**Assignment 3**

**Team Members**

|  |  |  |  |
| --- | --- | --- | --- |
| **Num** | **Full Name in ARABIC** | SEC | BN |
| 1 | عبدالرحمن محمد سالم حسن | 1 | 38 |
| 2 | احمد فوزى محمد إبراهيم | 1 | 8 |

**Table of contents:**

[1. Part One 3](#_Toc135281088)

[1.1 Gram-Schmidt Orthogonalization 3](#_Toc135281089)

[1.2 Signal Space Representation 5](#_Toc135281090)

[1.3 Signal Space Representation with adding AWGN 6](#_Toc135281091)

[1.4 Noise Effect on Signal Space 9](#_Toc135281092)

[2. Appendix A: Codes for Part One: 10](#_Toc135281093)

[A.1 Code for Gram-Schmidt Orthogonalization 10](#_Toc135281094)

[A.2 Code for Signal Space representation 11](#_Toc135281095)

[A.3 Code for plotting the bases functions 11](#_Toc135281096)

[A.4 Code for plotting the Signal space Representations 12](#_Toc135281097)

[A.5 Code for effect of noise on the Signal space Representations 15](#_Toc135281098)

**List of Figures**

[Figure 1 Φ1 VS time after using the GM\_Bases function 3](#_Toc135281077)

[Figure 2 Φ2 VS time after using the GM\_Bases function 4](#_Toc135281078)

[Figure 3 Signal Space representation of signals s1,s2 5](#_Toc135281079)

[Figure 4 Signal Space representation of signals s1,s2 with E/σ¬2 =10dB 6](#_Toc135281080)

[Figure 5 Signal Space representation of signals s1,s2 with E/σ¬2 =0dB 7](#_Toc135281081)

[Figure 6 Signal Space representation of signals s1,s2 with E/σ¬2 =-5dB 8](#_Toc135281082)

# Part One

## 1.1 Gram-Schmidt Orthogonalization

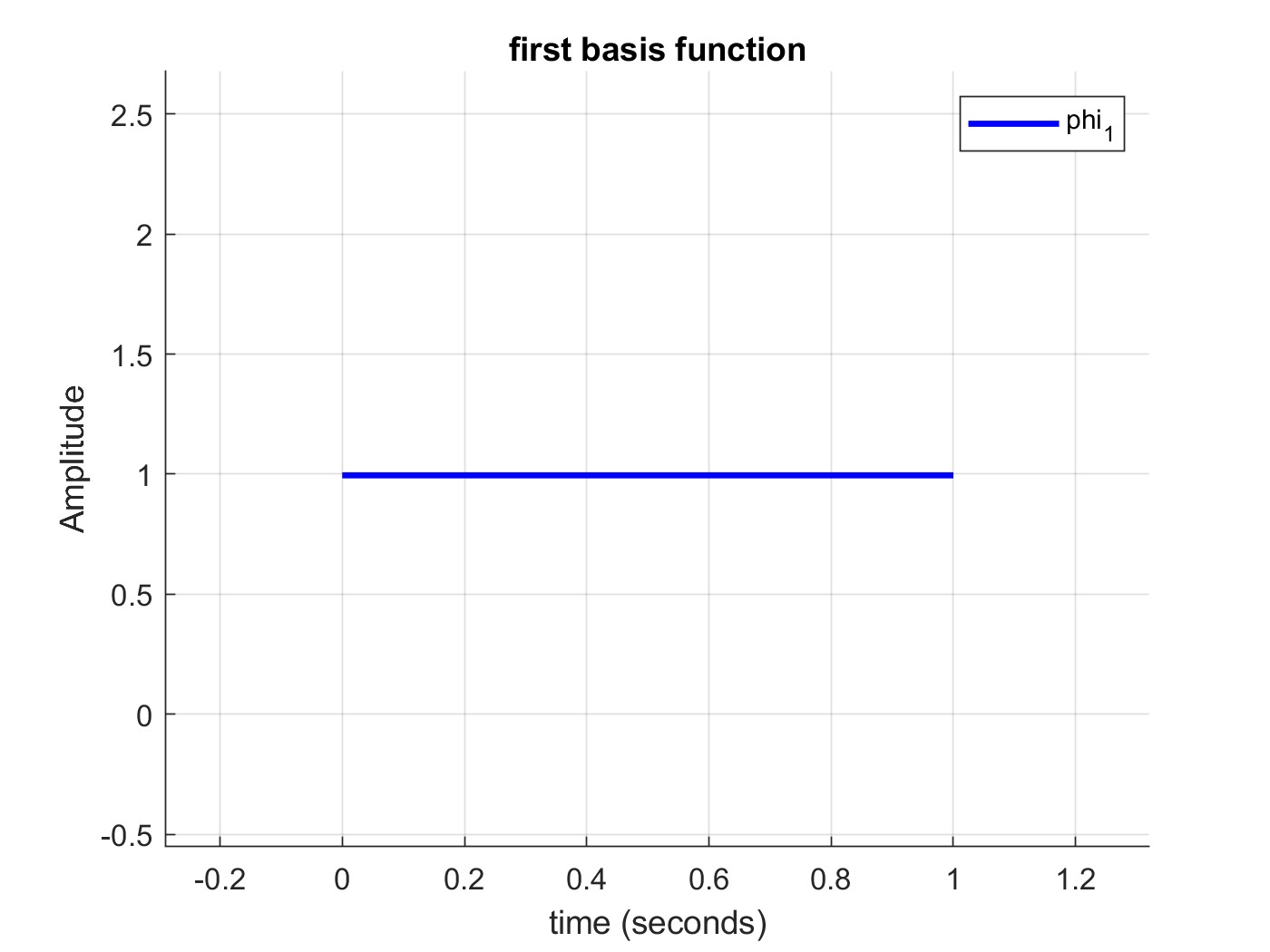
Here we are getting the basis functions of the 2 signals s1,s2 where s1 or s2 can be formed using equations using only phi1 and phi2 and some factors and also the dot product between phi1 and phi2 is 0

Figure Φ1 VS time after using the GM\_Bases function

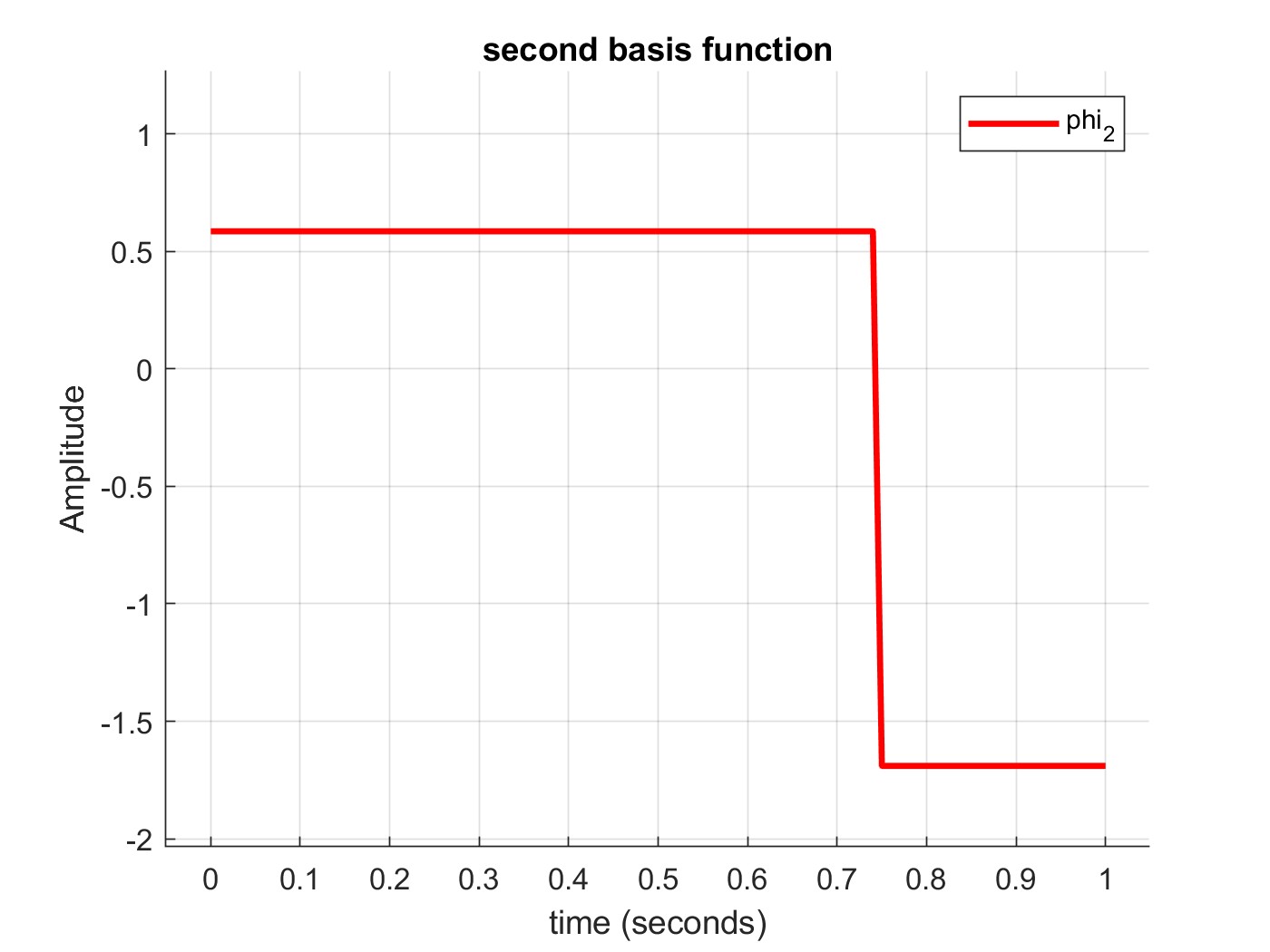


Figure Φ2 VS time after using the GM\_Bases function

## 1.2 Signal Space Representation

Here we represent the signals using the base functions.

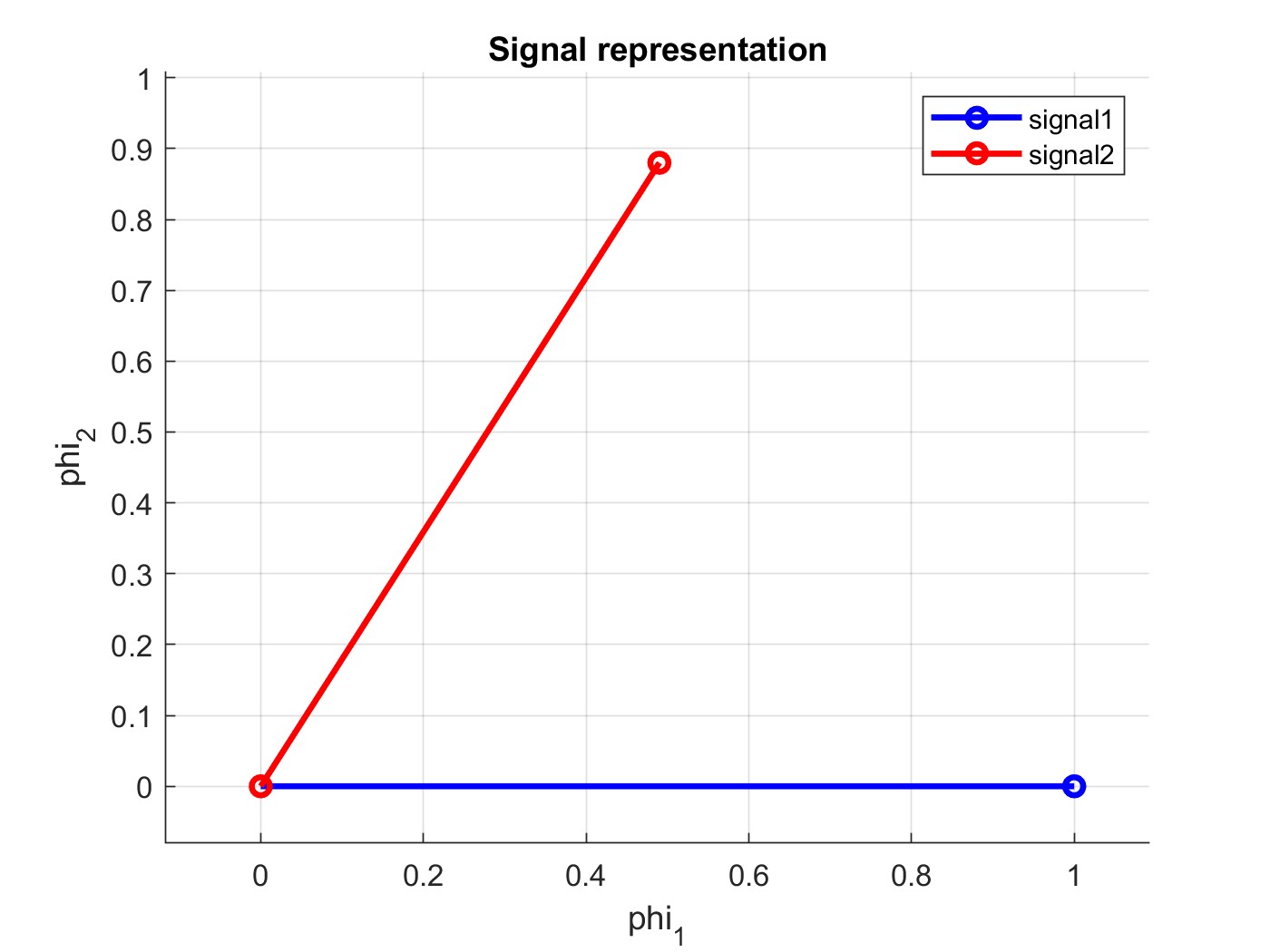


Figure Signal Space representation of signals s1,s2

## 1.3 Signal Space Representation with adding AWGN

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

**Case 1**:

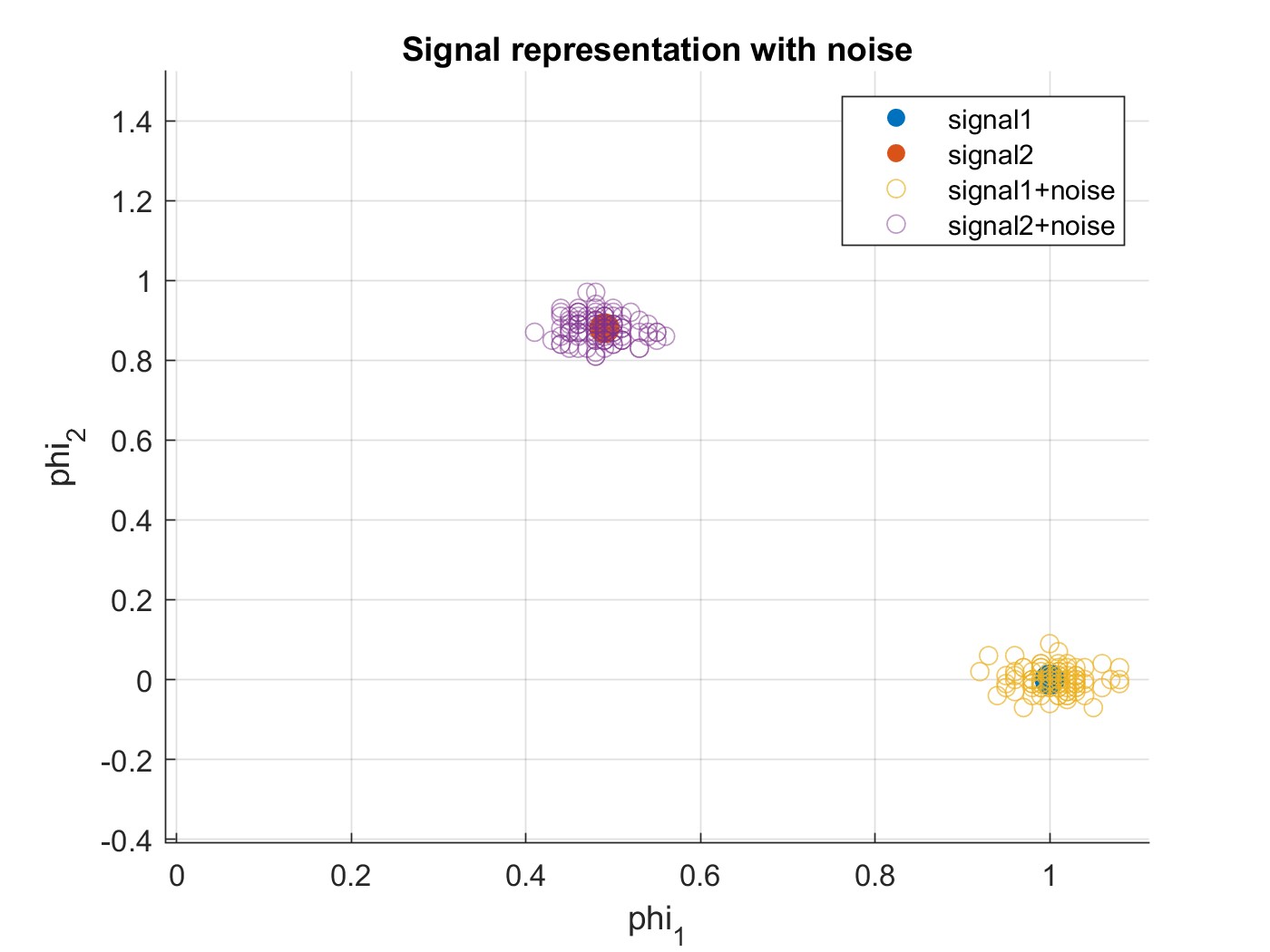


Figure Signal Space representation of signals s1,s2 with E/σ¬2 =10dB

**Case 2**:

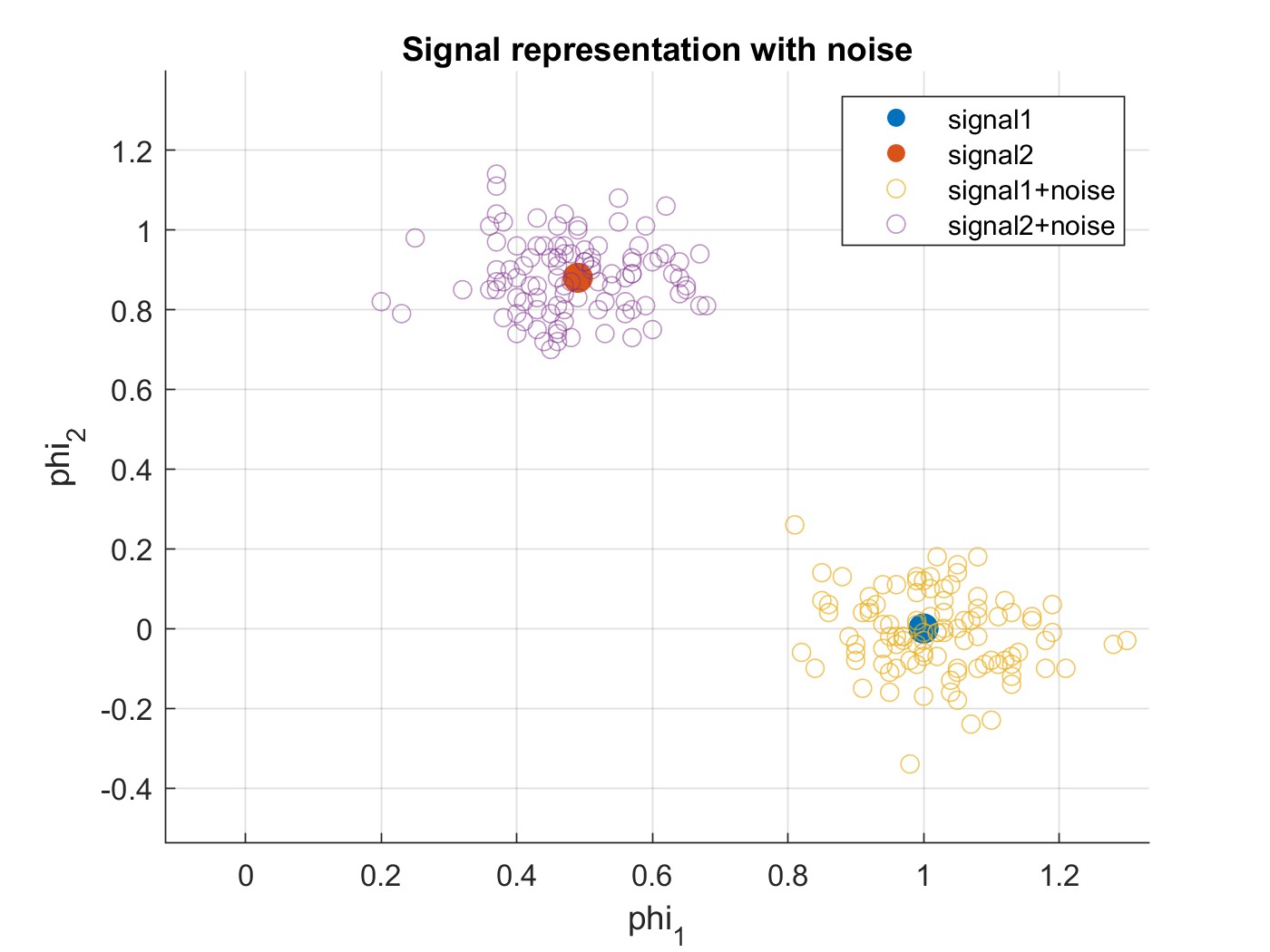


Figure Signal Space representation of signals s1,s2 with E/σ¬2 =0dB

**Case 3**:

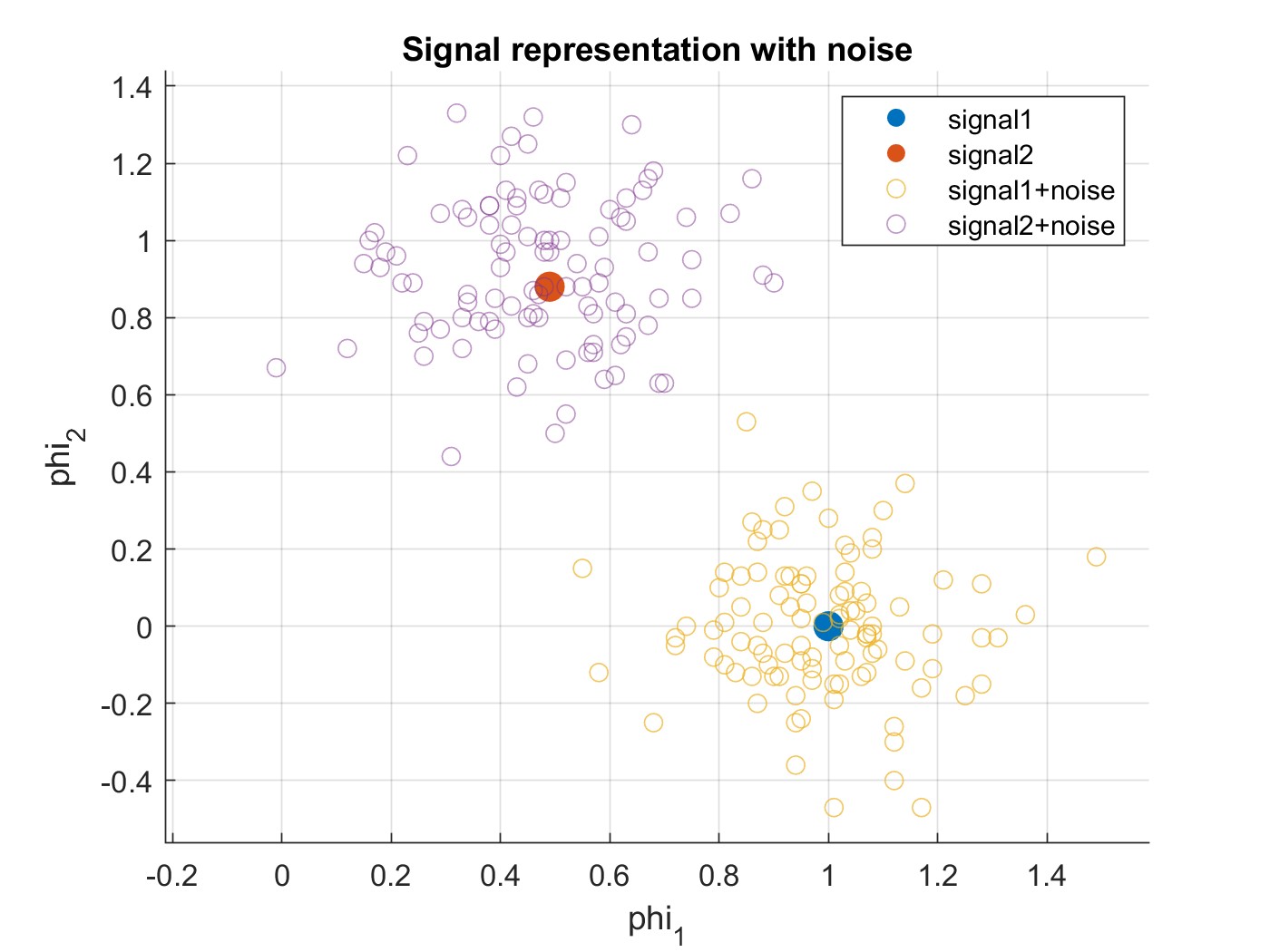


Figure Signal Space representation of signals s1,s2 with E/σ¬2 =-5dB

## 1.4 Noise Effect on Signal Space

With the increase of the variance of the noise, the resultant signal becomes more noisy so it becomes near to the decision boundary between the 2 signals and accordingly can be classified wrongly so we can notice that at snr=10db, the resultant signals are more compact near the ideal signal projection while at snr=-5, the resultant signals are more spread around the ideal space and near the decision boundary between the symbols.

# Appendix A: Codes for Part One:

## A.1 Code for Gram-Schmidt Orthogonalization

|  |
| --- |
| %-------------------------------------  % Requirment 1.1  %-------------------------------------  % this function is used to find the basis of 2 signals  function [phi1,phi2] = GM\_Bases(s1,s2)    % get the energy of the 1st signal  E1 = sum(abs(s1).^2) / 100;    % calculate phi1  phi1 = s1 ./ sqrt(E1);  % calculate S21  s21 = sum(phi1 .\* s2) / 100;  % get intermediate variable called g2  g2 = s2 - s21 .\* phi1;  % get the energy of g2  Eg2 = sum(abs(g2).^2) / 100;  % calculate phi2  phi2 = g2 ./ sqrt(Eg2);  % if the 2 basis are the same then make the 2nd basis 0  if phi2(:) == phi1(:)  phi2(:) = 0;  end  end |

## A.2 Code for Signal Space representation

|  |
| --- |
| %-------------------------------------  % Requirment 1.2  %-------------------------------------  % this function is used to get the signal space of s in terms of phi1 and  % phi2  function [v1,v2] = signal\_space(s, phi1,phi2)  % correlate the signal with 1st basis  v1 = sum(phi1 .\* s) / 100;    % correlate the signal with 2nd basis  v2 = sum(phi2 .\* s) / 100;  % round the values to the nearest 2 digits  v1 = round(v1, 2);  v2 = round(v2, 2);    end |

## A.3 Code for plotting the bases functions

|  |
| --- |
| % construct our signals  T = 0:0.01:1;  s1 = ones(1, 101);  s2 = ones(1, 101);  s2(76:end) = -1;  % get the bases of these 2 signals and plot them  [phi1, phi2] = GM\_Bases(s1, s2);  temp = sum(dot(phi2, phi1));  % plot the first basis function  figure;  hold on  title("first basis function");  xlabel('time (seconds)');  ylabel('Amplitude');  plot(T, phi1, 'DisplayName', 'phi\_1', 'LineWidth', 2, 'Color', 'blue');  grid on  legend;  hold off  % plot the second basis function  figure;  hold on  title("second basis function");  xlabel('time (seconds)');  ylabel('Amplitude');  plot(T, phi2, 'DisplayName', 'phi\_2', 'LineWidth', 2, 'Color', 'red');  grid on  legend;  hold off |

## A.4 Code for plotting the Signal space Representations

|  |
| --- |
| % get the signal projection in the domain of phi1 and phi2  [S1\_Projection\_Phi1, S1\_Projection\_Phi2]= signal\_space(s1, phi1, phi2);  [S2\_Projection\_Phi1, S2\_Projection\_Phi2]= signal\_space(s2, phi1, phi2);  % plot the projections of the signals in the axis of phi1 and phi2  figure;  hold on  title("Signal representation");  xlabel('phi\_1');  ylabel('phi\_2');  plot([0, S1\_Projection\_Phi1], [0, S1\_Projection\_Phi2], '-o', 'DisplayName', 'signal1', 'LineWidth', 2, 'Color', 'blue');  plot([0, S2\_Projection\_Phi1], [0, S2\_Projection\_Phi2], '-o', 'DisplayName', 'signal2', 'LineWidth', 2, 'Color', 'red');  grid on  legend;  hold off |

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

|  |
| --- |
| % scatter when snr = -5 db  plotNoisyProjectionSignal(phi1, phi2, s1, s2, S1\_Projection\_Phi1, S1\_Projection\_Phi2, S2\_Projection\_Phi1, S2\_Projection\_Phi2, 101, -5);  % scatter when snr = 0 db  plotNoisyProjectionSignal(phi1, phi2, s1, s2, S1\_Projection\_Phi1, S1\_Projection\_Phi2, S2\_Projection\_Phi1, S2\_Projection\_Phi2, 101, 0);  % scatter when snr = 10 db  plotNoisyProjectionSignal(phi1, phi2, s1, s2, S1\_Projection\_Phi1, S1\_Projection\_Phi2, S2\_Projection\_Phi1, S2\_Projection\_Phi2, 101, 10);  % this is a utility function to plot the actual signal projection and noisy  % signal projection  function [] = plotNoisyProjectionSignal(phi1, phi2, s1, s2, S1\_Projection\_Phi1, S1\_Projection\_Phi2, S2\_Projection\_Phi1, S2\_Projection\_Phi2, lenOfSample, snr)  % 2 variables will hold the projection of signal in the domain of phi1,  % phi2  Symbol1\_NoiseSamples\_XVal = zeros(1, lenOfSample);  Symbol1\_NoiseSamples\_YVal = zeros(1, lenOfSample);  Symbol2\_NoiseSamples\_XVal = zeros(1, lenOfSample);  Symbol2\_NoiseSamples\_YVal = zeros(1, lenOfSample);  % generate 100 sample with added noise where snr = -5db and plot it  for i = 1:lenOfSample    % generate 2 signals with noise  symbol1 = addNoise(s1, snr);  symbol2 = addNoise(s2, snr);  % get the projections of these signals in the basis domain  [symbol1\_projectionPhi1, symbol1\_projectionPhi2] = signal\_space(symbol1, phi1, phi2);  [symbol2\_projectionPhi1, symbol2\_projectionPhi2] = signal\_space(symbol2, phi1, phi2);    % add the point to the array  Symbol1\_NoiseSamples\_XVal(i) = symbol1\_projectionPhi1;  Symbol1\_NoiseSamples\_YVal(i) = symbol1\_projectionPhi2;  Symbol2\_NoiseSamples\_XVal(i) = symbol2\_projectionPhi1;  Symbol2\_NoiseSamples\_YVal(i) = symbol2\_projectionPhi2;    end  % plot the projections of the signals in the axis of phi1 and phi2  figure;  hold on  title("Signal representation with noise");  xlabel('phi\_1');  ylabel('phi\_2');  scatter(S1\_Projection\_Phi1, S1\_Projection\_Phi2, 100, 'filled', 'DisplayName', 'signal1');  scatter(S2\_Projection\_Phi1, S2\_Projection\_Phi2, 100, 'filled', 'DisplayName', 'signal2');  scatter(Symbol1\_NoiseSamples\_XVal, Symbol1\_NoiseSamples\_YVal, 'DisplayName', 'signal1+noise', 'MarkerEdgeAlpha', 0.7);  scatter(Symbol2\_NoiseSamples\_XVal, Symbol2\_NoiseSamples\_YVal, 'DisplayName', 'signal2+noise', 'MarkerEdgeAlpha', 0.5);  grid on  legend;  hold off  end  %-------------------------------------  % Requirment 1.3  %-------------------------------------  % this function is used to generate gaussian noise and add it to the signal  function [r\_t] = addNoise(signal, snr)    % generate the AWGN noise given the snr (since awgn function is only  % available on communication toolbox and I don't have on my matlab  % student version so I implemented awgn noise function)  r\_t = awgn(signal, snr);  end  % this is the implemented awgn function and I implemented it as awgn  % function is only available on communcation toolbox that I don't have it  % on my matlab student licenece, the implementation of this function is  % taken from this link:  % https://www.gaussianwaves.com/gaussianwaves/wp-content/uploads/2015/06/How\_to\_generate\_AWGN\_noise.pdf  function [resultSignal] = awgn(signal, snr\_db)    % get the length of the signal  L = length(signal);  % convert the SNR from log scale to linear scale  snr = 10 ^ (snr\_db / 10);  % calculate the energy of the symbol  E = sum(abs(signal) .^ 2) / L;  % get the N0 of the symbol  N0 = E / snr;  % get the the actual noise  noise = sqrt(N0) \* randn(1, L);    % return the result  resultSignal = signal + noise;  end |

## A.6 whole code in one file

|  |
| --- |
| %-------------------------------------  % Requirment 1.4  %-------------------------------------  % construct our signals  T = 0:0.01:1;  s1 = ones(1, 101);  s2 = ones(1, 101);  s2(76:end) = -1;  % get the bases of these 2 signals and plot them  [phi1, phi2] = GM\_Bases(s1, s2);  temp = sum(dot(phi2, phi1));  % plot the first basis function  figure;  hold on  title("first basis function");  xlabel('time (seconds)');  ylabel('Amplitude');  plot(T, phi1, 'DisplayName', 'phi\_1', 'LineWidth', 2, 'Color', 'blue');  grid on  legend;  hold off  % plot the second basis function  figure;  hold on  title("second basis function");  xlabel('time (seconds)');  ylabel('Amplitude');  plot(T, phi2, 'DisplayName', 'phi\_2', 'LineWidth', 2, 'Color', 'red');  grid on  legend;  hold off  % get the signal projection in the domain of phi1 and phi2  [S1\_Projection\_Phi1, S1\_Projection\_Phi2]= signal\_space(s1, phi1, phi2);  [S2\_Projection\_Phi1, S2\_Projection\_Phi2]= signal\_space(s2, phi1, phi2);  % plot the projections of the signals in the axis of phi1 and phi2  figure;  hold on  title("Signal representation");  xlabel('phi\_1');  ylabel('phi\_2');  plot([0, S1\_Projection\_Phi1], [0, S1\_Projection\_Phi2], '-o', 'DisplayName', 'signal1', 'LineWidth', 2, 'Color', 'blue');  plot([0, S2\_Projection\_Phi1], [0, S2\_Projection\_Phi2], '-o', 'DisplayName', 'signal2', 'LineWidth', 2, 'Color', 'red');  grid on  legend;  hold off  % scatter when snr = -5 db  plotNoisyProjectionSignal(phi1, phi2, s1, s2, S1\_Projection\_Phi1, S1\_Projection\_Phi2, S2\_Projection\_Phi1, S2\_Projection\_Phi2, 101, -5);  % scatter when snr = 0 db  plotNoisyProjectionSignal(phi1, phi2, s1, s2, S1\_Projection\_Phi1, S1\_Projection\_Phi2, S2\_Projection\_Phi1, S2\_Projection\_Phi2, 101, 0);  % scatter when snr = 10 db  plotNoisyProjectionSignal(phi1, phi2, s1, s2, S1\_Projection\_Phi1, S1\_Projection\_Phi2, S2\_Projection\_Phi1, S2\_Projection\_Phi2, 101, 10);  % this is a utility function to plot the actual signal projection and noisy  % signal projection  function [] = plotNoisyProjectionSignal(phi1, phi2, s1, s2, S1\_Projection\_Phi1, S1\_Projection\_Phi2, S2\_Projection\_Phi1, S2\_Projection\_Phi2, lenOfSample, snr)  % 2 variables will hold the projection of signal in the domain of phi1,  % phi2  Symbol1\_NoiseSamples\_XVal = zeros(1, lenOfSample);  Symbol1\_NoiseSamples\_YVal = zeros(1, lenOfSample);  Symbol2\_NoiseSamples\_XVal = zeros(1, lenOfSample);  Symbol2\_NoiseSamples\_YVal = zeros(1, lenOfSample);  % generate 100 sample with added noise where snr = -5db and plot it  for i = 1:lenOfSample    % generate 2 signals with noise  symbol1 = addNoise(s1, snr);  symbol2 = addNoise(s2, snr);  % get the projections of these signals in the basis domain  [symbol1\_projectionPhi1, symbol1\_projectionPhi2] = signal\_space(symbol1, phi1, phi2);  [symbol2\_projectionPhi1, symbol2\_projectionPhi2] = signal\_space(symbol2, phi1, phi2);    % add the point to the array  Symbol1\_NoiseSamples\_XVal(i) = symbol1\_projectionPhi1;  Symbol1\_NoiseSamples\_YVal(i) = symbol1\_projectionPhi2;  Symbol2\_NoiseSamples\_XVal(i) = symbol2\_projectionPhi1;  Symbol2\_NoiseSamples\_YVal(i) = symbol2\_projectionPhi2;    end  % plot the projections of the signals in the axis of phi1 and phi2  figure;  hold on  title("Signal representation with noise");  xlabel('phi\_1');  ylabel('phi\_2');  scatter(S1\_Projection\_Phi1, S1\_Projection\_Phi2, 100, 'filled', 'DisplayName', 'signal1');  scatter(S2\_Projection\_Phi1, S2\_Projection\_Phi2, 100, 'filled', 'DisplayName', 'signal2');  scatter(Symbol1\_NoiseSamples\_XVal, Symbol1\_NoiseSamples\_YVal, 'DisplayName', 'signal1+noise', 'MarkerEdgeAlpha', 0.7);  scatter(Symbol2\_NoiseSamples\_XVal, Symbol2\_NoiseSamples\_YVal, 'DisplayName', 'signal2+noise', 'MarkerEdgeAlpha', 0.5);  grid on  legend;  hold off  end  %-------------------------------------  % Requirment 1.3  %-------------------------------------  % this function is used to generate gaussian noise and add it to the signal  function [r\_t] = addNoise(signal, snr)    % generate the AWGN noise given the snr (since awgn function is only  % available on communication toolbox and I don't have on my matlab  % student version so I implemented awgn noise function)  r\_t = awgn(signal, snr);  end  % this is the implemented awgn function and I implemented it as awgn  % function is only available on communcation toolbox that I don't have it  % on my matlab student licenece, the implementation of this function is  % taken from this link:  % https://www.gaussianwaves.com/gaussianwaves/wp-content/uploads/2015/06/How\_to\_generate\_AWGN\_noise.pdf  function [resultSignal] = awgn(signal, snr\_db)    % get the length of the signal  L = length(signal);  % convert the SNR from log scale to linear scale  snr = 10 ^ (snr\_db / 10);  % calculate the energy of the symbol  E = sum(abs(signal) .^ 2) / L;  % get the N0 of the symbol  N0 = E / snr;  % get the the actual noise  noise = sqrt(N0) \* randn(1, L);    % return the result  resultSignal = signal + noise;  end  %-------------------------------------  % Requirment 1.1  %-------------------------------------  % this function is used to find the basis of 2 signals  function [phi1,phi2] = GM\_Bases(s1,s2)    % get the energy of the 1st signal  E1 = sum(abs(s1).^2) / 100;    % calculate phi1  phi1 = s1 ./ sqrt(E1);  % calculate S21  s21 = sum(phi1 .\* s2) / 100;  % get intermediate variable called g2  g2 = s2 - s21 .\* phi1;  % get the energy of g2  Eg2 = sum(abs(g2).^2) / 100;  % calculate phi2  phi2 = g2 ./ sqrt(Eg2);  % if the 2 basis are the same then make the 2nd basis 0  if phi2(:) == phi1(:)  phi2(:) = 0;  end  end  %-------------------------------------  % Requirment 1.2  %-------------------------------------  % this function is used to get the signal space of s in terms of phi1 and  % phi2  function [v1,v2] = signal\_space(s, phi1,phi2)  % correlate the signal with 1st basis  v1 = sum(phi1 .\* s) / 100;    % correlate the signal with 2nd basis  v2 = sum(phi2 .\* s) / 100;  % round the values to the nearest 2 digits  v1 = round(v1, 2);  v2 = round(v2, 2);    end |