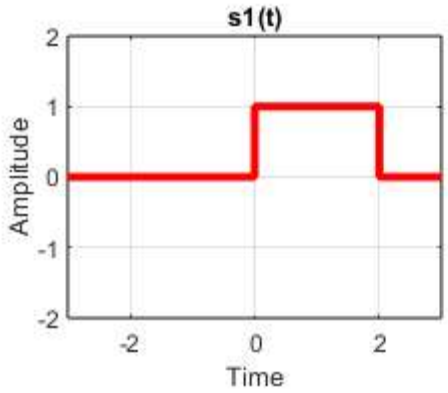
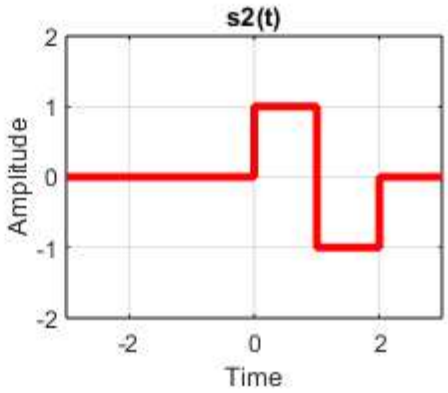
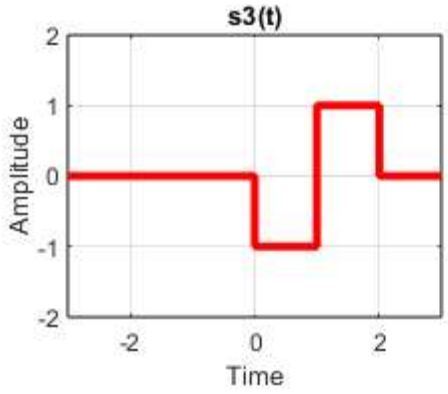
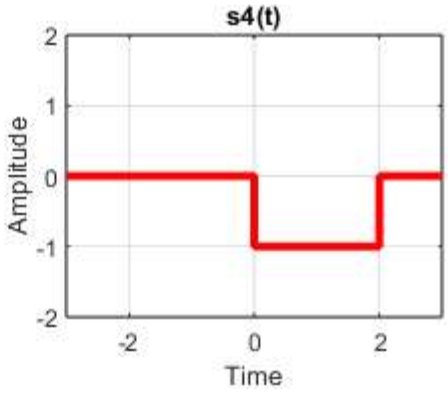
```
% Grahm-Schmidt Orthogonalization-1
% Name: Sachin Chauhan
% Roll# 23SP06007
% Course: Advanced Communication Laboratory
% ////////////////////////////////////////////////
clc;
clear all;
close all;
% ////////////////////////////////////////////

syms x;
%defining time limits
t_low=0;
t_high=3;
%defining signals
s1=heaviside(x)-heaviside(x-2);
s2=heaviside(x)-2*heaviside(x-1)+heaviside(x-2);
s3=(-1)*s2;
s4=(-1)*s1;
phi=[];%basis fucntion array
%calculating basis function
s=[s1,s2,s3,s4];
syms f_tilda;
for i=1:length(s)
    f_tilda=s(i);
    if i>1
        for j=1:i-1%calculating f_tilda
            f_tilda=f_tilda-innerProduct(phi(j),f_tilda,t_low,t_high)*phi(j);
        end
    end
    % if(isnan(f_tilda)==1)
        phi=[phi f_tilda/(calc_norm(f_tilda,t_low,t_high))];
    % end
end
%storing coefficients in an array for plotting the constellation diagram
for i=1:length(s)
    for j=1:length(phi)
        if(isnan(phi(j))==0)
            points(i,j)=innerProduct(phi(j),s(i),t_low,t_high);
        end
    end
end
% points
%plotting signals
figure(1);
for i=1:length(s)
    subplot(2,2,i);
    fplot(s(i),'r',LineWidth=3);
    xlim([-3 3]);
    ylim([-2 2]);
    grid on;
    xlabel('Time');
    ylabel('Amplitude');
    title(sprintf('s%d(t)',i));

end
```

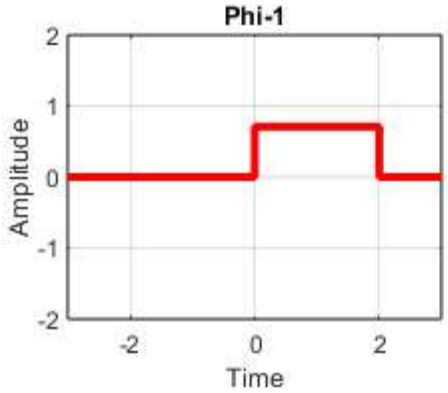
```
%plotting the basis functions
figure(2);
for i=1:length(phi)
    hold on;
    if(isnan(phi(i))==0)
        subplot(2,2,i);
        fplot(phi(i),'r',LineWidth=3);
        grid on;
        xlim([-3 3]);
        ylim([-2 2]);
        xlabel('Time');
        ylabel('Amplitude');
        title(sprintf('Phi-%d',i));
    end
end
%plotting the constellation diagram
[row col]=size(points);
x=points(:,1);
y=points(:,2);
figure(3);
scatter(x,y,'r','filled');
xlabel('Phi1');
ylabel('Phi2');
title('Constellation Plot');
grid on;

%user defined function for calculating norm
function norm=calc_norm(s,t_low,t_high)
    norm=sqrt(int(s*s,t_low,t_high));
end
%user defined function for calculating innerproduct of two signals
function inner_prod=innerProduct(f1,f2,t_low,t_high)
    inner_prod=int(f1*f2,t_low,t_high);
end
```

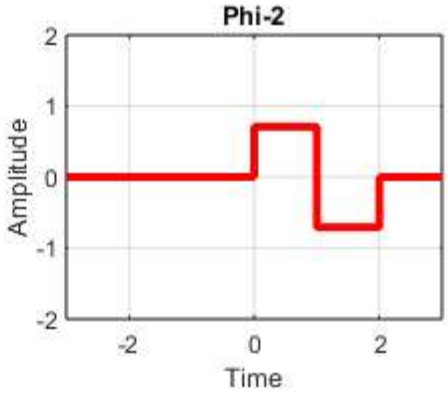

s1(t)
Amplitude
Time
2
1
0
-1
-2
-2
0
2

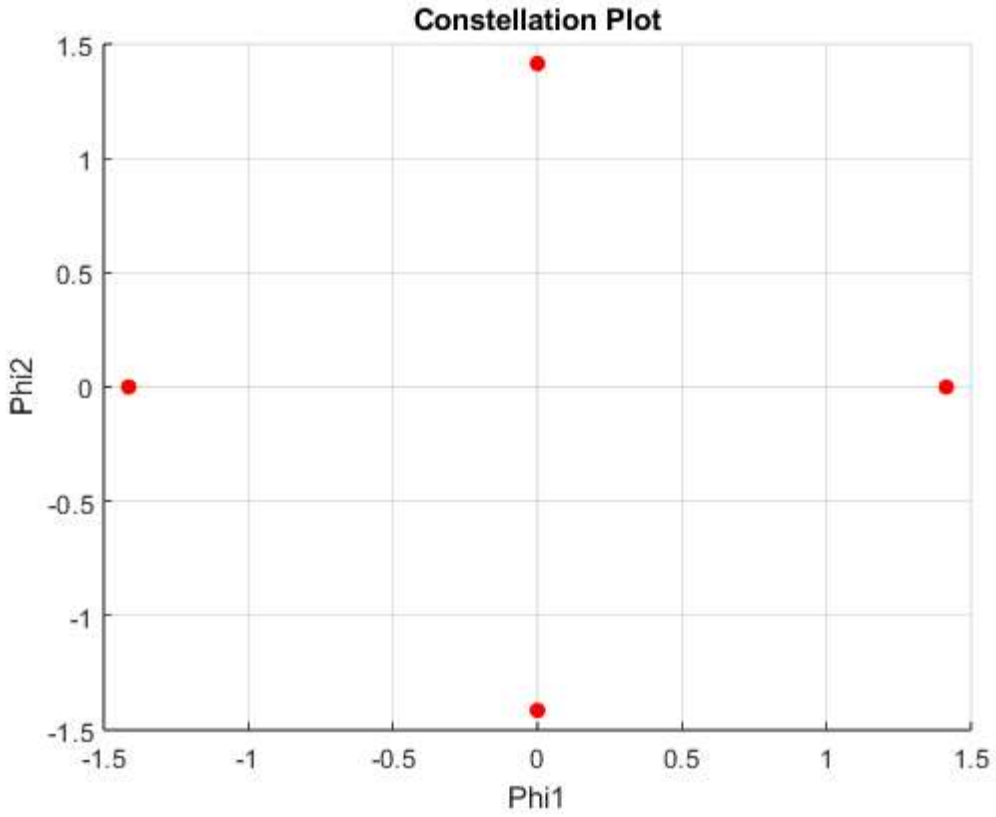

s2(t)
Amplitude
Time
2
1
0
-1
-2
-2
0
2


s3(t)
Amplitude
Time
2
1
0
-1
-2
-2
0
2


s4(t)
Amplitude
Time
2
1
0
-1
-2
-2
0
2


Phi-1
Amplitude
Time
2
1
0
-1
-2
-2
0
2

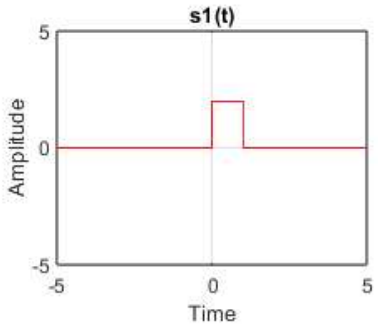

Phi-2
Amplitude
Time
2
1
0
-1
-2
-2
0
2

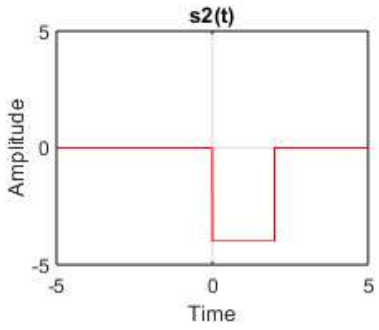


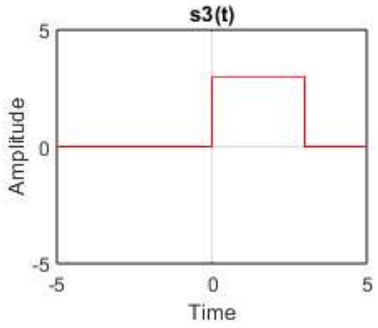
Published with MATLAB® R2023a

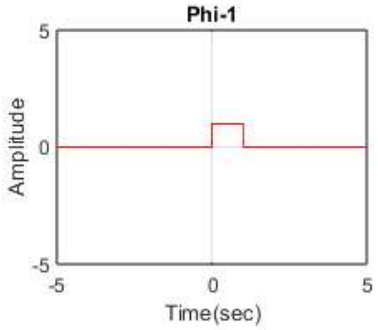
```
% Grahm-Schmidt Orthogonalization-2
% Name: Sachin Chauhan
% Roll# 23SP06007
% Course: Advanced Communication Laboratory
% ////////////////////////////////////////////////
clc;
clear global;
close all;
% ////////////////////////////////////////////

syms x;
%defining time limits
t_low=0;
t_high=3;
%defining signals
s1=2*(heaviside(x)-heaviside(x-1));
s2=4*(-heaviside(x)+heaviside(x-2));
s3=3*(heaviside(x)-heaviside(x-3));
phi=[];%basis fucntion array
s=[s1,s2,s3];%signal array
%calculating basis function
for i=1:length(s)
    f_tilda=s(i);
    if i>1
        for j=1:i-1%calculating f_tilda
            f_tilda=f_tilda-innerProduct(phi(j),f_tilda,t_low,t_high)*phi(j);
        end
    end
    % if(f_tilda~=0)
        phi=[phi f_tilda/(calc_norm(f_tilda,t_low,t_high))];%calulating phi(t) for the ith signal and appending it to the phi array
    % end
end

%storing coefficients in an array for plotting the constellation diagram
for i=1:length(s)
    for j=1:length(phi)
       if(isnan(phi(j))==0)
            points(i,j)=innerProduct(phi(j),s(i),t_low,t_high);
        end
    end
end

%plotting signals
figure(1);
for i=1:length(s)
    subplot(2,2,i);
    fplot(s(i),'r');
    xlim([-5 5]);
    ylim([-5 5]);
    grid on;
    xlabel('Time');
    ylabel('Amplitude');
    title(sprintf('s%d(t)',i));

end
%plotting the basis functions
figure(2);
for i=1:length(phi)
    hold on;
    subplot(2,2,i);
    % fplot(f_tilda_arr(i));
    fplot(phi(i),'r');
    grid on;
    xlim([-5 5]);
    ylim([-5 5]);
    xlabel('Time(sec)');
    ylabel('Amplitude');
    title(sprintf('Phi-%d',i));
```

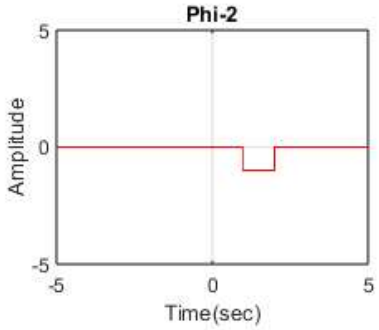
```
end
%plotting the constellation diagram
[row col]=size(points);
x=points(:,1);
y=points(:,2);
z=points(:,3);
figure(3);
scatter3(x,y,z,'r','filled');
xlabel('Phi1');
ylabel('Phi2');
zlabel('Phi3');
title('Constellation Plot');
grid on;

%user defined function for calculating norm
function norm=calc_norm(s,t_low,t_high)
    norm=sqrt(int(s*s,t_low,t_high));
end
%user defined function for calculating innerproduct of two signals
function inner_prod=innerProduct(f1,f2,t_low,t_high)
    inner_prod=int(f1*f2,t_low,t_high);
end
```

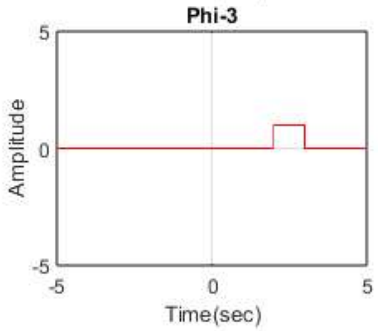

s1(t)
Amplitude
Time
5
0
-5
-5
0
5

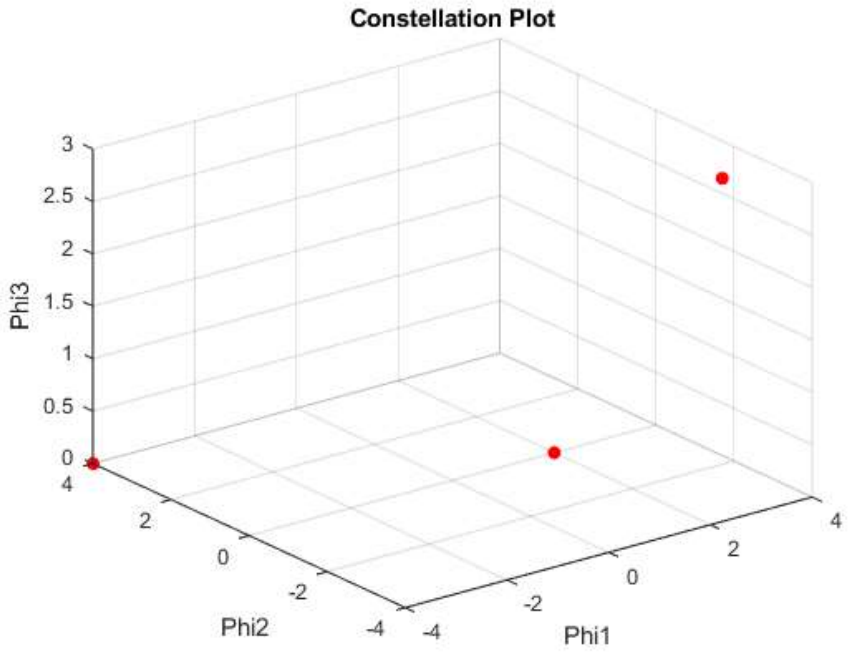

s2(t)
Amplitude
Time
5
0
-5
-5
0
5


s3(t)
Amplitude
Time
5
0
-5
-5
0
5


Phi-1
Amplitude
Time(sec)
5
0
-5
-5
0
5


Phi-2
Amplitude
Time(sec)
5
0
-5
-5
0
5

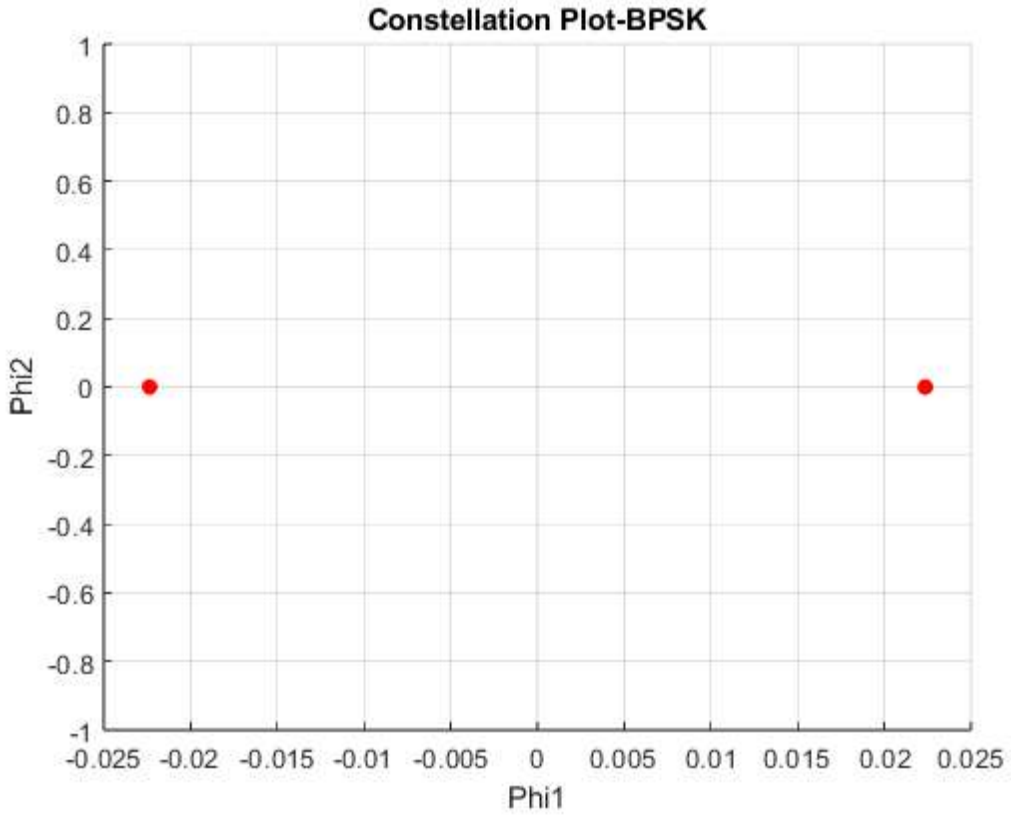

Phi-3
Amplitude
Time(sec)
5
0
-5
-5
0
5

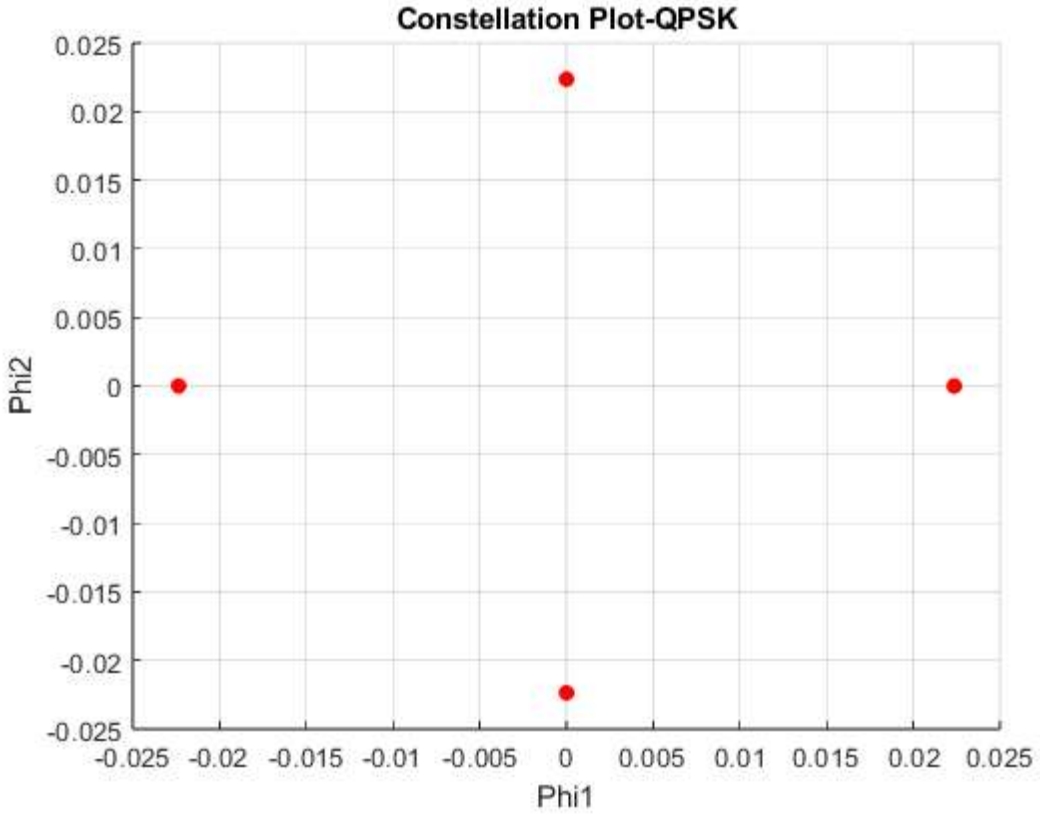


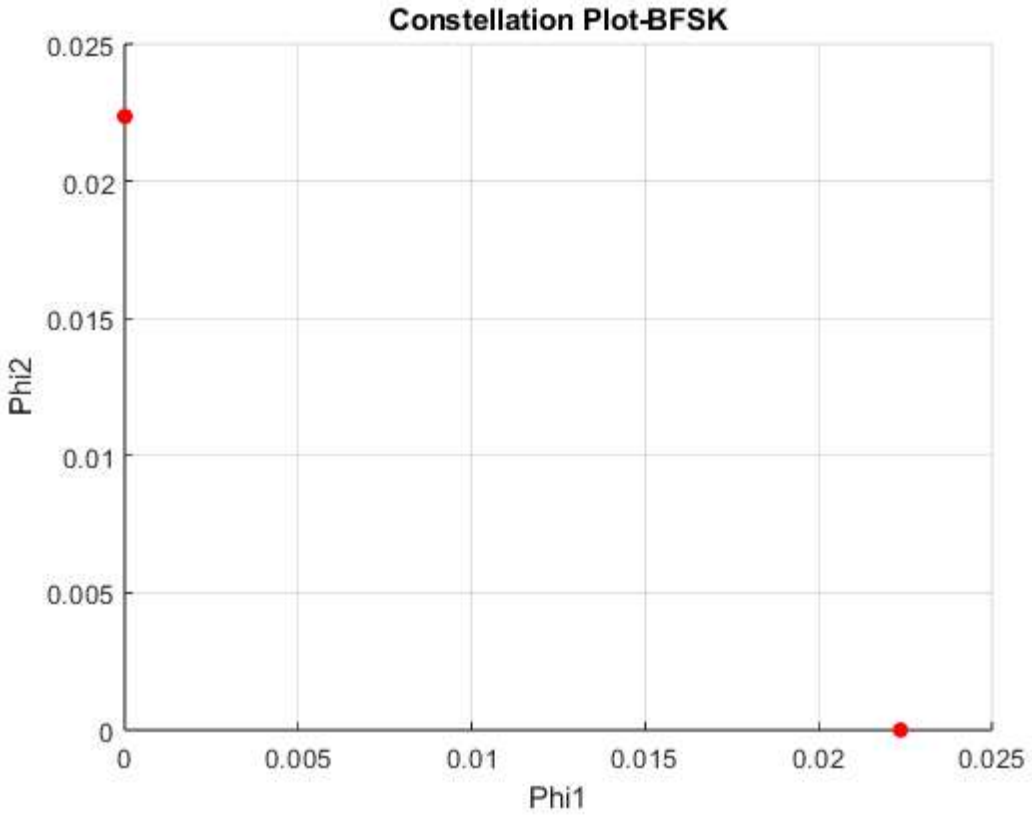
Published with MATLAB® R2023a

```
% Q3
% ------------
clc;
clear all;
close all;
%-------------
syms t;
x=[];y=[];
fc=5000;
Ts=0.001;
A=1;
M1=2;
M2=4;

% Random 4-bit sequence generation
bit_seq=[];
for i=1:4
    if rand < 0.5
         bit_seq=[bit_seq 0];
    else
         bit_seq=[bit_seq 1];
    end
end

for i=1:M1
    x=[x ((A*sqrt(Ts/2)*cos(2*pi*(i-1)/M1)))];
    y=[y ((A*sqrt(Ts/2))*round(sin(2*pi*(i-1)/M1)))];
end
figure(1);
scatter(x,y,'r','filled');
xlabel('Phi1');
ylabel('Phi2');
title('Constellation Plot-BPSK');
grid on;
x=[];y=[];

%QPSK
for i=1:M2
    x=[x ((A*sqrt(Ts/2)*(cos(2*pi*(i-1)/M2))))];
    y=[y ((A*sqrt(Ts/2))*round(sin(2*pi*(i-1)/M2)))];
end
figure(2);
scatter(x,y,'r','filled');
xlabel('Phi1');
ylabel('Phi2');
title('Constellation Plot-QPSK');
grid on;
x=[];y=[];

%BFSK
f_high=fc+(fc/4);
f_low=fc-(fc/4);
s1=cos(2*pi*f_high*t);
s2=cos(2*pi*f_low*t);

x=zeros(1:2);
```

```
y=zeros(1:2);

x(1)=calc_norm(s1,0,Ts);
y(2)=calc_norm(s1,0,Ts);
figure(3);
scatter(x,y,'r','filled');
xlabel('Phi1');
ylabel('Phi2');
title('Constellation Plot-BFSK');
grid on;

%user defined function for calculating norm
function norm=calc_norm(s,t_low,t_high)
    norm=sqrt(int(s*s,t_low,t_high));
end
```


Constellation Plot-BPSK
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
Phi2
-0.025 -0.02 -0.015 -0.01 -0.005 0 0.005 0.01 0.015 0.02 0.025
Phi1


Constellation Plot-QPSK
0.025
0.02
0.015
0.01
0.005
0
-0.005
-0.01
-0.015
-0.02
-0.025
Phi2
-0.025 -0.02 -0.015 -0.01 -0.005 0 0.005 0.01 0.015 0.02 0.025
Phi1



Published with MATLAB® R2023a

```
% Grahm-Schmidt Orthogonalization-4
% Name: Sachin Chauhan
% Roll# 23SP06007
% Course: Advanced Communication Laboratory
%--------------------------------------------------
clc;
clear global;
close all;
%--------------------------------------------------

u1=[3,-2,-1]%inital basis vectors
u2=[-2,-1,3]%inital basis vectors
u=[u1;u2];
phi=[];%new basis vectors
phi(1,:)=u1/calc_norm(u1,u1);
f_tilda=u2-(innerProduct(phi(1,:),u2).*phi(1,:));
phi(2,:)=f_tilda/calc_norm(f_tilda,f_tilda);

w1=phi(1,:)
w2=phi(2,:)

%user defined function for calculating norm
function norm=calc_norm(s1,s2)
    % v=s1.*s2
    norm=sqrt(innerProduct(s1,s2));
end
%user defined function for calculating innerproduct of two signals
function inner_prod=innerProduct(v,u)
    inner_prod=2*((v(1)*u(1))+(v(2)*u(2))+(v(3)*u(3))) + (v(1)*u(2))+(v(2)*u(1))+(v(2)*u(3)) + (v(3)*u(2))+(v(3)*u(1))+(v(1)*u(3));
    % inner_prod
end
```

```
u1 =

     3    -2    -1


u2 =

    -2    -1     3


w1 =

    0.8018   -0.5345   -0.2673


w2 =

   -0.1543   -0.6172    0.7715
```

Published with MATLAB® R2023a