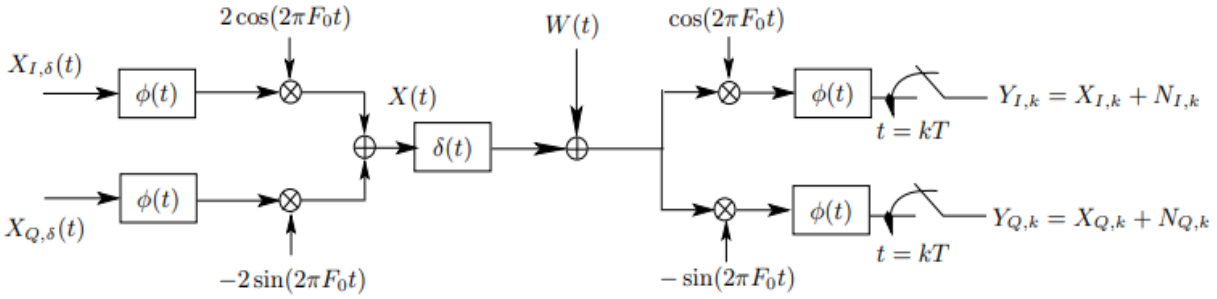
Πολυτεχνείο Κρήτης
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Τηλεπικοινωνιακά Συστήματα 1
[ΗΜΜΥ277]
3η Εργαστηριακή Άσκηση

Ομάδα 16
Αυγουστή Σαββίνα, 2018030200
Ιωαννίδης Χρήστος 2018030006

1ο Μέρος

Στην 3η εργαστηριακή άσκηση προσομοιώθηκε το τηλεπικοινωνιακό σύστημα του παρακάτω σχήματος, χρησιμοποιώντας διαμόρφωση 16-PSK και κατόπιν μελετήθηκε η απόδοση του.



Ερώτημα 1:

Για N=100 δημιουργήθηκε η δυαδική ακολουθία *bit_seq* με στοιχεία 4N ισοπίθανα bits.

Κώδικας Matlab:

```
%16-PSK
N=100; %number of symbols
bit_seq = (sign(randn(4*N, 1)) + 1)/2; %create random bits
X = bits_to_PSK_16(bit_seq); %turn bits into 2x100 gray coded coordinates
rot90(bit_seq);
%divide coordinates into x,y
real=X(1,:);
imag=X(2,:);
%PSK ASTERISM PLOT
figure(1)
plot(real,imag,'o')
```

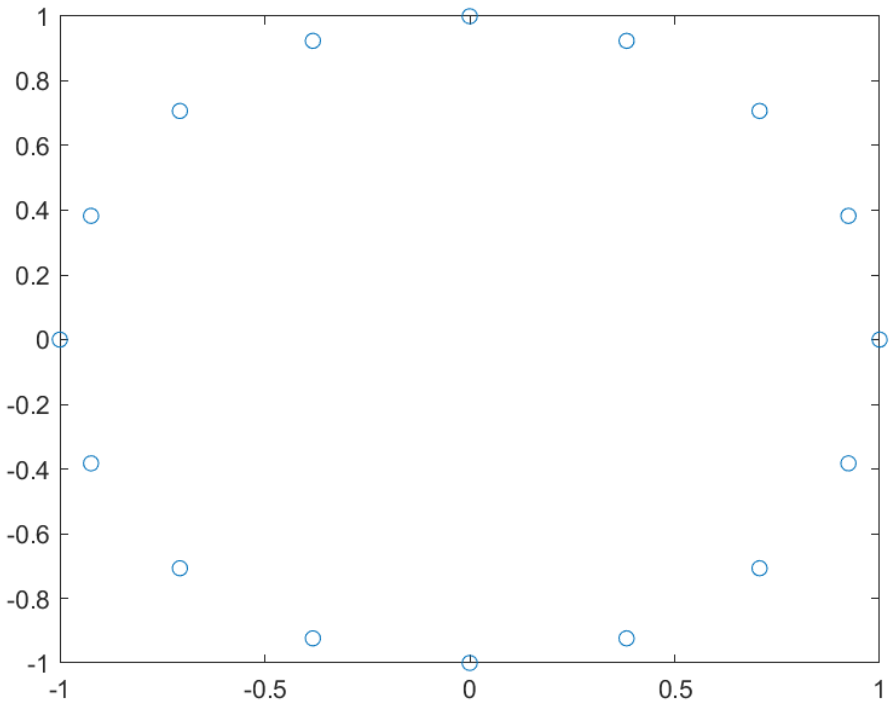
Ερώτημα 2:

Κατόπιν γράφτηκε η συνάρτηση *function X = bits_to_PSK(bit_seq)* η οποία χρησιμοποιεί κωδικοποίηση Gray και απεικονίζει τη δυαδική ακολουθία εισόδου *bit_seq* σε ακολουθία 16-PSK συμβόλων X, μήκους N, με στοιχεία τα δυσδιάστατα διανύσματα

$$Xn = \left[\frac{X_{I,n}}{X_{Q,n}}\right], \text{ για } n = 0,\ldots, N-1$$

Κώδικας Matlab:

```
function [ output ] = bits_to_PSK_16( input )

input=rot90(input);

gray_conversion_Table = [
0 0 0 0;
0 0 0 1;
0 0 1 1;
0 0 1 0;
0 1 1 0;
0 1 1 1;
0 1 0 1;
0 1 0 0;
1 1 0 0;
1 1 0 1;
1 1 1 1;
1 1 1 0;
1 0 1 0;
1 0 1 1;
1 0 0 1;
1 0 0 0
];

bin_symbols = reshape(input, 4, []).';

for i=1:length(input)/4
      bin_symbol=bin_symbols(i,:);

      dec_symbol=bi2de(bin_symbol,'left-msb');

      gray_bin_symbol = gray_conversion_Table(dec_symbol +1, :); % Convert
      binary to decimal and lookup corresponding Gray code

      gray_dec_symbol=bi2de(gray_bin_symbol,'left-msb');
      output(1,i)=cos((2*pi*gray_dec_symbol)/16);
      output(2,i)=sin((2*pi*gray_dec_symbol)/16);
end
end
```



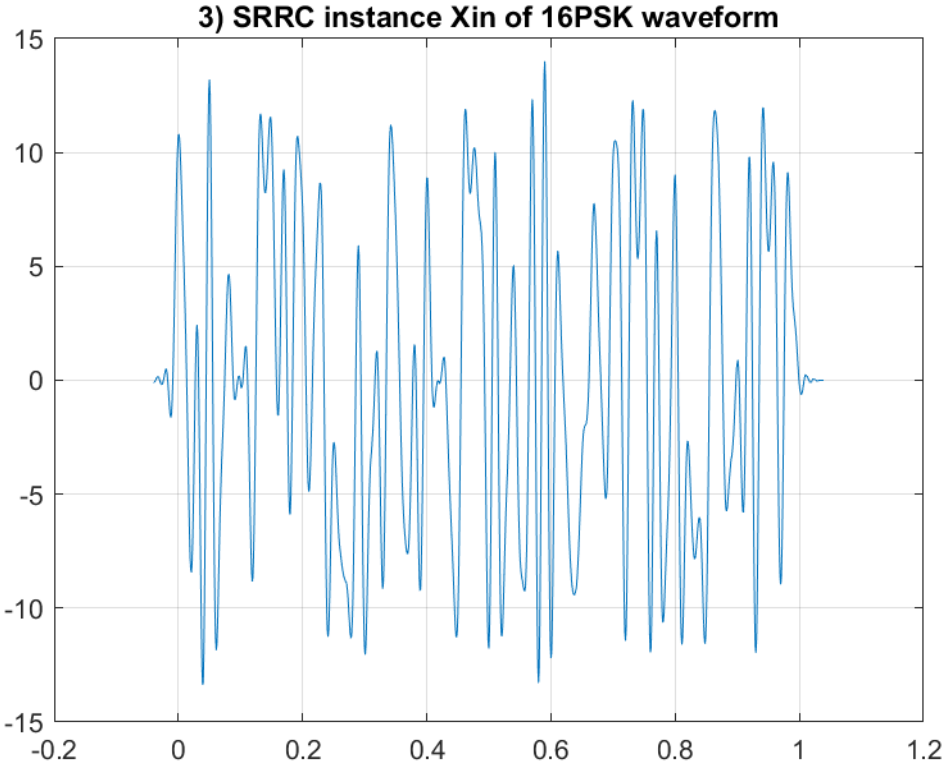
Ερώτημα 3:

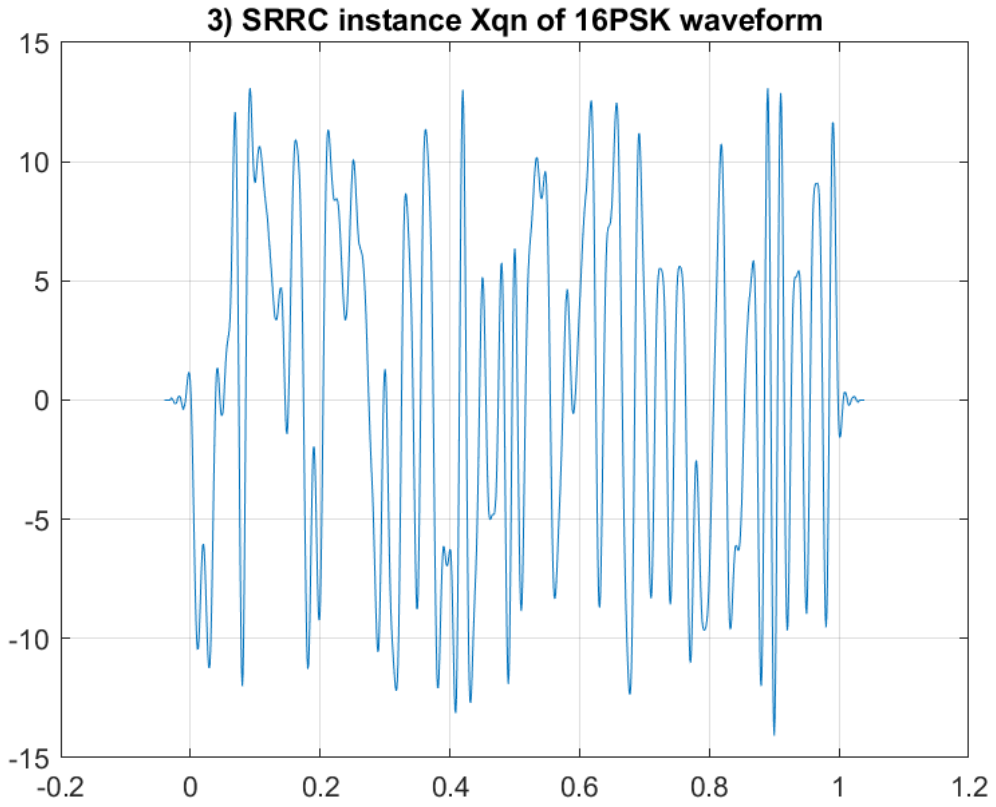
Περάστηκαν οι ακολουθίες $\{X_{i,n}\}$ και $\{X_{Q,n}\}$ από το SRRC φίλτρα μορφοποίησης και τέθηκαν $T = 10^{-2}$ sec, over = 10 και Ts = T/over. Έπειτα σχηματίστηκαν και σχεδιάστηκαν οι κυματομορφές εξόδου και δόθηκαν τα περιοδιαγράμματα τους.

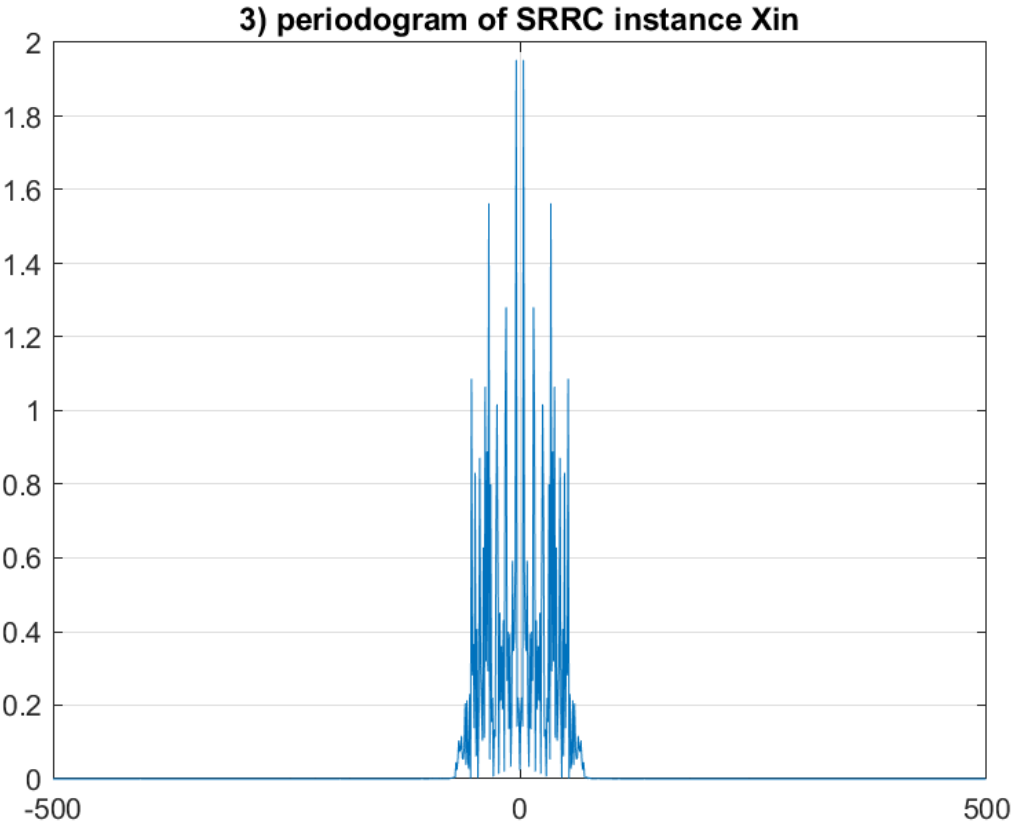
Κώδικας Matlab:

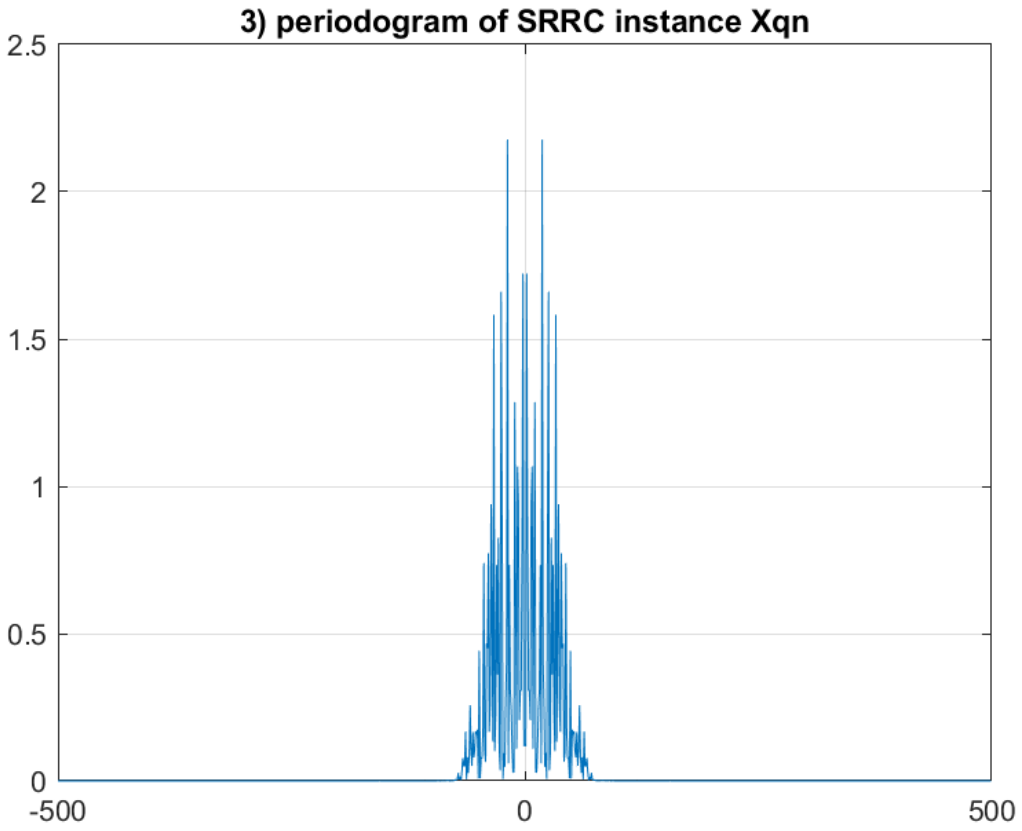
```
%3
%initial parameters
T=0.01;
over=10;
Ts=T/over;
A=4;
a=0.5;

%create SRRC pulse
[phi, t] = srrc_pulse(T, over, A, a);

%upsample
X_real_delta = 1/Ts * upsample(real, over);
X_imag_delta = 1/Ts * upsample(imag, over);

%define time
time = 0:Ts:N*Ts*over-Ts;
signal_t = [time(1)+t(1):Ts:time(end)+t(end)];

Xin_srrc = conv(X_real_delta,phi)*Ts; %convolute symbols waveform with SRRC
pulse
Xqn_srrc = conv(X_imag_delta,phi)*Ts; %convolute symbols waveform with SRRC
pulse
```

```
figure(2)
plot(signal_t,Xin_srrc);
grid on;
title('3) SRRC instance Xin of 16PSK waveform')
figure(3)
plot(signal_t,Xqn_srrc);
grid on;
title('3) SRRC instance Xqn of 16PSK waveform')

%periodogram plot
Nf=1024;
Fs = 1/Ts; % sampling frequency
freq = (-Fs/2:Fs/Nf:Fs/2-1/Nf); % zero-centered frequency range

%fft Xin SRRC
fftshift_SRRC_Xin = fftshift(fft(Xin_srrc,Nf)*Ts);
power_fftshift_SRRC_Xin = abs(fftshift_SRRC_Xin).^2; % zero-centered power
%fft Xin SRRC
fftshift_SRRC_Xqn = fftshift(fft(Xqn_srrc,Nf)*Ts);
power_fftshift_SRRC_Xqn = abs(fftshift_SRRC_Xqn).^2; % zero-centered power

figure(4)
plot(freq,power_fftshift_SRRC_Xin)
grid on;
title('3) periodogram of SRRC instance Xin')
figure(5)
plot(freq,power_fftshift_SRRC_Xqn)
grid on;
title('3) periodogram of SRRC instance Xqn')
```




3) SRRC instance Xqn of 16PSK waveform
15
10
5
0
-5
-10
-15
-0.2
0
0.2
0.4
0.6
0.8
1
1.2


3) periodogram of SRRC instance Xin
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-500
0
500



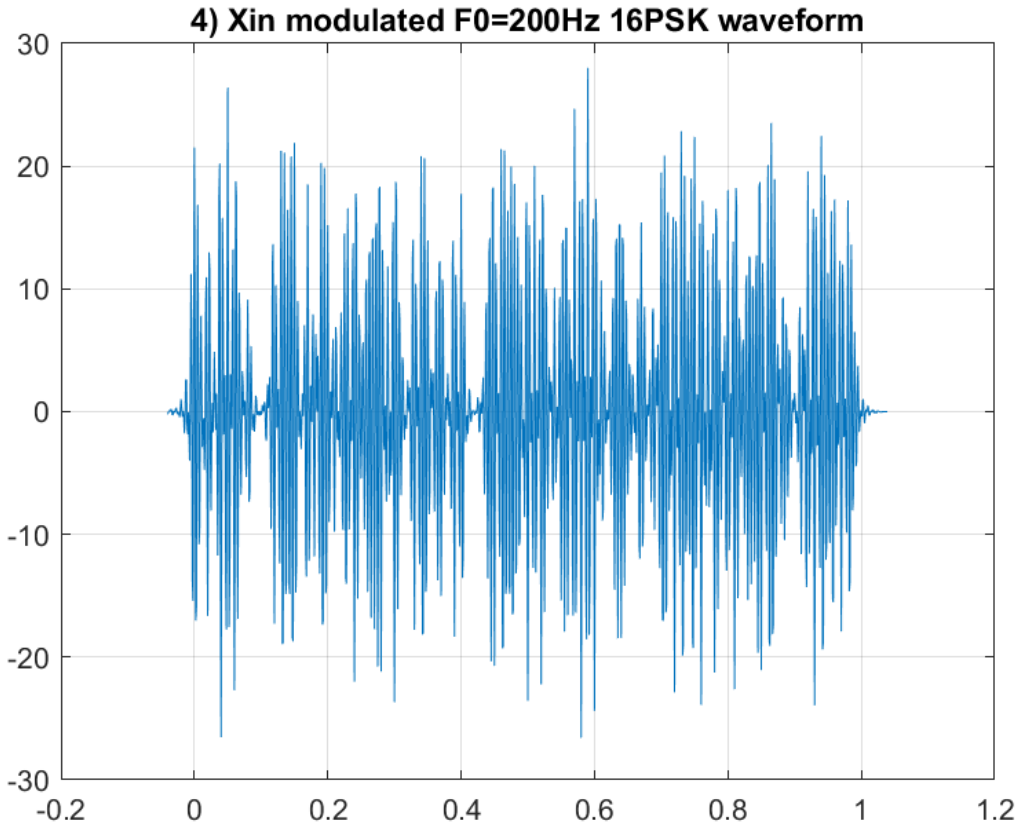
Ερώτημα 4:

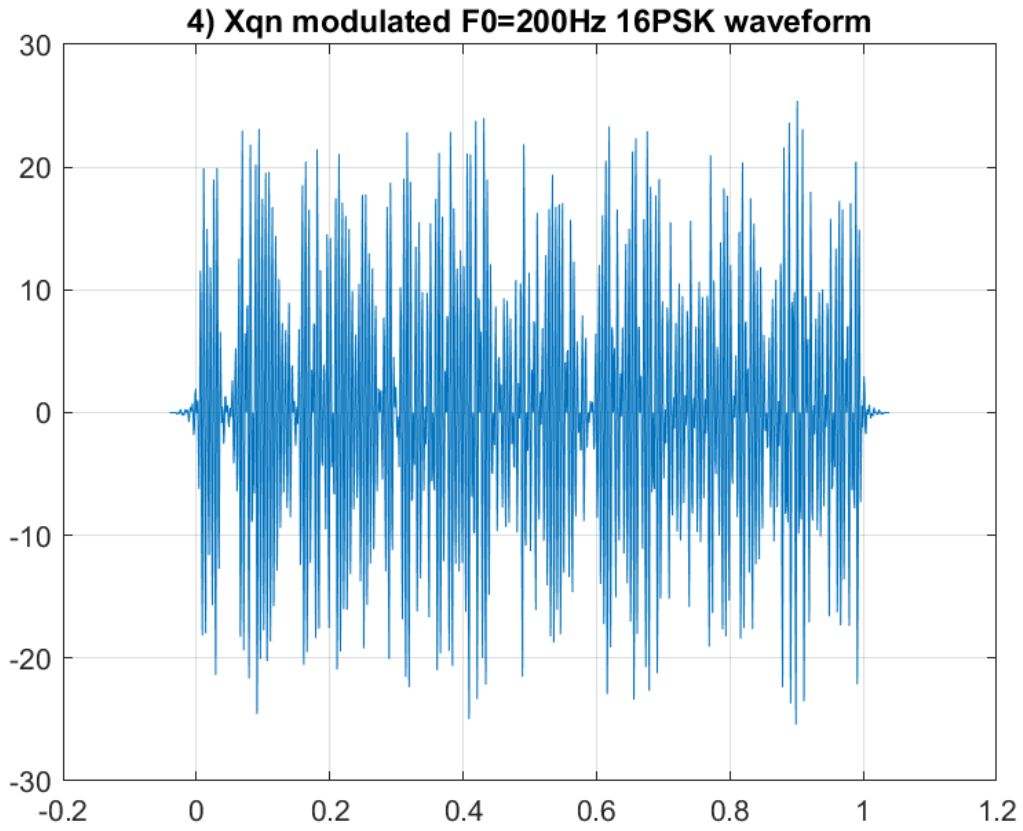
Πολλαπλασιάστηκαν οι έξοδοι των φίλτρων με φορείς συχνότητας Fo = 200Hz και σχεδιάστηκαν οι κυματομορφές $X_I(t)$ και $X_Q(t)$ και δόθηκαν τα αντίστοιχα περιοδιαγράμματα.

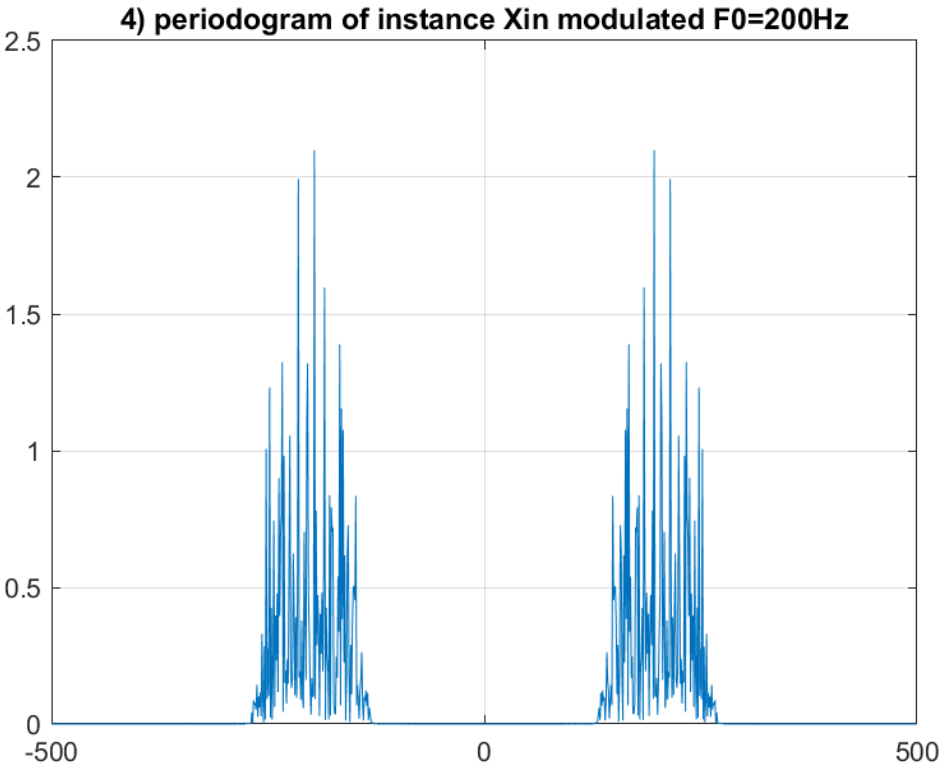
Κώδικας Matlab:

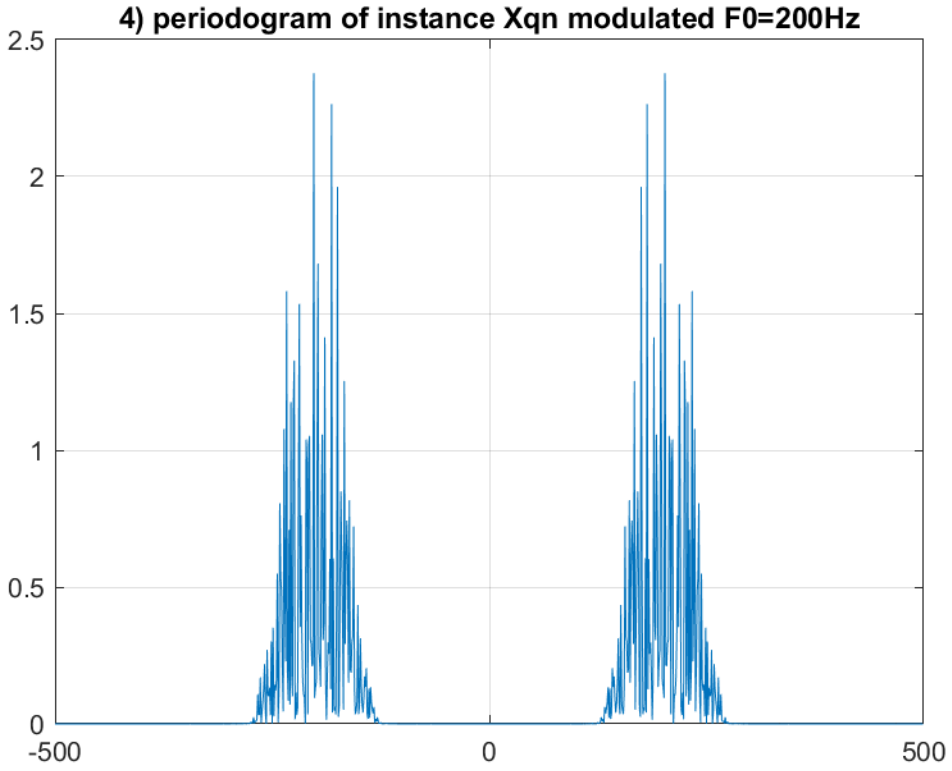
```
%4
%define carrier signals
F0=200 %Hz carrier freq
i_carrier=2*cos(2*pi*F0*signal_t);
q_carrier=-2*sin(2*pi*F0*signal_t);

%multiply main signals with the 2 orthogonal carrier signals in order to be
%sent in parralel
Xi=Xin_srrc.*i_carrier;
Xq=Xqn_srrc.*q_carrier;

figure(6)
plot(signal_t,Xi);
grid on;
title('4) Xin modulated F0=200Hz 16PSK waveform')
figure(7)
plot(signal_t,Xq);
grid on;
title('4) Xqn modulated F0=200Hz 16PSK waveform')

%fft Xin SRRC with carrier
fftshift_SRRC_Xin = fftshift(fft(Xi,Nf)*Ts);
```

```
power_fftshift_SRRC_Xin = abs(fftshift_SRRC_Xin).^2; % zero-centered power

%fft Xin SRRC with carrier
fftshift_SRRC_Xqn = fftshift(fft(Xq,Nf)*Ts);
power_fftshift_SRRC_Xqn = abs(fftshift_SRRC_Xqn).^2; % zero-centered power

figure(8)
plot(freq,power_fftshift_SRRC_Xin)
grid on;
title('4) periodogram of instance Xin modulated F0=200Hz')
figure(9)
plot(freq,power_fftshift_SRRC_Xqn)
grid on;
title('4) periodogram of instance Xqn modulated F0=200Hz')
```




4) Xqn modulated F0=200Hz 16PSK waveform
30
20
10
0
-10
-20
-30
-0.2
0
0.2
0.4
0.6
0.8
1
1.2


4) periodogram of instance Xin modulated F0=200Hz
2.5
2
1.5
1
0.5
0
-500
0
500



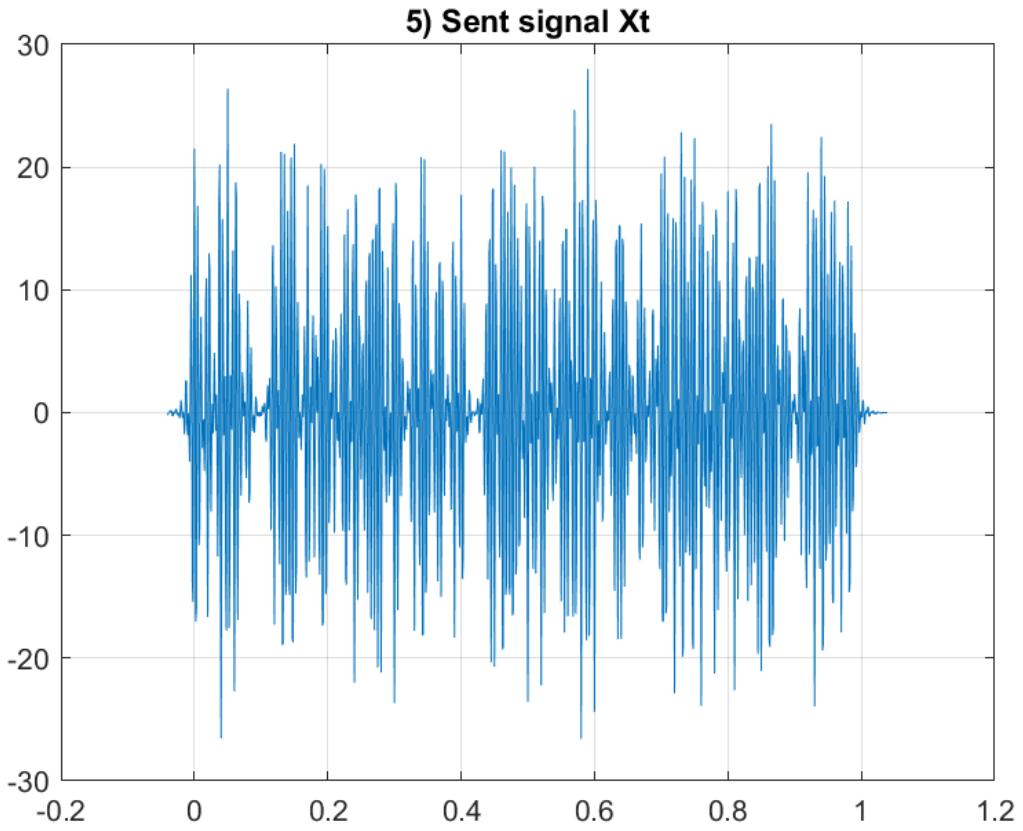
Παρατήρηση:
Παρατηρουμε οτι το φάσμα συχνοτήτων των κυματομορφών μετα τον πολλαπλασιασμό τους με το carrier signal μεταφέρθηκε γύρω απο τα 200Hz όπως και ηταν αναμενόμενο.

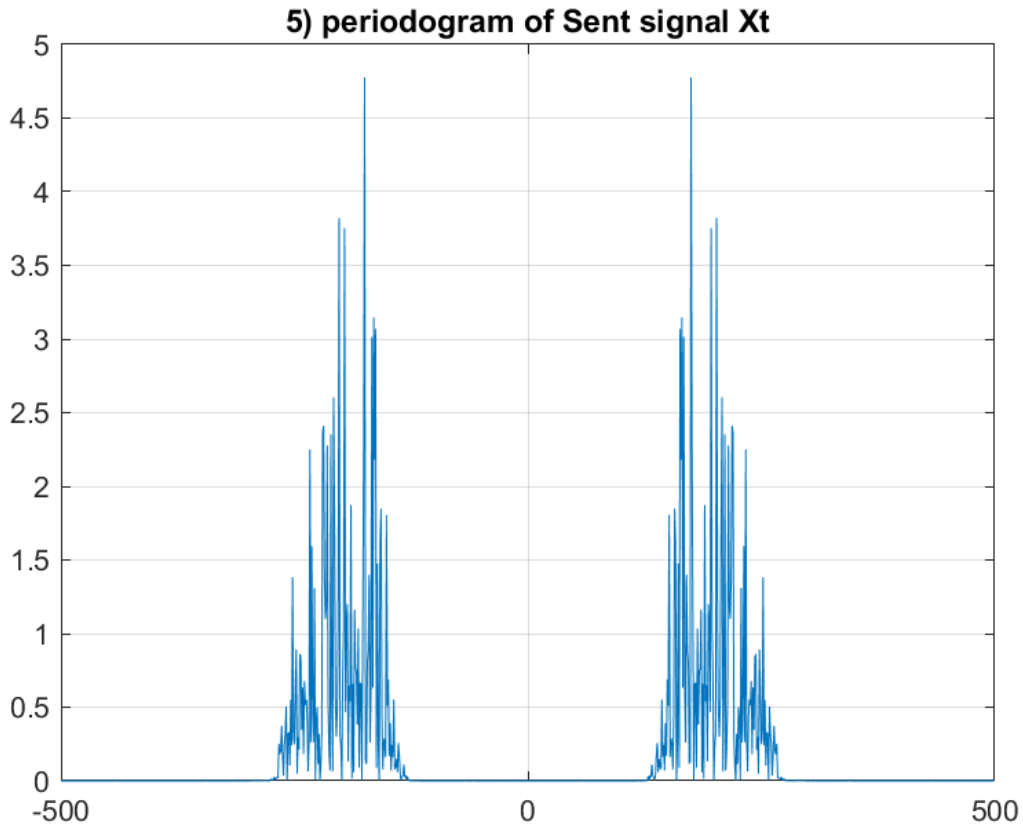
Ερώτημα 5:

Σχηματίστηκε και σχεδιάστηκε η είσοδος του καναλιού X(t) και το περιοδιαγγραμα της.

Κώδικας Matlab:

```
%5 add upp the 2 signals to be sent
Xt=Xi+Xq;
figure(10)
plot(signal_t,Xi);
grid on;
title('5) Sent signal Xt')

%periodogram of sender output
fftshift_Xt = fftshift(fft(Xt,Nf)*Ts);
power_fftshift_Xt = abs(fftshift_Xt).^2; % zero-centered power

figure(11)
plot(freq,power_fftshift_Xt)
grid on;
title('5) periodogram of Sent signal Xt')
```


5) Sent signal Xt
30
20
10
0
-10
-20
-30
-0.2
0
0.2
0.4
0.6
0.8
1
1.2


5) periodogram of Sent signal Xt
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
-500
0
500

Παρατήρηση:
Παρατηρουμε οτι και το φάσμα συχνοτήτων του σήματος εξόδου μετα τον πολλαπλασιασμό του με το carrier signal μεταφέρθηκε γύρω απο τα 200Hz.

Ερώτημα 7:

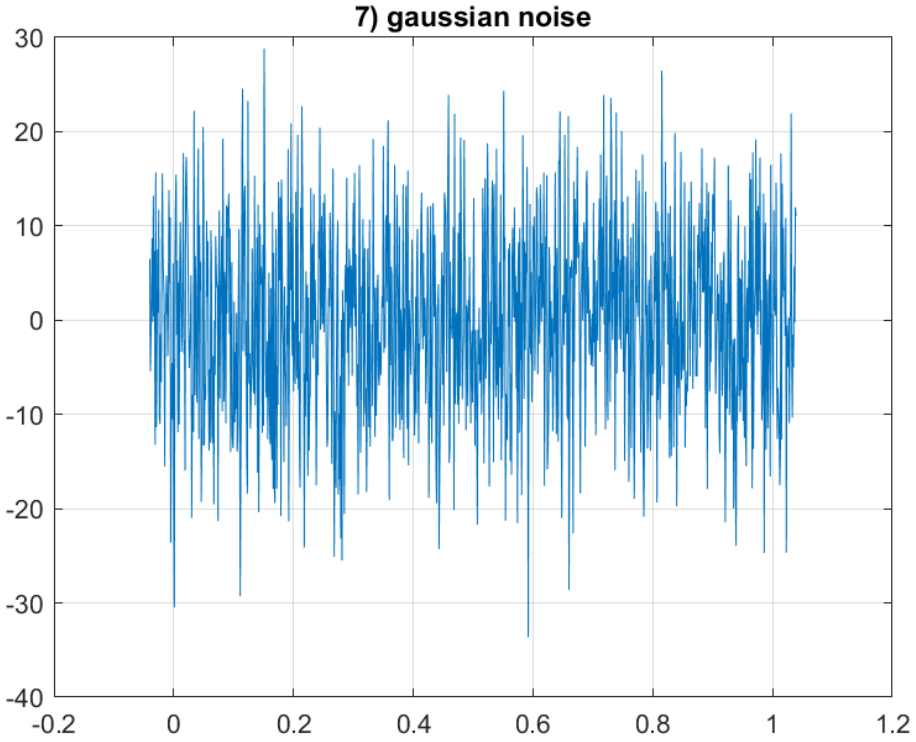
Στην συνέχεια υποθετουμε οτι το κανάλι είναι ιδανικό και έπειτα στην έξοδο του καναλιού προστέθηκε λευκό Gaussian θόρυβο W(t) με διασπορά ίση με

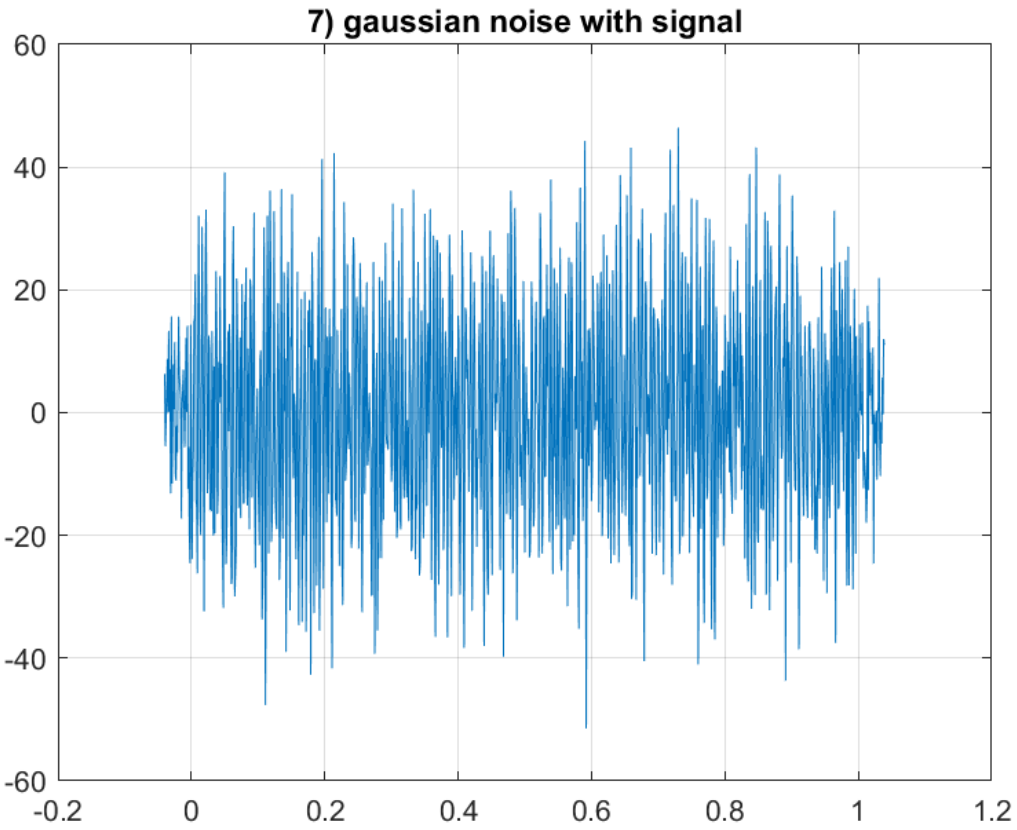
$$\sigma_W^2 = \frac{1}{Ts \times 10^{\frac{SNR_{db}}{10}}}$$

και με ενθόρυβη κυματομορφή $Y(t) = X(t) + W(t)$

Κώδικας Matlab:

```
%7 create gaussian noise
SNR=10;
noiseVar=1/(Ts*(10^(SNR/10)));
W=wgn(1,length(signal_t),noiseVar,'linear');

Y=Xt+W; %add noise

figure(12)
plot(signal_t,W);
grid on;
title('7) gaussian noise')

figure(13)
plot(signal_t,Y);
grid on;
title('7) gaussian noise with signal')
```


7) gaussian noise
30
20
10
0
-10
-20
-30
-40
-0.2
0
0.2
0.4
0.6
0.8
1
1.2

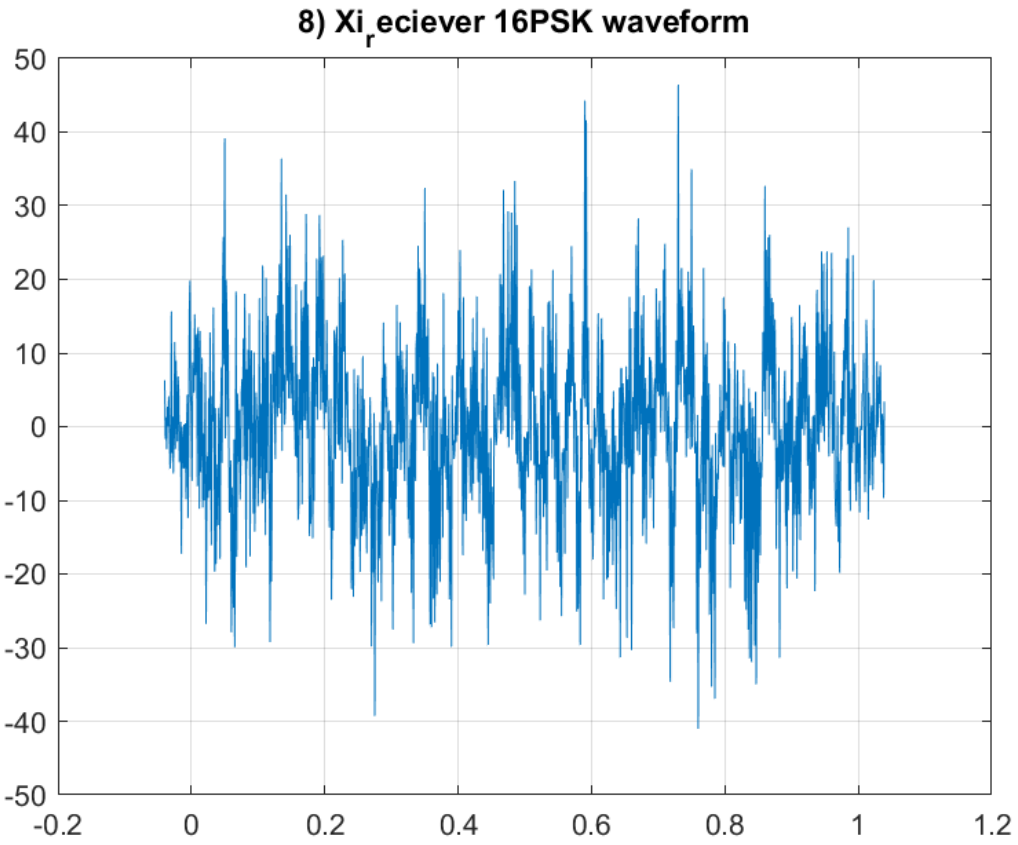

7) gaussian noise with signal
60
40
20
0
-20
-40
-60
-0.2
0
0.2
0.4
0.6
0.8
1
1.2

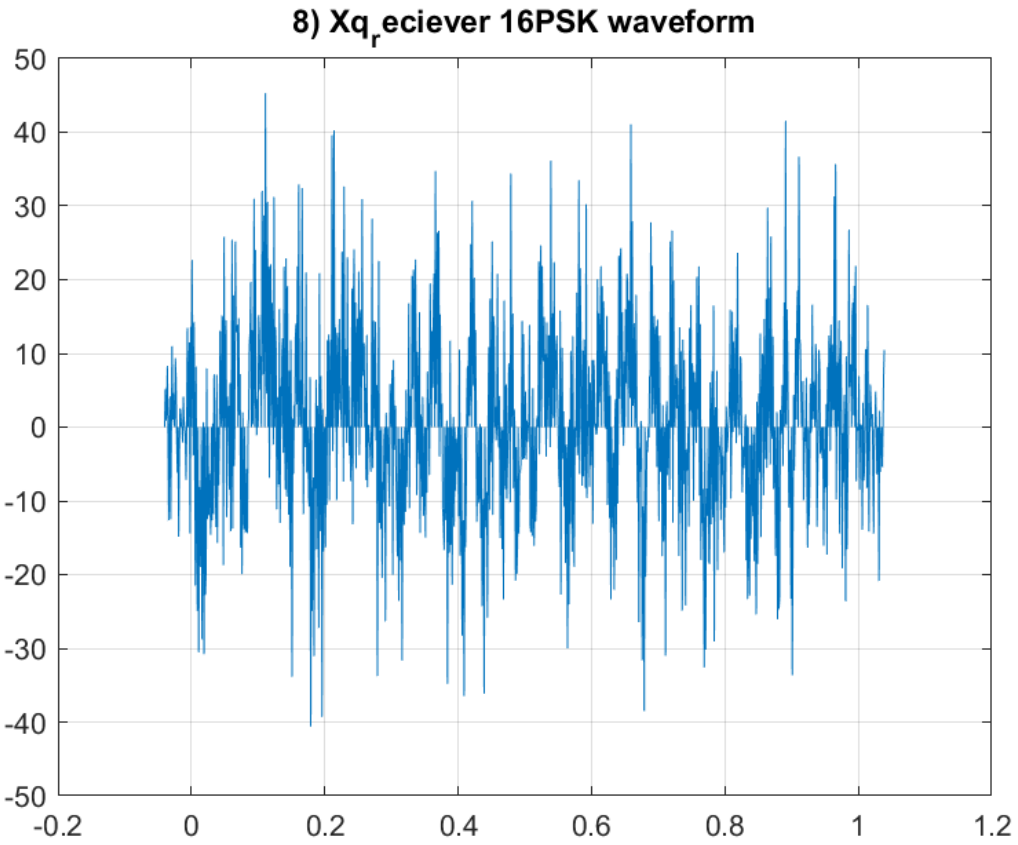
Ερώτημα 8:

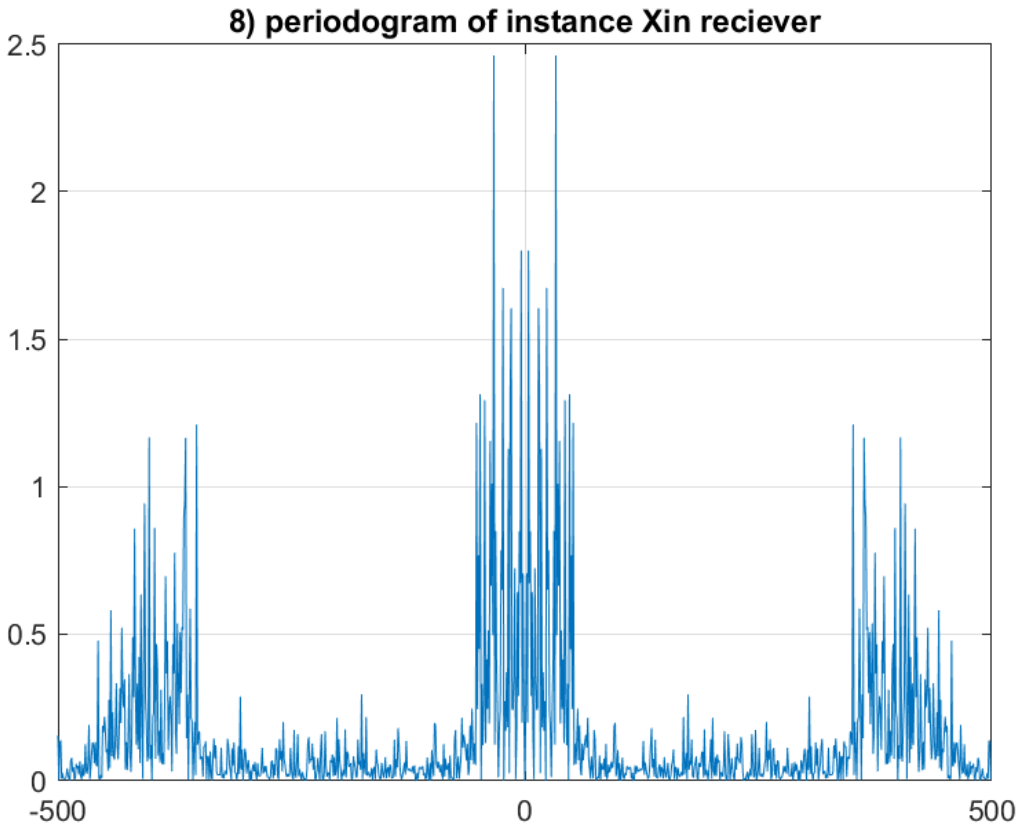
Πολλαπλασιάστηκε η ενθόρυβη κυματομορφή Y(t) στο δέκτη με τους κατάλληλους φορείς και σχεδιάστηκαν οι κυματομορφές που προκύπτουν και αντιστοίχως τα περιοδιαγράμματα τους.

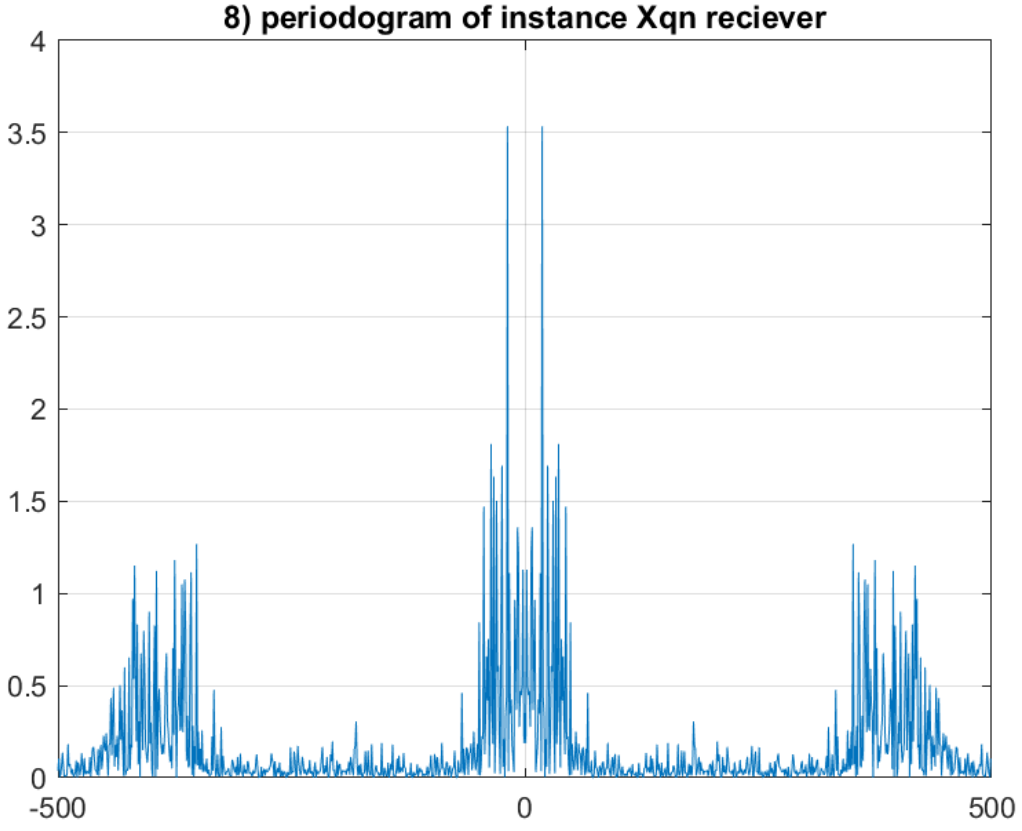
Κώδικας Matlab:

```
%8
F0=200 %Hz reciever freq
i_reciever=cos(2*pi*F0*signal_t);
q_reciever=-sin(2*pi*F0*signal_t);

Xi_reciever=Y.*i_reciever;
Xq_reciever=Y.*q_reciever;

figure(14)
plot(signal_t,Xi_reciever);
grid on;
title('8) Xi_reciever 16PSK waveform')
figure(15)
plot(signal_t,Xq_reciever);
grid on;
title('8) Xq_reciever 16PSK waveform')

%reciever periodogram
%fft Xin reciever
fftshift_reciever_Xin = fftshift(fft(Xi_reciever,Nf)*Ts);
power_fftshift_reciever_Xin = abs(fftshift_reciever_Xin).^2; % zero-centered
power

%fft Xin reciever
fftshift_reciever_Xqn = fftshift(fft(Xq_reciever,Nf)*Ts);
power_fftshift_reciever_Xqn = abs(fftshift_reciever_Xqn).^2; % zero-centered
power

figure(16)
plot(freq,power_fftshift_reciever_Xin)
grid on;
title('8) periodogram of instance Xin reciever')
figure(17)
plot(freq,power_fftshift_reciever_Xqn)
grid on;
title('8) periodogram of instance Xqn reciever')
```


8) Xi$_r$eciever 16PSK waveform
50
40
30
20
10
0
-10
-20
-30
-40
-50
-0.2
0
0.2
0.4
0.6
0.8
1
1.2


8) Xq$_r$eciever 16PSK waveform
50
40
30
20
10
0
-10
-20
-30
-40
-50
-0.2
0
0.2
0.4
0.6
0.8
1
1.2


8) periodogram of instance Xin reciever
2.5
2
1.5
1
0.5
0
-500
0
500


8) periodogram of instance Xqn reciever
4
3.5
3
2.5
2
1.5
1
0.5
0
-500
0
500

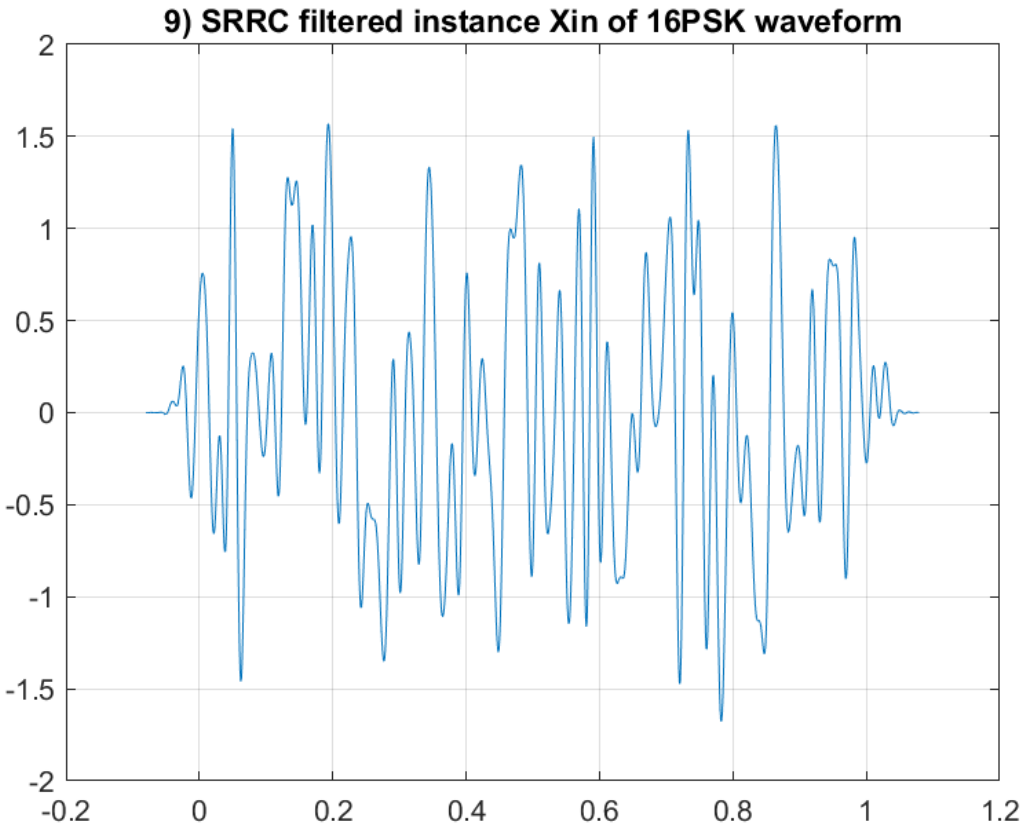
Παρατήρηση:
Χάρη στην ορθογωνιότητα των carrier signals μπορουμε να ξεχωρισουμε τα αρχικά σήματα και να τα ξανακάνουμε center στα 0Hz (Ο θόρυβος βέβαια παραμένει).

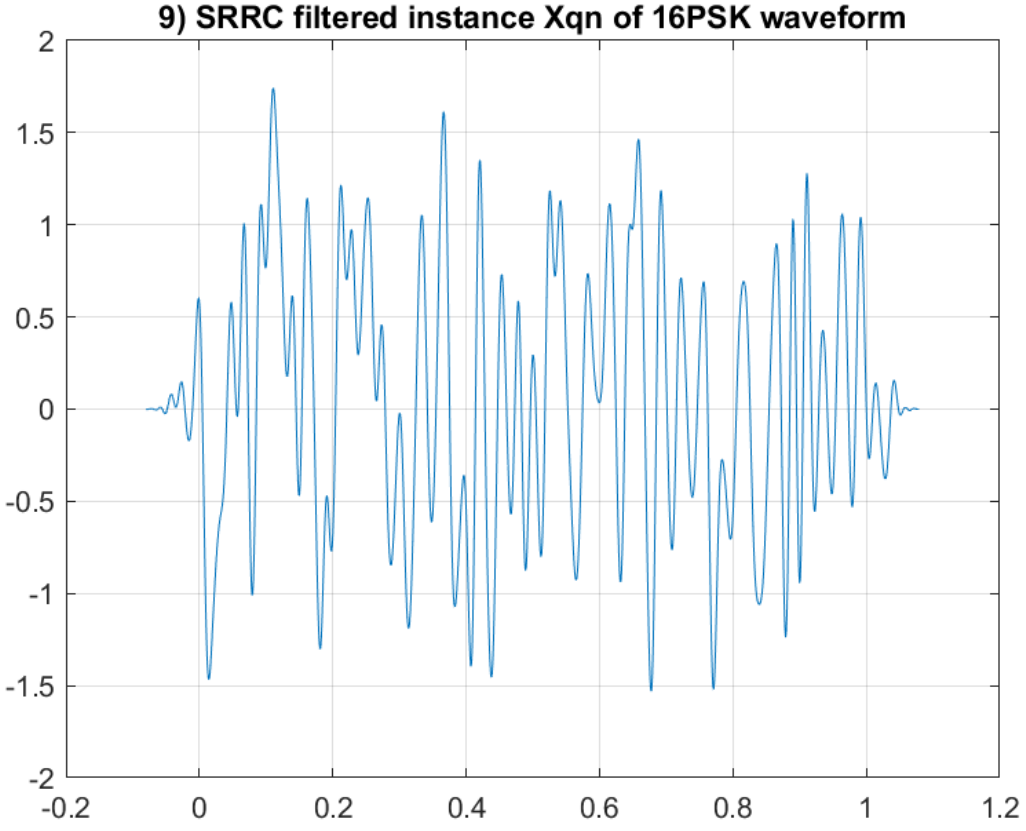
Ερώτημα 9:

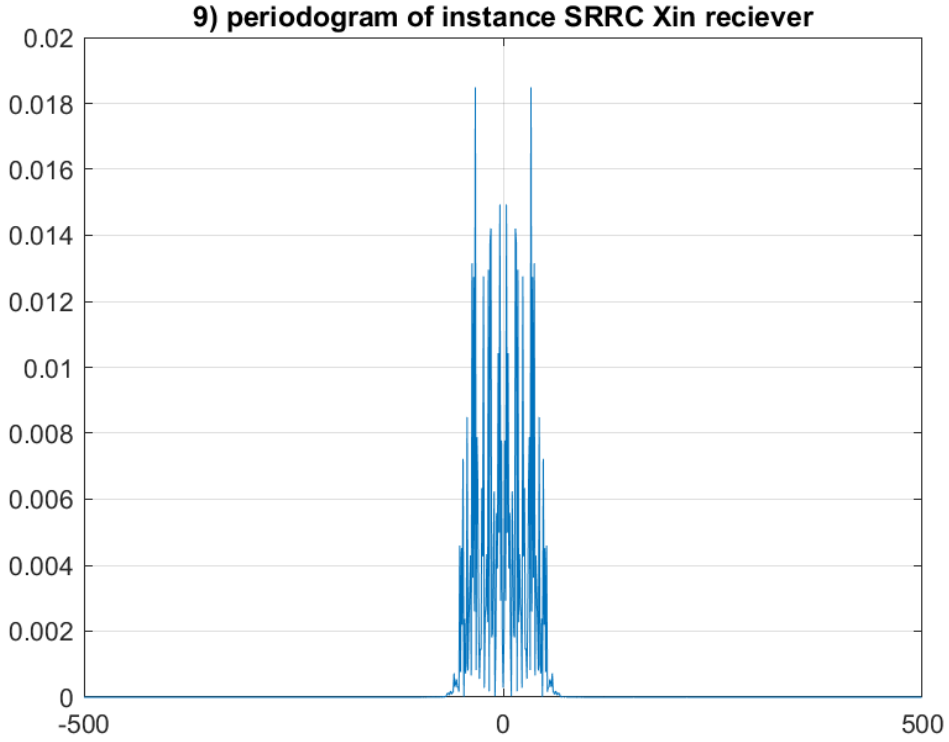
Περάστηκαν οι κυματομορφές που υπολογίστηκαν στο προηγούμενο βήμα από τα προσαρμοσμένα φίλτρα. Κατόπιν σχεδιάστηκαν οι κυματομορφές που προκύπτουν μαζί με τα περιοδιαγράμματα τους.

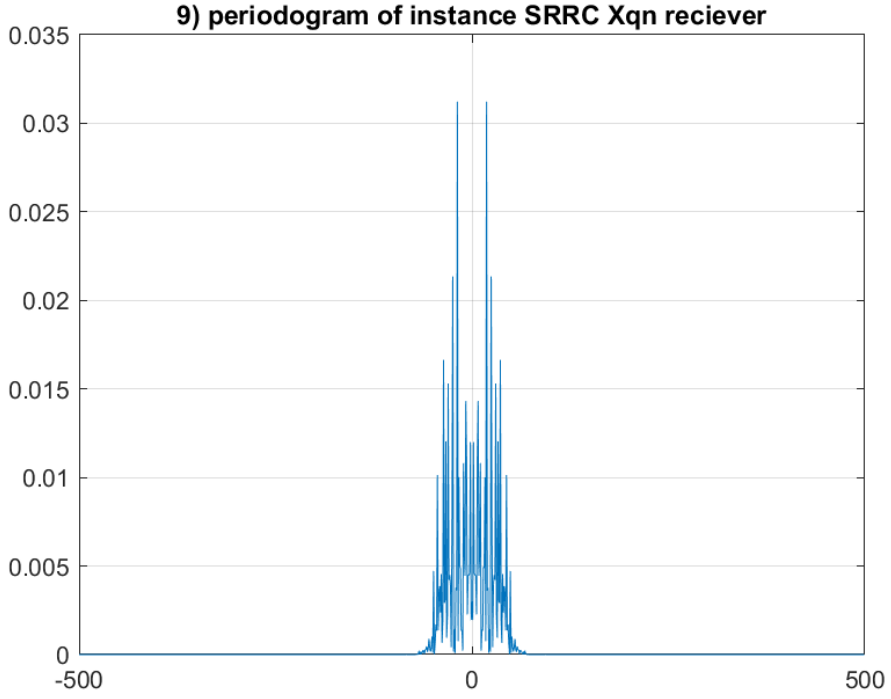
Κώδικας Matlab:

```
%9
Xin_srrc_reciever = conv(Xi_reciever,phi)*Ts; %convolute symbols waveform
with SRRC pulse
Xqn_srrc_reciever = conv(Xq_reciever,phi)*Ts; %convolute symbols waveform
with SRRC pulse
signal_t_reciever = [signal_t(1)+t(1):Ts:signal_t(end)+t(end)];
figure(18)
plot(signal_t_reciever,Xin_srrc_reciever);
grid on;
title('9) SRRC filtered instance Xin of 16PSK waveform')
figure(19)
plot(signal_t_reciever,Xqn_srrc_reciever);
grid on;
title('9) SRRC filtered instance Xqn of 16PSK waveform')

%defiltered reciever periodogram
%fft Xin reciever
fftshift_srrc_reciever_Xin = fftshift(fft(Xin_srrc_reciever,Nf)*Ts);
power_fftshift_srrc_reciever_Xin = abs(fftshift_srrc_reciever_Xin).^2; %
zero-centered power

%fft Xin reciever
fftshift_srrc_reciever_Xqn = fftshift(fft(Xqn_srrc_reciever,Nf)*Ts);
power_fftshift_srrc_reciever_Xqn = abs(fftshift_srrc_reciever_Xqn).^2; %
zero-centered power

figure(20)
plot(freq,power_fftshift_srrc_reciever_Xin)
grid on;
title('9) periodogram of instance SRRC Xin reciever')
figure(21)
plot(freq,power_fftshift_srrc_reciever_Xqn)
grid on;
title('9) periodogram of instance SRRC Xqn reciever')
```


9) SRRC filtered instance Xin of 16PSK waveform
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-0.2
0
0.2
0.4
0.6
0.8
1
1.2


9) SRRC filtered instance Xqn of 16PSK waveform
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-0.2
0
0.2
0.4
0.6
0.8
1
1.2





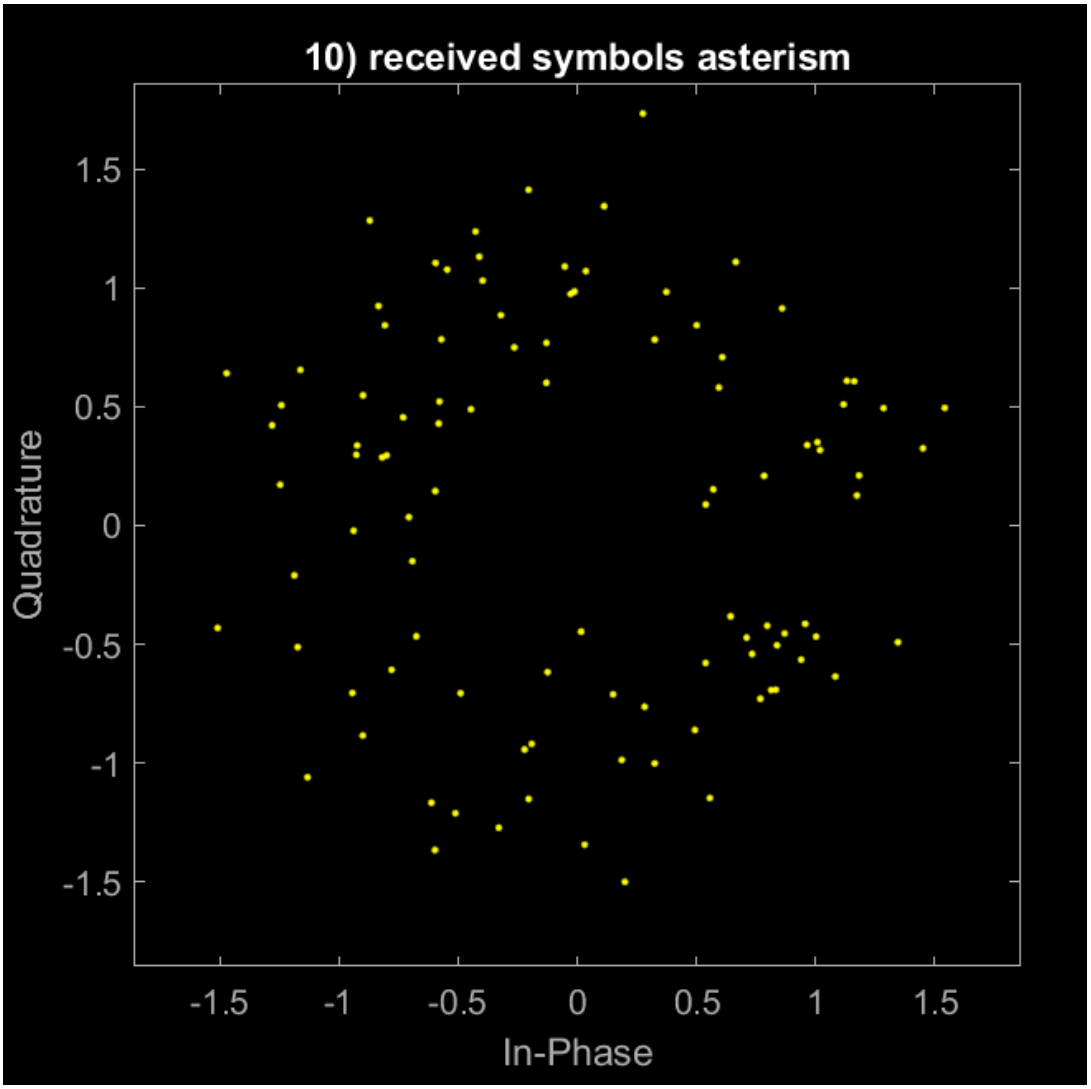
Παρατήρηση:
Παρατηρούμε οτι πλην του effect του θορυβου έχουμε μια πολύ κοντινή ανακατασκευη των σημάτων που στείλαμε μέσω του πομπού.

Ερώτημα 10:

Δειγματοληπτήθηκε η έξοδος των προσαρμοσμένων φίλτρων στις κατάλληλες χρονικές στιγμές και σχεδιάστηκε η ακολουθία εξόδου Y χρησιμοποιώντας την εντολή *scatterplot.*

<u>Κώδικας Matlab:</u>

```
%10
Xi_sampled=Xin_srrc_reciever(1:over:end);
Xq_sampled=Xqn_srrc_reciever(1:over:end);
Xi_sampled=Xi_sampled(9:end-8);
Xq_sampled=Xq_sampled(9:end-8);
X_reciever(1,:)=Xi_sampled;
X_reciever(2,:)=Xq_sampled;
figure(22)
scatterplot(rot90(X_reciever));
title('10) received symbols asterism')
```



Ερώτημα 11:

Γράφτηκε η συνάρτηση function *[est_X, est_bit_seq] = detect_PSK_16(Y)* η οποία χρησιμοποιεί τον κανόνα εγγύτερου γείτονα και αποφασίζει για την ακολουθία

Κώδικας Matlab:

```
function [output] = detect_PSK_16(input)

%input = rot90(input); % Convert input to column vector

bin_conversion_Table = [
0 0 0 0;
0 0 0 1;
0 0 1 0;
0 0 1 1;
0 1 0 0;
0 1 0 1;
0 1 1 0;
0 1 1 1;
1 0 0 0;
1 0 0 1;
1 0 1 0;
1 0 1 1;
1 1 0 0;
1 1 0 1;
1 1 1 0;
1 1 1 1
];

gray_conversion_Table = [
0 0 0 0;
0 0 0 1;
0 0 1 1;
0 0 1 0;
0 1 1 0;
0 1 1 1;
0 1 0 1;
0 1 0 0;
1 1 0 0;
1 1 0 1;
1 1 1 1;
1 1 1 0;
1 0 1 0;
1 0 1 1;
1 0 0 1;
1 0 0 0
];

%output = zeros(numel(input) * 4, 1); % Preallocate output vector
output=[];

for i = 1:length(input)
      symbol_in = input(1,i);
```

```
        symbol_qn = input(2,i);
        angles_in = cos((0:15) * 2 * pi / 16); % Compute angles for the 16-PSK
        angles_qn = sin((0:15) * 2 * pi / 16); % Compute angles for the 16-PSK
        constellation
        distances = sqrt((symbol_in - angles_in).^2 + (symbol_qn -
        angles_qn).^2);

        [~, index] = min(distances) ; % Find the index of the nearest neighbor
        gray_bin=bin_conversion_Table(index,:);
        [~, dec_symbol] = ismember(gray_bin, gray_conversion_Table, 'rows');
        bin=bin_conversion_Table(dec_symbol,:);
        output = [output bin];
end

end
```

Κώδικας Matlab:

```
%11
out_b=detect_PSK_16(X_reciever);
bit_error=num_of_bit_errors(out_b,rot90(bit_seq))
sym_error=num_of_symbol_errors(X,X_reciever)
out_b(end-11:end)
rot90(bit_seq(end-11:end))
```

Ερώτημα 12:

Υλοποιήθηκε η συνάρτηση *function num_of_symbol_errors = symbol_errors(est_X, X)*

Κώδικας Matlab:

```
function [output] = num_of_symbol_errors (arr1,arr2)
output = 0;
for i = 1:length(arr1)*2
        if abs(arr1(i) - arr2(i)) >= 0.05
                output = output + 1;
        end
end
end
```

Ερώτημα 13:

Υλοποιήθηκε η συνάρτηση *function num_of_bits_errors = bits_errors(est_bit_seq, b)* η οποία υπολογίζει το πλήθος των σφαλμάτων εκτίμησης bit.

Κώδικας Matlab:

```
function [ output ] = num_of_bit_errors( arr1,arr2 )
output = sum(arr1 ~= arr2);
end
```

2ο Μέρος

Σε αυτό το μέρος της εργαστηριακής άσκησης εκτιμήθηκε η πιθανότητα σφάλματος συμβόλου και bit με την χρήση της μεθόδου Monte Carlo.

Ερώτημα 1:

Για SNR_{dB} = [-2 : 2 : 24], εκτιμήθηκε η πιθανότητα σφάλματος απόφασης συμβόλου και bit επαναλαμβάνοντας τα παραπάνω βήματα για K = 1000, για κάθε SNR.

Πιθανότητα σφάλματος συμβόλου:

$$\widehat{P}(Esymbol) = \frac{\text{συνολικό πλήθος σφαλμάτων απόφασης συμβόλου}}{\text{συνολικό πλήθος απεσταλμένων συμβόλων}}$$

Πιθανότητα σφάλματος bit:

$$\widehat{P}(Ebit) = \frac{\text{συνολικό πλήθος σφαλμάτων απόφασης } bit}{\text{συνολικό πλήθος απεσταλμένων } bits}$$

Ερώτημα 2:

Κατόπιν σχεδιάστηκε σε semilogy η εκτιμωμενη πιθανότητα σφάλματος συμβόλου ως συνάρτηση SNR_{dB}. Επιπρόσθετα στο ίδιο σχήμα σχεδιάστηκε και το έξυπνο άνω φράγμα για πιθανότητα σφάλματος συμβόλου.

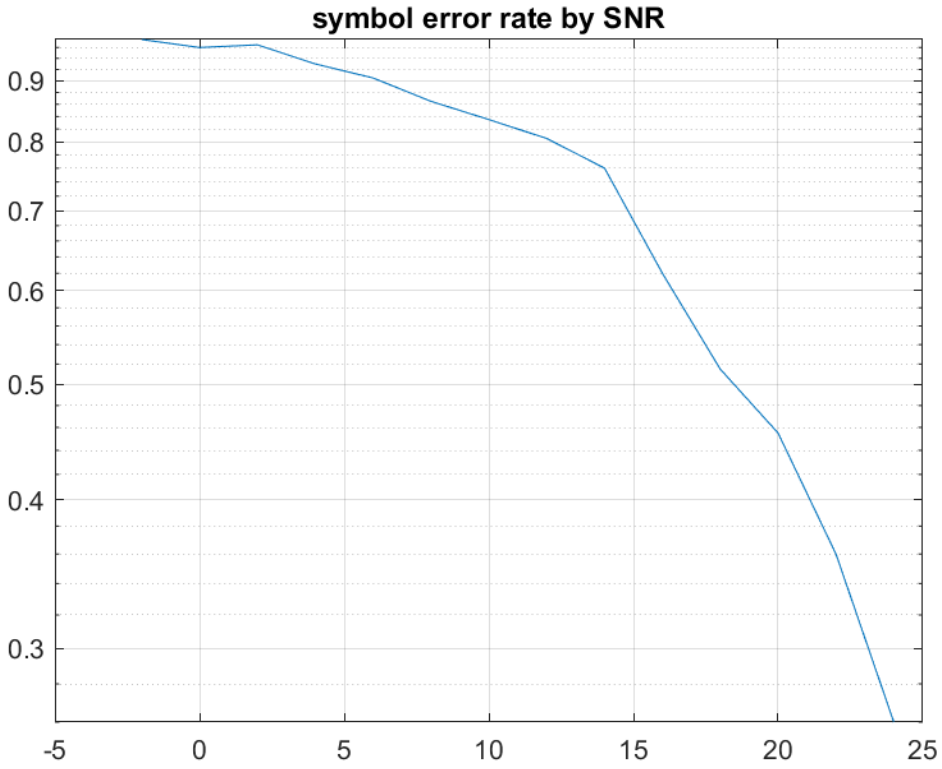
Ερώτημα 3:

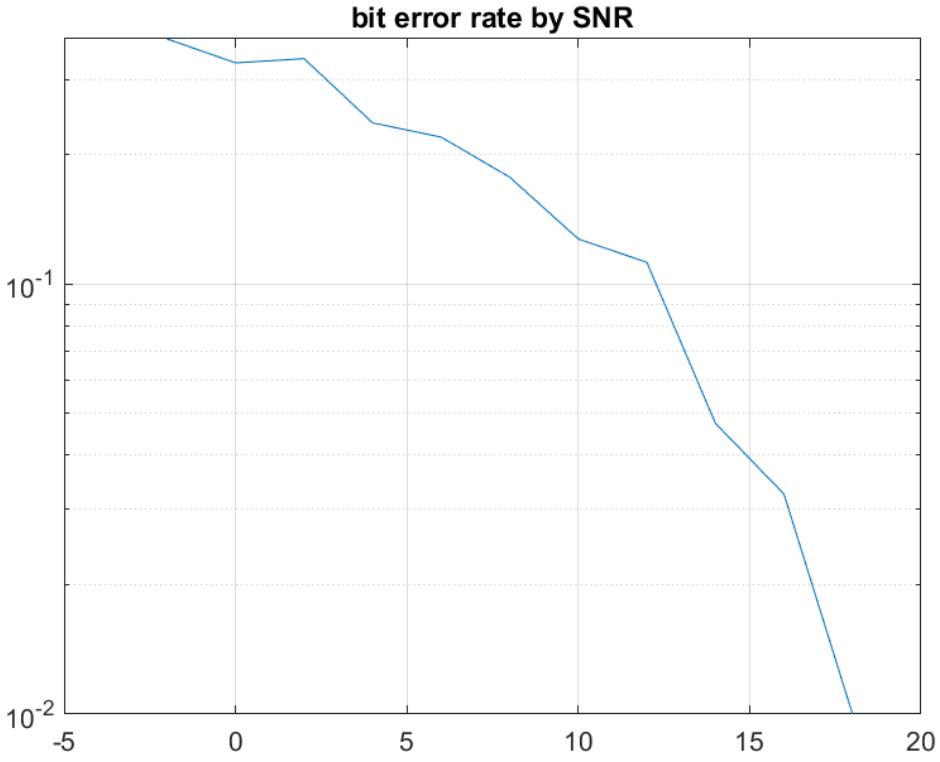
Τέλος σχεδιάστηκε σε semilogy η εκτιμωμενη πιθανότητα σφάλματος bit ως συνάρτηση του SNR_{dB}. Επιπρόσθετα στο ίδιο σχήμα σχεδιάστηκε το κάτω φράγμα για την πιθανότητα σφάλματος bit, το οποίο προκύπτει από την εκτιμώμενη πιθανότητα σφάλματος συμβόλου και της κωδικοποίησης Gray.

Κώδικας Matlab:

```
clear all
close all

K=1000
j=0;

for SNR=-2:2:24
      j=j+1;
      for i=1:K
            %16-PSK
            N=100; %number of symbols
            bit_seq = (sign(randn(4*N, 1)) + 1)/2; %create random bits
            X = bits_to_PSK_16(bit_seq); %turn bits into 2x100 gray coded
            coordinates
            rot90(bit_seq);
            %divide coordinates into x,y
            real=X(1,:);
            imag=X(2,:);

            %3
            %initial parameters
            T=0.01;
            over=10;
            Ts=T/over;
            A=4;
            a=0.5;
            %create SRRC pulse
            [phi, t] = srrc_pulse(T, over, A, a);
            %upsample
            X_real_delta = 1/Ts * upsample(real, over);
            X_imag_delta = 1/Ts * upsample(imag, over);
            %define time
            time = 0:Ts:N*Ts*over-Ts;
            signal_t = [time(1)+t(1):Ts:time(end)+t(end)];
            Xin_srrc   =   conv(X_real_delta,phi)*Ts;   %convolute   symbols
            waveform with SRRC pulse
            Xqn_srrc   =   conv(X_imag_delta,phi)*Ts;   %convolute   symbols
            waveform with SRRC pulse

            %4
            %define carrier signals
            F0=200; %Hz carrier freq
            i_carrier=2*cos(2*pi*F0*signal_t);
            q_carrier=-2*sin(2*pi*F0*signal_t);
            %multiply main signals with the 2 orthogonal carrier signals in
            order to be
            %sent in parralel
            Xi=Xin_srrc.*i_carrier;
            Xq=Xqn_srrc.*q_carrier;

            %5 add upp the 2 signals to be sent
            Xt=Xi+Xq;
```

```
            %7 create gaussian noise
            noiseVar=1/(Ts*(10^(SNR/10)));
            W=wgn(1,length(signal_t),noiseVar,'linear');
            Y=Xt+W; %add noise

            %8
            F0=200 %Hz reciever freq
            i_reciever=cos(2*pi*F0*signal_t);
            q_reciever=-sin(2*pi*F0*signal_t);
            Xi_reciever=Y.*i_reciever;
            Xq_reciever=Y.*q_reciever;

            %9
            Xin_srrc_reciever = conv(Xi_reciever,phi)*Ts; %convolute symbols
            waveform with SRRC pulse
            Xqn_srrc_reciever = conv(Xq_reciever,phi)*Ts; %convolute symbols
            waveform with SRRC pulse
            signal_t_reciever = [signal_t(1)+t(1):Ts:signal_t(end)+t(end)];

            %10
            Xi_sampled=Xin_srrc_reciever(1:over:end);
            Xq_sampled=Xqn_srrc_reciever(1:over:end);
            Xi_sampled=Xi_sampled(9:end-8);
            Xq_sampled=Xq_sampled(9:end-8);
            X_reciever(1,:)=Xi_sampled;
            X_reciever(2,:)=Xq_sampled;

            %11
            out_b=detect_PSK_16(X_reciever);
            P_bit_error(j)=num_of_bit_errors(out_b,rot90(bit_seq))/(4*N);
            P_symbol_error(j)=num_of_symbol_errors(X,X_reciever)/(2*N);
end
end

P_bit_error
P_symbol_error
t=-2:2:24
figure(25);
semilogy(t,P_symbol_error);
grid on;
title('symbol error rate by SNR')
figure(26);
semilogy(t,P_bit_error);
grid on;
title('bit error rate by SNR')
```





<u>Παρατήρηση:</u>

Παρατηρούμε ότι έχουμε εμφανώς μικρότερο error rate μετα απο την μετάφραση από κώδικα gray, κάτι που οφείλεται στην ιδιοτητα του κάθε σύμβολο να διαφέρει μόνο 1 bit κάνοντας τον έτσι πιο ανθεκτικό στα σφάλματα.