Faculty of Engineering
Computer Department
Communications (ELC 325B) – Spring 2023

Assignment 3

Submitted to

Eng. Mohamed Khaled

Team Members

Num	**Full Name in ARABIC**	SEC	BN
1	كريم محمود كمال محمد	2	10
2	محمد وليد فتحي	2	20

Table of contents:

List of Figures

Faculty of Engineering
Computer Department
Communications (ELC 325B) – Spring 2023

1. Part One

1.1 Gram-Schmidt Orthogonalization

Gram-Schmidt Orthogonalization is a technique used in digital communication to transform a set of input signals into an orthonormal basis that can be used to represent the signals in a signal space.

$\boxed{1}$

$E_1 = \int_0^T s_1^2(t) = \int_0^1 1\,dt = t\big|_0^1 = \boxed{1}$

$\boxed{1}$ $\phi_1(t) = \frac{s_1(t)}{\sqrt{E_1}} = \frac{s_1(t)}{\sqrt{1}} = \boxed{s_1(t)}$

$\boxed{2}$ $g_2(t) = s_2(t) - s_{21}\phi_1(t)$

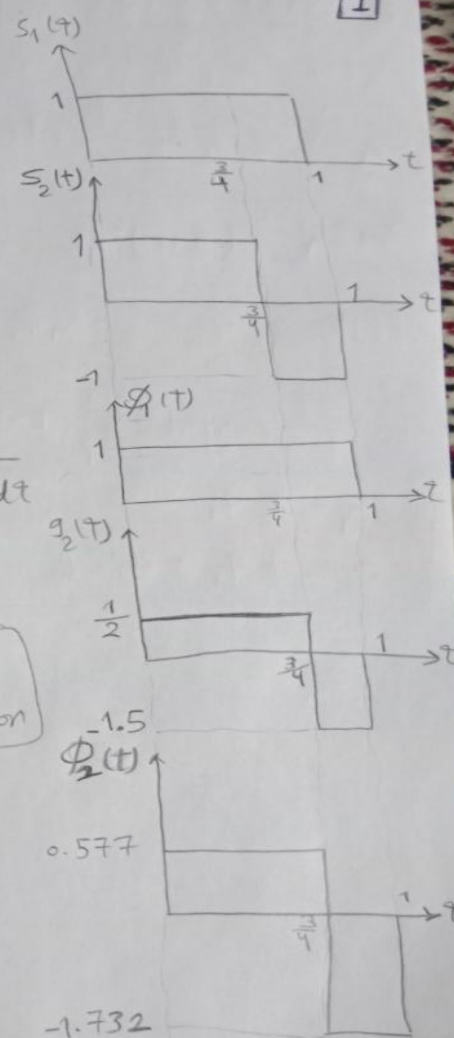
$s_{21} = \int_0^T s_2(t)\phi_1(t)\,dt = \int_0^{3/4} 1\,dt + \int_{3/4}^1 -1*1\,dt$

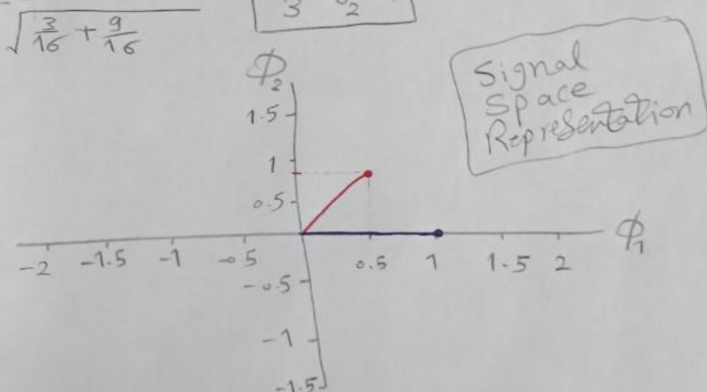
$= \frac{3}{4} + \left(\frac{-1}{4}\right) = \boxed{\frac{1}{2}}$

$g_2(t) = \boxed{s_2(t) - \frac{1}{2}s_1(t)}$

$$\phi_2(t) = \frac{g_2(t)}{\sqrt{\int_0^T g_2^2(t)\,dt}} = \frac{g_2(t)}{\sqrt{\int_0^{0.75}\left(\frac{1}{2}\right)^2 dt + \int_{0.75}^1 (-1.5)^2 dt}}$$

$$= \frac{g_2(t)}{\sqrt{\frac{3}{16} + \frac{9}{16}}} = \boxed{\frac{2\sqrt{3}}{3} g_2(t)}$$





$s_1(t) = s_{11}\phi_1 + s_{12}\phi_2$

$s_{11} = \int_0^T s_1\phi_1\,dt = \int_0^1 1\,dt = \boxed{1}$

$s_{12} = \int_0^T s_1\phi_2\,dt = \int_0^{3/4} 0.577\,dt + \int_{3/4}^1 -1.732\,dt = 0.43275 - 0.433 = \boxed{\text{Zero}}$

→ It is clear that we can represent s_1 in ϕ_1 & ϕ_2 axes without calculations (Just for explanation)

$\boxed{2}$

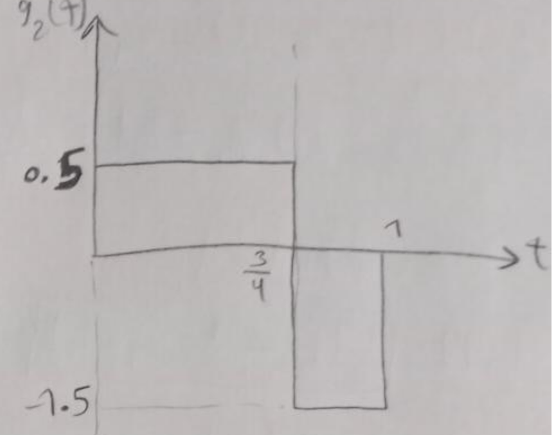
$$E_2 = \int_0^{\frac{3}{4}} 1\,dt + \int_{\frac{3}{4}}^{1} -1\,dt = \boxed{\tfrac{1}{2}}$$

$$S_2(t) = S_{21}\,\phi_1 + S_{22}\,\phi_2$$

$$S_{21} = \int_0^T S_2\,\phi_1\,dt = \int_0^{\frac{3}{4}} 1\,dt + \int_{\frac{3}{4}}^{1} -1\,dt = \frac{3}{4} - \frac{1}{4} = \boxed{\tfrac{1}{2}}$$

$$S_{22} = \sqrt{\int_0^T (g_2(t))^2\,dt}$$

$$g_2(t) = S_{22}\,\phi_2 = S_2 - S_{21}\phi_1 = S_2 - \frac{1}{2}\phi_1$$



$$S_{22} = \sqrt{\int_0^{\frac{3}{4}} (0.5)^2\,dt + \int_{\frac{3}{4}}^{1} (-1.5)^2} = \frac{\sqrt{3}}{2} = \boxed{0.866}$$

$$\boxed{\begin{aligned} \therefore\ S_1 &= 1\,\phi_1 \\ S_2 &= \tfrac{1}{2}\phi_1 + \tfrac{\sqrt{3}}{2}\phi_2 \end{aligned}} \Longrightarrow \text{Signal Representation Space}$$

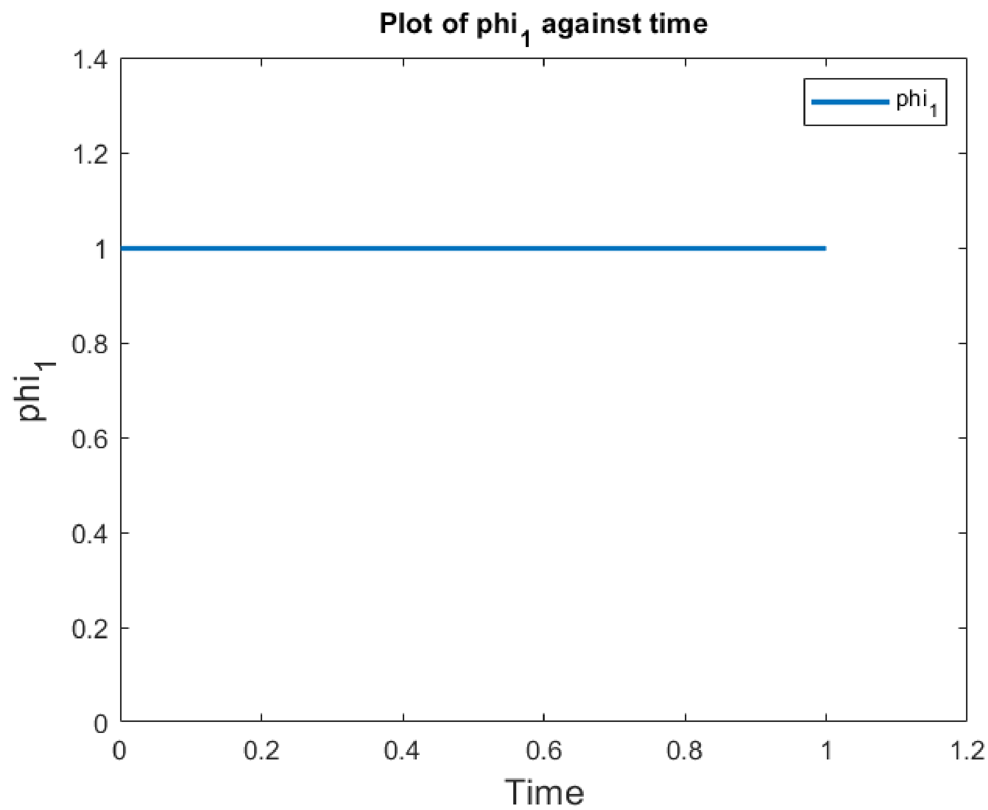


Figure 1 Φ1 VS time after using the GM_Bases function

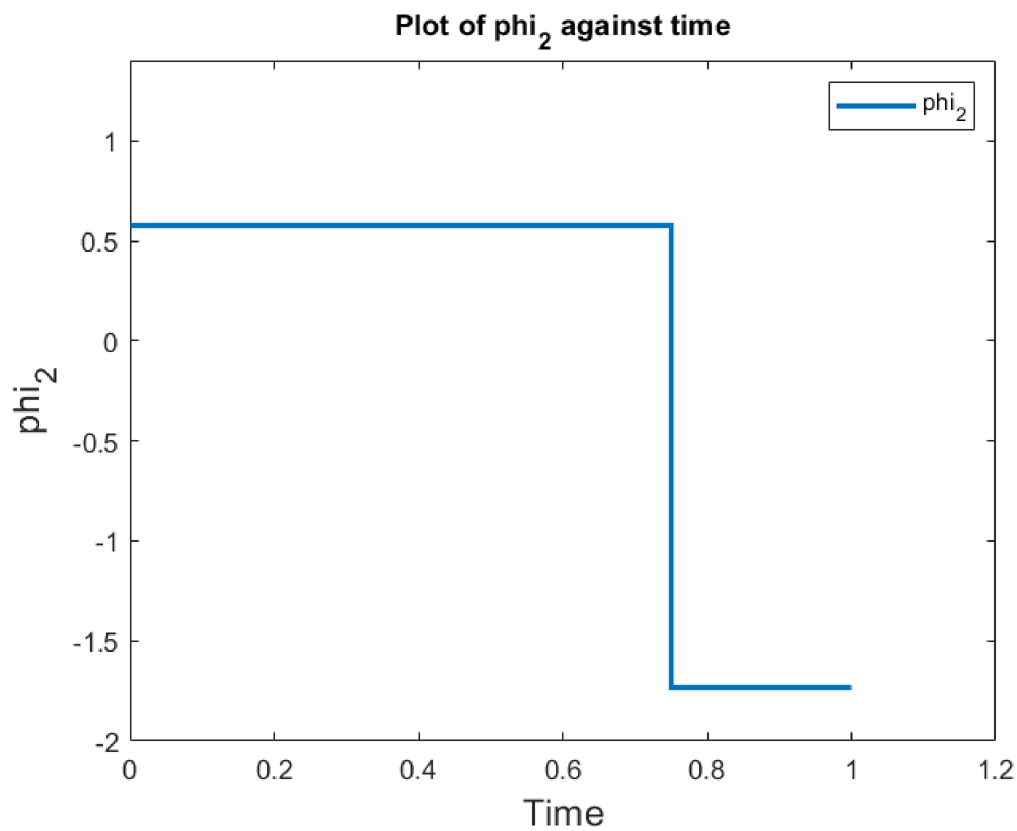


Figure 2 Φ2 VS time after using the GM_Bases function

1.2 Signal Space Representation

Here we represent the signals using the base functions.

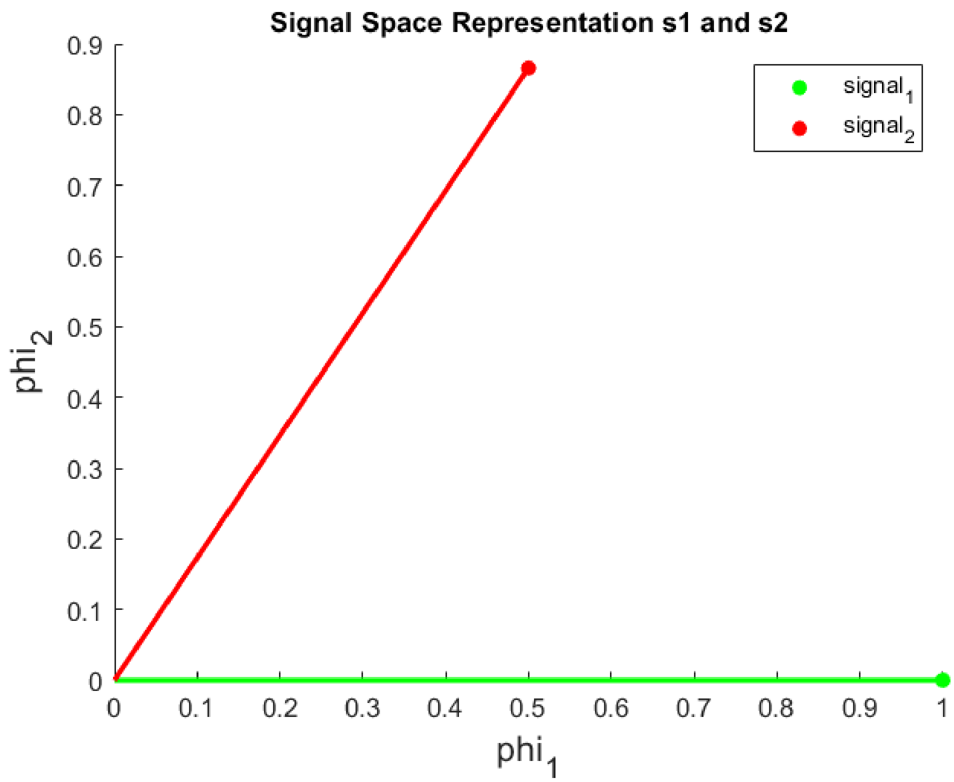


Figure 3 Signal Space representation of signals s1,s2

1.3 Signal Space Representation with adding AWGN

-the expected real points will be solid and the received will be hollow

Case 1: $10\, log(E/\sigma^2) \;=\; 10\, dB$

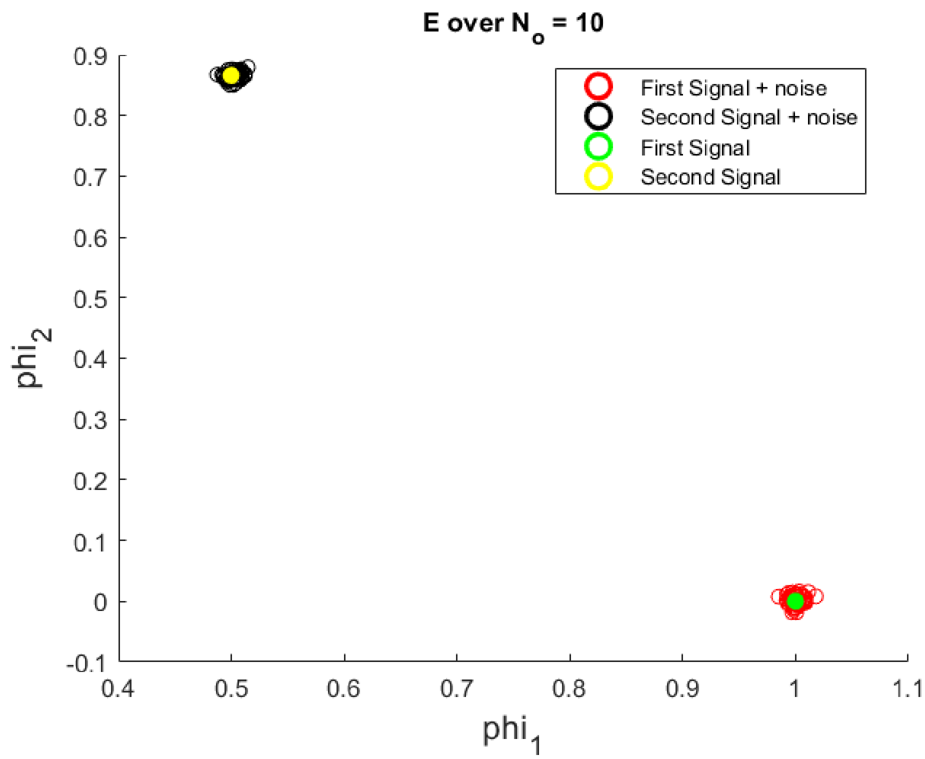


Figure 4 Signal Space representation of signals s1,s2 with E/σ¬2 =10dB

Case 2: $10\,log(E/\sigma^2) \;=\; 0\,dB$

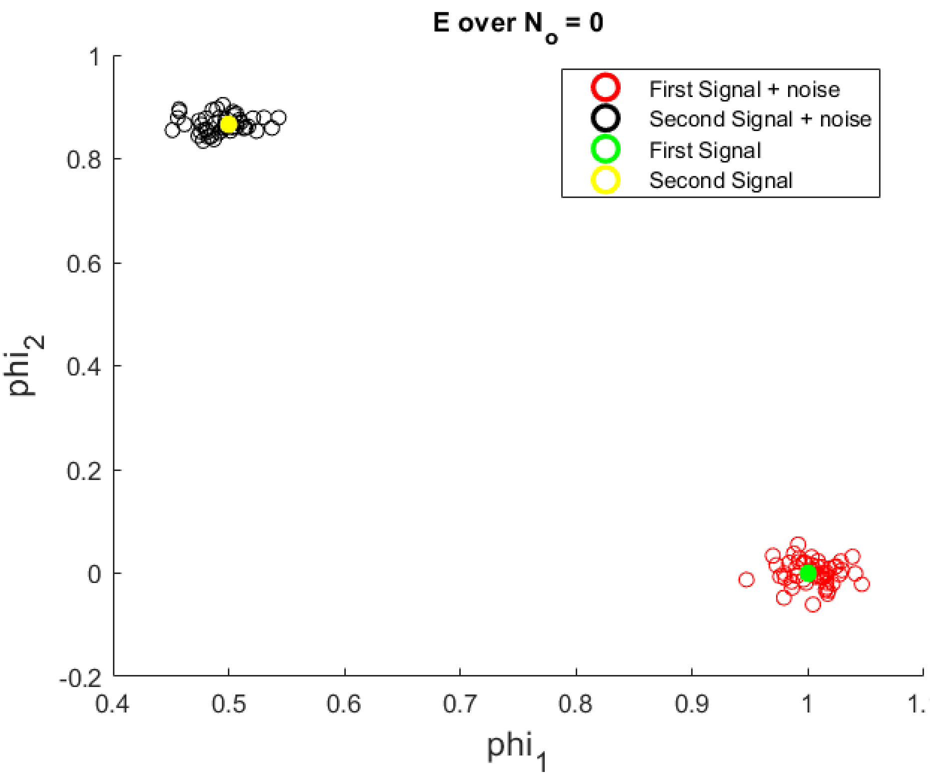


Figure 5 Signal Space representation of signals s1,s2 with E/σ¬2 =0dB

Case 3: $10\ log(E/\sigma^{2})\ =\ -\ 5\ dB$

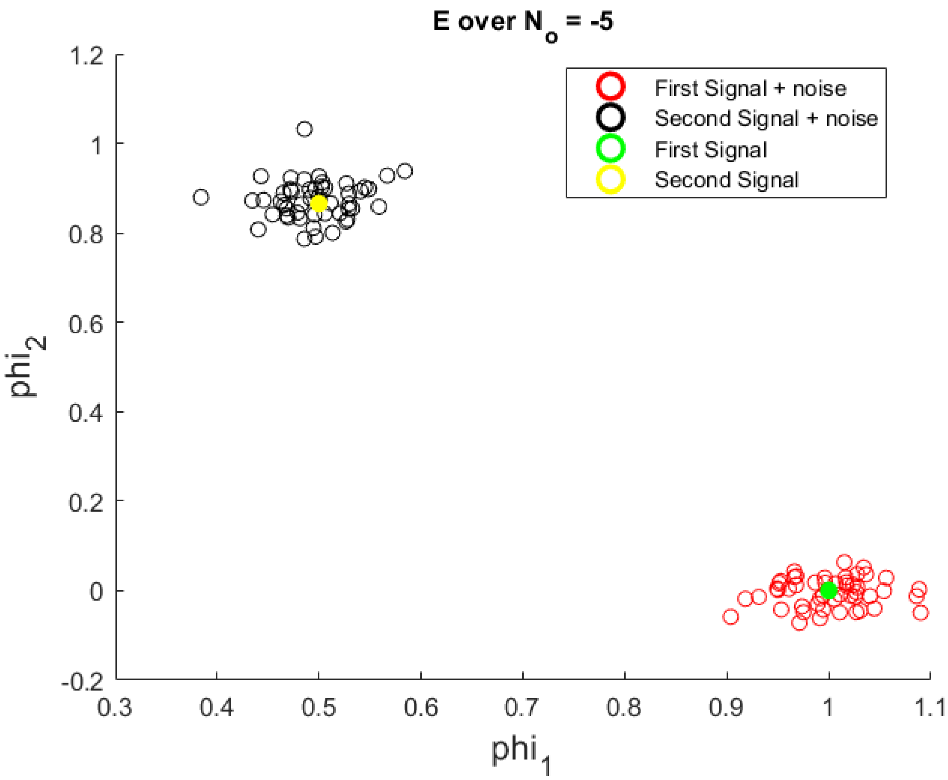


Figure 6 Signal Space representation of signals s1,s2 with E/σ¬2 =-5dB

1.4 Noise Effect on Signal Space

- The AWGN affects the signal space by spreading out the signal points and making it more difficult to distinguish between different signals.
- The effect of noise increases with increasing variance (σ^2) which means that the noise has a greater effect on the signal, leading to a more diffuse and difficult-to-distinguish signal space.

Appendix A: Codes for Part One:

A.1 Code for Gram-Schmidt Orthogonalization

```
% Input Signals: first_signal, second_signal
    % ==> two 1xN vectors
    % Output orthonormal bases: phi_1, phi_2:
    % ==> two 1xN vectors
    function [phi_1, phi_2] = Gramm_Schmidt(s1, s2)
    %%%%%%%%% STEPS FROM SLIDE 4 %%%%%%%%%
    % Initialize phi1 and phi2 as zero vectors
    phi_1 = zeros(1, length(s1));
    phi_2 = zeros(1, length(s2));
    % Calculate phi1
    phi_1 = s1 / norm(s1);
    % Calculate s2_1
    s2_1 = dot(s2, phi_1);
    % Calculate phi2
    % orthogonalize s2 with respect to phi1
    g2 = s2 - s2_1 * phi_1;
    % normalize g2 to obtain phi2
    phi_2 = g2 / norm(g2);
    end
```

A.2 Code for Signal Space representation

```
    % Inputs: the signal, phi_1, phi_2

    % Outputs: v1, v2:

    % ==> the projections of the signal over phi_1 and phi_2

function [first_projection, second_projection] = Signal_Space_Representation(signal, phi_1, phi_2)

    samples = 1000;


    % Calculate the projections of s onto phi1 and phi2 (Divide by sqrt(samples)
to normalize the vectors)

    first_projection = dot(signal, phi_1)/sqrt(samples);

    second_projection = dot(signal, phi_2)/sqrt(samples);


    end
```

A.3 Code for plotting the bases functions

```
% test Gramm schmitt
    [phi_1, phi_2] = Gramm_Schmidt(first_signal, second_signal);

    time = linspace(0, 1, length(phi_1));

    % plotting the orthonormal bases (phi_1, phi_2) against time
    % phi_1
    figure(3);
    stairs(time, phi_1*sqrt(samples), 'LineWidth', 2);
    axis([0 1.2 0 1.4]);
    xlabel('Time', 'FontSize', 14);
    ylabel('phi_1', 'FontSize', 14);
    title('Plot of phi_1 against time');
    legend('phi_1');

    % phi_2
    figure(4);
    stairs(time, phi_2*sqrt(samples), 'LineWidth', 2);
    axis([0 1.2 -2 1.4]);
    xlabel('Time', 'FontSize', 14);
    ylabel('phi_2', 'FontSize', 14);
    title('Plot of phi_2 against time');
    legend('phi_2');
```

A.4 Code for plotting the Signal space Representations

```
    % the projections of first signal
 [first_projection_s1, second_projection_s1] = Signal_Space_Representation(first_signal,phi_1, phi_2 );

    % the projections of second signal
 [first_projection_s2, second_projection_s2] = Signal_Space_Representation(second_signal,phi_1, phi_2 );

    disp('The projections of first signal are:');
    disp(first_projection_s1);
    disp(second_projection_s1);

    disp('The projections of second signal are:');
    disp(first_projection_s2);
    disp(second_projection_s2);
    % scatter the projections of first and second signal (This draw a point
representing each signal in the signal space)
    figure(5);
    scatter(first_projection_s1, second_projection_s1, 'filled', 'g');
    hold on;
    scatter(first_projection_s2, second_projection_s2 , 'filled', 'r');
    xlabel('phi_1', 'FontSize', 14);
    ylabel('phi_2', 'FontSize', 14);
    title('Signal Space Representation s1 and s2');

    % Construct the vectors from the origin (These vectors are connected to the
projections of the signals)
    origin = [0 0];
    line([origin(1) first_projection_s1], [origin(2) second_projection_s1], 'Color',
'g', 'LineWidth', 2);
    line([origin(1) first_projection_s2], [origin(2) second_projection_s2], 'Color',
'r', 'LineWidth', 2);
    legend('signal_1', 'signal_2');
```

A.5 Code for the effect of noise on the Signal space Representations

First: Function for Adding Noise

```
    % Inputs:
    % s: a 1xN vector that represents the input signal
    % sigma2: the variance of the additive white Gaussian noise
    % Outputs:
    % r: a 1xN vector that represents the received signal
    function r = noise(s, Eb_over_N0)
    Eb = 1; % Energy per bit
    % Calculate the noise variance
    N0=Eb/(10^(Eb_over_N0/10));
    sigma2=N0/2;
    % Generate the noise
    w = sqrt(sigma2) * randn(1, length(s));
    % Add the noise to the input signal
    r = s + w;
    end
```

Second: Plotting

```
    E_over_n= [10, 0, -5];
    E_over_n_length = length(E_over_n);
    number_of_samples = 50;
    % figure number (Only for plotting purposes)
    fig_num = 6;
    % Generate samples of r1(t) and r2(t) using first and second signals for each
E_over_n
    % ==> r1(t) = s1(t) + w(t) &&  r2(t) = s2(t) + w(t)
    % ==> w(t) is a zero mean AWGN with variance σ^2
    for i = 1:E_over_n_length

        % Figure number
        figure(fig_num+i-1);
        % σ^2 of the noise based on the E_over_n
        % One call to function noise() for each signal will yield one point in the
signal space
        % Thus, we call the function noise() 50 times for each signal to generate 50
points in the signal space
        for j = 1:number_of_samples
```

```
            % Adding Noise to the signals
            r_1 = noise(first_signal, E_over_n(i));
            r_2 = noise(second_signal, E_over_n(i));

            % signal space representation
            [first_projection_r1, second_projection_r1] =
Signal_Space_Representation(r_1, phi_1, phi_2);
            [first_projection_r2, second_projection_r2] =
Signal_Space_Representation(r_2, phi_1, phi_2);

            hold on;
            scatter(first_projection_r1, second_projection_r1, 'r');
            hold on;
            scatter(first_projection_r2, second_projection_r2, 'k');
        end

        % Plot the signal points
        scatter(first_projection_s1, second_projection_s1, 50 , 'g', 'filled');
        hold on;
        scatter(first_projection_s2, second_projection_s2, 50 , 'y', 'filled');

        xlabel('phi_1', 'FontSize', 14);
        ylabel('phi_2', 'FontSize', 14);
        title([‘E_over_n = ' , num2str(E_over_n(i))]);


        h(1)=plot(nan, nan, 'o', 'MarkerSize', 10,'Linewidth', 2, 'DisplayName',
'First Signal + noise', 'color', 'r');

        h(2)=plot(nan, nan, 'o', 'MarkerSize', 10,'Linewidth', 2, 'DisplayName',
'Second Signal + noise', 'color', 'k');

        h(3)=plot(nan, nan, 'o', 'MarkerSize', 10,'Linewidth', 2, 'DisplayName',
'First Signal', 'color', 'g');

        h(4)=plot(nan, nan, 'o', 'MarkerSize', 10,'Linewidth', 2, 'DisplayName',
'Second Signal', 'color', 'y');

        legend(h, 'Location', 'best');

    end
```

Signals Initialization

```
% initialize the signals
samples = 1000;
t= linspace(0, 1, samples);
first_signal = ones(1, samples);
second_signal = zeros(1, samples);

% construct the second signal (-1 above 0.75 and 1 below 0.75)
second_signal(t>0.75) = -1;
second_signal(t<=0.75) = 1;
```