$ (3)

$$\hat{m} = \begin{cases} 1, & \frac{-\sqrt{T_s}}{2} < r_1 + r_2 < \frac{\sqrt{T_s}}{2}, \; r_1 - r_2 < \frac{\sqrt{T_s}}{2} \\ 2, & r_1 < 0, \; r_1 + r_2 < -\frac{\sqrt{T_s}}{2} \\ 3, & r_1 > 0, \; r_2 < 0, \; r_1 - r_2 > \frac{\sqrt{T_s}}{2} \\ 4, & r_2 > 0, \; r_1 + r_2 < \frac{\sqrt{T_s}}{2} \end{cases}$$

c) $d_{min} = \sqrt{\frac{T_s}{2}}$

Loose bound

$$(M-1)\, Q\left(\sqrt{\frac{d_{min}^2}{2N_0}}\right) = (4-1)\, Q\left(\sqrt{\frac{T_s}{4N_0}}\right) \Rightarrow \boxed{P_e \leq 3\, Q\left(\sqrt{\frac{T_s}{4N_0}}\right)}$$

Union bound

$d_{12} = d_{21} = d_{13} = d_{31} = d_{14} = d_{41} = \sqrt{\frac{T_s}{2}}$

$d_{23} = d_{32} = d_{24} = d_{42} = \sqrt{T_s}$

$d_{34} = d_{43} = \sqrt{2T_s}$

$$P_e \leq \frac{1}{M} \sum_{m=1}^{M} \sum_{\substack{k=1 \\ m \neq k}}^{M} Q\left(\sqrt{\frac{d_{ik}^2}{2N_0}}\right)$$

$$P_e \leq \frac{6}{4}\, Q\left(\sqrt{\frac{T_s}{4N_0}}\right) + \frac{4}{4} Q\left(\sqrt{\frac{T_s}{2N_0}}\right) + \frac{2}{4}\, Q\left(\sqrt{\frac{T_s}{N_0}}\right)$$

As seen, union bound is more detailed and gives more information about the UPPER BOUND, which is more precise.

d) $P_{e,1} = 1 - [P(-A < r_1 + r_2 < A, \; r_1 - r_2 < A)]$

$P_{e,1} = 1 - [(P(r_1 + r_2 > -A) - P(r_1 + r_2 > A)) \cdot P(r_1 - r_2 < A)]$

$P_{e,1} = 1 - [(P(n_1 + n_2 > -A) - P(n_1 + n_2 > A)) \cdot P(n_1 - n_2 < A)]$; $Var(n_1 + n_2) = Var(n_1) + Var(n_2)$

$P_{e,1} = 1 - \left[\left(1 - Q\left(\frac{A}{\sqrt{N_0}}\right) - Q\left(\frac{A}{\sqrt{N_0}}\right)\right)\right]$

$$\boxed{P_{e,1} = 2\, Q\left(\frac{A}{\sqrt{N_0}}\right)}$$