Τηλεπικοινωνιακά Συστήματα 1

Αναφορά $3^{ης}$ άσκησης

17/12/2020

Στοιχεία Ομάδας

Αλέξανδρος Σεργάκης: 2017030199

Ευάγγελος Κατσούπης: 2017030077

Α. Σε αυτή την άσκηση, θα προσομοιώσουμε το τηλεπικοινωνιακό σύστημα του Σχήματος,υποθέτοντας ότι χρησιμοποιείται διαμόρφωση 8-PSK, και θα μελετήσουμε την απόδοσή του.

1. Για δεδομένο N (ενδεικτικά, $N = 100$), να δημιουργήσετε δυαδική ακολουθία bit_ seq με στοιχεία $3N$ ισοπίθανα bits.

   Για την δημιουργία της δυαδικής ακολουθίας χρησιμοποιήσαμε την συνάρτηση randn για να δημιουργήσουμε μία ακολουθία με $3N$ βιτ.

   Ο κώδικας για την δημιουργία του bit array είναι η εξης γραμμή:

```
%% 1.Generate Bit seq
N = 100;
% 3N bits generation
b_seq = (sign(randn((3*N), 1)) + 1)/2;
```

2. Να γράψετε συνάρτηση $bits\_to\_PSK8(bit\_seq)$ η οποία, χρησιμοποιώντας κωδικοποίηση Gray, απεικονίζει τη δυαδική ακολουθία εισόδου bit_seq σε ακολουθία 8_PSK συμβόλων Q, μήκους N με στοιχεία τα διδιάστατα διανύσματα

$$X_n = \begin{bmatrix} X_{I,n} = \cos(\frac{2\pi m}{8}\pi) \\ X_{Q,n} = \sin(\frac{2\pi m}{8}\pi) \end{bmatrix}, \text{ για } n = 0, \cdots, N-1.$$

   Ο κώδικας που μετατρέπει μια δυαδική ακολουθία σε μια ακολουθία από σύμβολα 8_PSK είναι ο εξής:

```
function [X] = bits_to_PSK_8(b_seq)
%BITS_TO_PSK_8
%   Converts a 3 bit Gray Code (Xn) itno a 8_PSK Symbol (Xm).
%
%           | Xi,m=cos(2*pi*m/8)|
%      Xm = |                   |
%           | Xq,m=sin(2*pi*m/8)|
```

```
%
%   [000] ->  m = 0
%   [001] ->  m = 1
%   [011] ->  m = 2
%   [010] ->  m = 3
%   [110] ->  m = 4
%   [111] ->  m = 5
%   [101] ->  m = 6
%   [100] ->  m = 7
A = 1;
X=zeros(length(b_seq)/3,2);
for i = 1:3:length(b_seq)
    bits=b_seq(i:i+2);
    index = (i+2)/3;
    if bits == [0;0;0]
        X(index,:) = Coordinates(0);
    elseif  bits == [0;0;1]
        X(index,:) = Coordinates(1);
    elseif  bits == [0;1;1]
        X(index,:) = Coordinates(2);
    elseif  bits == [0;1;0]
        X(index,:) = Coordinates(3);
    elseif  bits == [1;1;0]
        X(index,:) = Coordinates(4);
    elseif  bits == [1;1;1]
        X(index,:) = Coordinates(5);
    elseif  bits == [1;0;1]
        X(index,:) = Coordinates(6);
    else
        X(index,:) = Coordinates(7);
    end
end
% Returns the Coordinates of the vector
function a = Coordinates(m)
    a=A*[cos((2*pi*m)/8) sin((2*pi*m)/8)];
end
end
```

Η ακολουθία από σύμβολα 8_PSK που δημιουργήσαμε είναι της εξής μορφής:

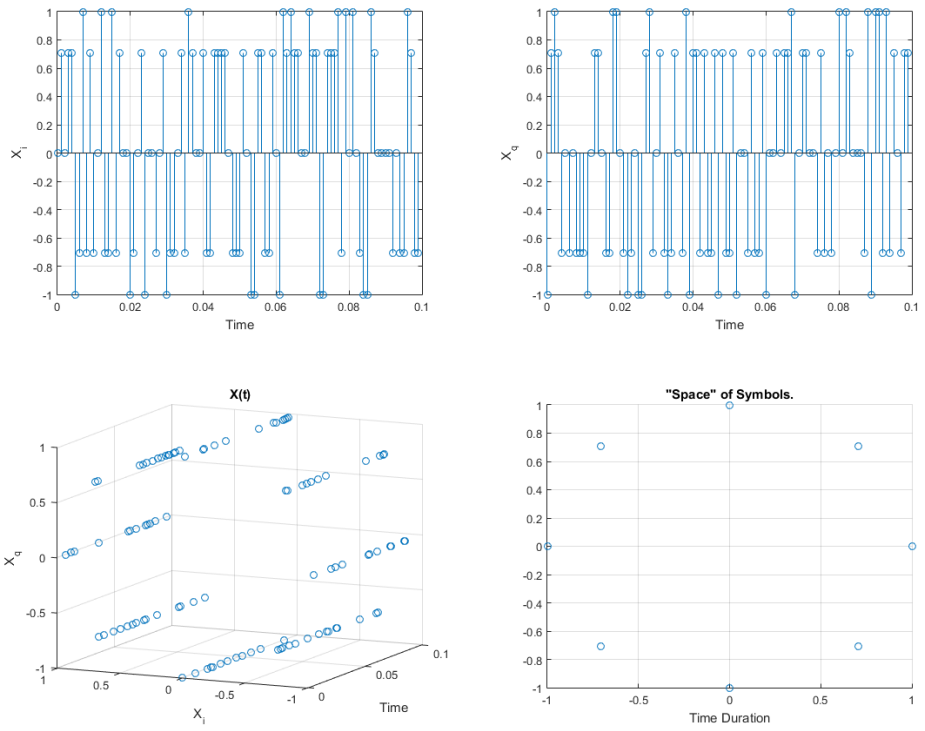

Figure 1: 1.$X_{In-phase}$, 2.$X_{Quadrature}$, 3. 3-D generated signal X, 4.Scatter plot X

3. Να περάσετε τις ακολουθίες $X_(I, n)$ και $X_(Q, n)$ από τα SRRC φίλτρα μορφοποίησης και υποθέτοντας, ενδεικτικά, περίοδο συμβόλου $T = 10^{-3}$ sec, $over = 10$, $T_s = \frac{T}{over}$, να σχηματίσετε και να σχεδιάσετε τις κυματομορφές εξόδου, και τα περιοδογράμματά τους.

   Αρχικά κάνουμε upsample και τις 2 ακολουθίες $X_(i, n)$ και $X_(Q, n)$ ώστε να φτάσουμε το βήμα του SRRC lowpass φίλτρου και μετά κάνουμε συνέλιξη των ακολουθιών με αυτό. Στην συνέχεια κάνουμε μετασχηματισμό Fourier στις ακολουθίες , για να υπολογήσουμε τα τις φασματικές πυκνότητες υσχύων τους και στην συνέχεια αποτυπώνουμε τα περιοδογράμματα τους σε γραμμική και ημιλογαριθμική κλίμακα.

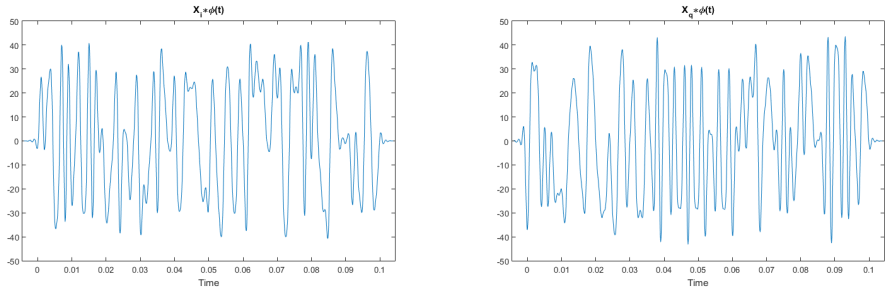

Figure 2: 1.$X_{In-phase}$, 2.$X_{Quadrature}$

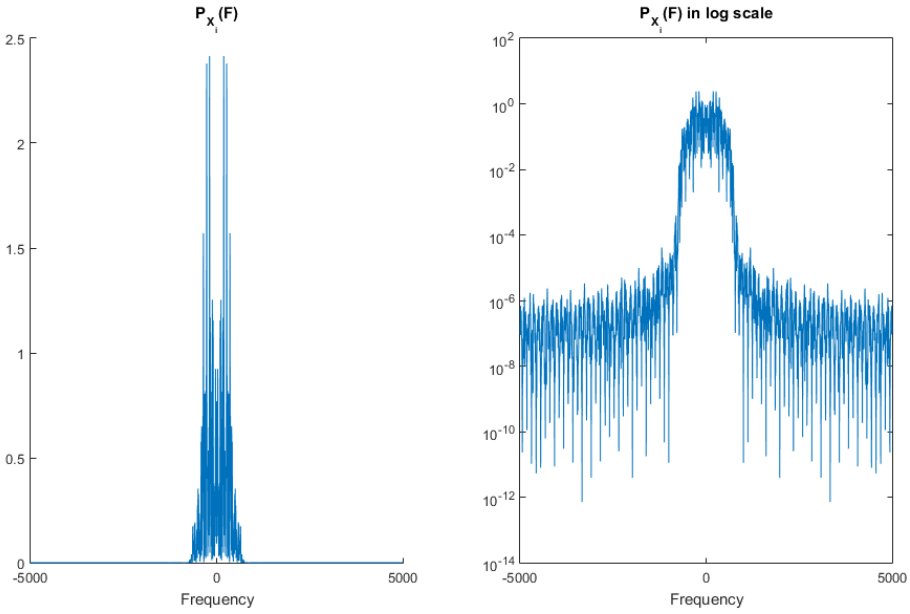

Figure 3: $X_{In-phase}$ Periodograms

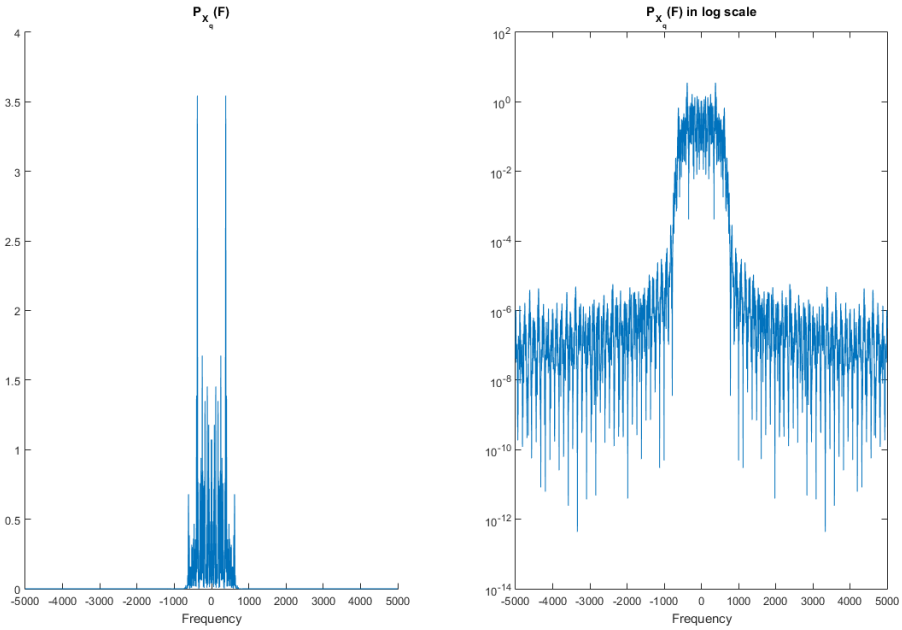

Figure 4: $X_{Quadrature}$ Periodograms

Για την δημιουργία του SRRC παλμού χρησιμοποιήθηκε η γνώστή πλέων συνάρτηση srrc_pulse με τα παρακάτω στοιχεία:

```
% Generate the palse
T = 0.001;
over = 10;
Ts = T/over;
a = 0.5;
A = 4;
[phi, t] = srrc_pulse(T, over, A, a);
```

Ο κώδικας για την υλοποίηση της μορφοποίησης αλλά και για την εκτύπωση των plots:

```
X_upsample=(1/Ts) .* upsample(X, over);
Xi_upsample = X_upsample(:,1);
Xq_upsample = X_upsample(:,2);
% Signal Duration
X_time = 0:Ts:(N*T)-Ts;
```

```
% Convolition with SRRC
Xi_conv = conv(Xi_upsample, phi)*Ts;
Xq_conv = conv(Xq_upsample, phi)*Ts;
time_conv = t(1) + X_time(1):Ts:t(end) + X_time(end);

X_conv(:,1) = Xi_conv;
X_conv(:,2) = Xq_conv;

figure('Name','X-PHI');
subplot(1,2,1);
plot(time_conv,Xi_conv);
title('X_i\ast\phi(t)');
xlabel('Time');
xlim([(time_conv(1)-5*Ts) (time_conv(end)+5*Ts)]);
hold on;
subplot(1,2,2);
plot(time_conv,Xq_conv);
title('X_q\ast\phi(t)');
xlabel('Time');
xlim([(time_conv(1)-5*Ts) (time_conv(end)+5*Ts)]);

% Periodograms
Nf = 4096;
Fs = 1/Ts; % Sampling Frequency
f_axis = linspace(-Fs/2,(Fs/2-Fs/Nf), Nf);
% Xi_F
Xi_F = fftshift(fft(Xi_conv, Nf)*Ts);
Pxi_F_num = power(abs(Xi_F),2);
Pxi_F = Pxi_F_num./(time_conv(end)-time_conv(1));

%Xq_F
Xq_F = fftshift(fft(Xq_conv, Nf)*Ts);
Pxq_F = power(abs(Xq_F),2)./(time_conv(end)-time_conv(1));

% Periodogram of Xi
figure('Name','X_i Peridogram');
subplot(1,2,1);
```

```
hold on;
plot(f_axis, Pxi_F);
title('P_{X_i}(F)');
xlabel('Frequency');
xlim([-Fs/2 Fs/2]);
hold on;
subplot(1,2,2);
semilogy(f_axis, Pxi_F);
title('P_{X_i}(F) in log scale');
xlabel('Frequency');
xlim([-Fs/2 Fs/2]);

% Periodogram of Xq
figure('Name','X_q Peridogram');
subplot(1,2,1);
hold on;
plot(f_axis, Pxq_F);
title('P_{X_q}(F)');
xlabel('Frequency');
xlim([-Fs/2 Fs/2]);
hold on;
subplot(1,2,2);
semilogy(f_axis, Pxq_F);
title('P_{X_q}(F) in log scale');
xlabel('Frequency');
xlim([-Fs/2 Fs/2]);
```

Λόγου του ότι η διαδηκασία απεικόνησης των περιοδογραμμάτων ακολούθεί όλλες τις φορές την παραπάνω διαδηκασία (figures σε λογαριθμική και γραμμική μορφή) δεν θα ξαναδειχθεί στην αναφορά για τα επόμενα ερωτήματα.

4. Να πολλαπλασιάσετε τις κυματομορφές με τους αντίστοιχους φορείς (ενδεικτικά, $F0 = 2000Hz$) και να σχεδιάσετε τις κυματομορφές που προκύπτουν, $X_I(t)$ και $X_Q(t)$,καθώς και τα αντίστοιχα περιοδογράμματα. Τι παρατηρείτε;

   Πολλαπλασιάσαμε την ακολουθία Q_I με τον φορέα $2 \cdot cos(2\pi F_0 t)$ και την Q_Q με το $(-2) \cdot sin(2\pi Fot)$ όπου $F_0 = 2000Hz$. Στην συνέχεια κάνουμε μετασχηματισμό Fourier και αποτυπώνουμε τα περιοδογράμματα τους σε γραμμική και ημιλογαριθμική κλίμακα. Αυτό που βλέπουμε με την βοήθεια των ημιλογαριθμικών περιοδογραμμάτων είναι ότι η $X_I(t)$ και η $Q_Q(t)$ μεταδίδονται γύρω από την συχνότητα διαμόρφωσης $F0 = 2000Hz$.

   Πολλαπλασιασμός με φορείς:

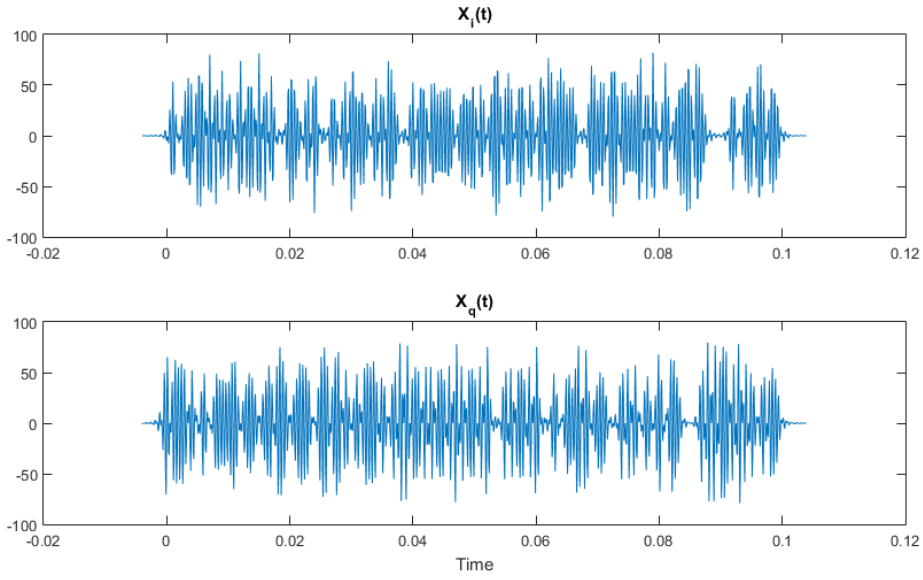

Figure 5: 1.$X_{In-phase}$, 2.$X_{Quadrature}$

Περιοδογράμματα:

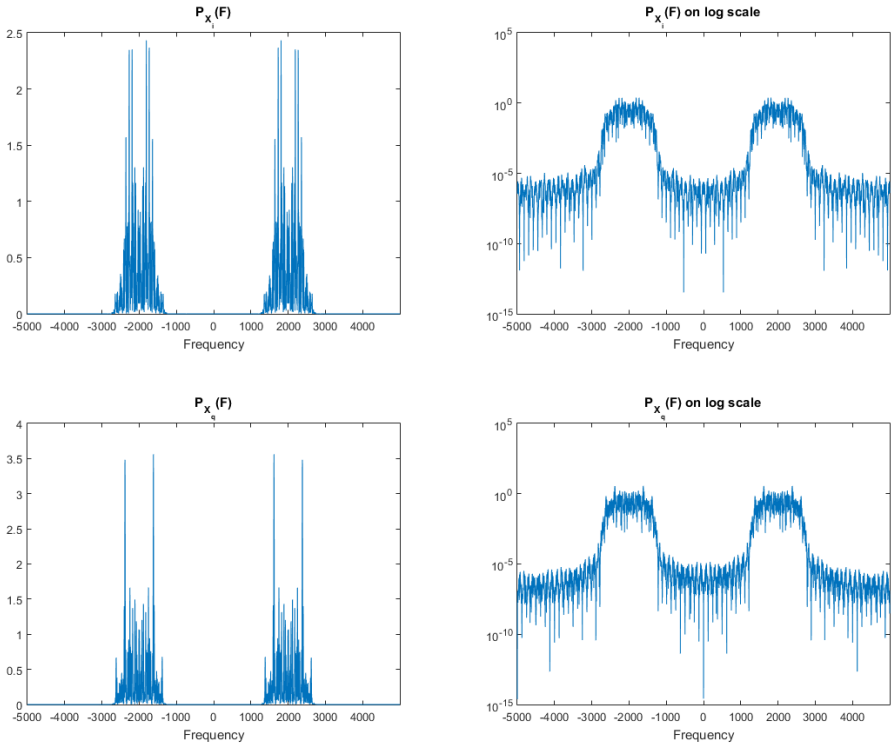

Figure 6: 1.$P_{X_{In-phase}}(F)$, 2.$P_{X_{Quadrature}}(F)$

Κώδικας για την υλοποίηση του ερωτήματος:

```
%% 4.multiply with carries
time = time_conv;
F0 = 2000;
% Caries
Ci = (2*cos(2*pi*F0*time))';
Cq = ((-2)*sin(2*pi*F0*time))';

Xi_c = Xi_conv .* Ci;
Xq_c = Xq_conv .* Cq;

% Plots - Periodograms
figure('Name','Xi_c');
subplot(2,1,1);
plot(time,Xi_c);
```

```
title('X_i(t)');
hold on;
subplot(2,1,2);
plot(time,Xq_c);
title('X_q(t)');
xlabel('Time');

% Periogramms
% P_Xi
Xi_c_F = fftshift(fft(Xi_c, Nf)*Ts);
P_Xi_c_F = power(abs(Xi_c_F),2)./(time(end)-time(1));
% P_Xq
Xq_c_F = fftshift(fft(Xq_c, Nf)*Ts);
P_Xq_c_F = power(abs(Xq_c_F),2)./(time(end)-time(1));
```

5. Να σχηματίσετε και να σχεδιάσετε την είσοδο του καναλιού, $X(t)$, και το περιοδόγραμμά της. Τι παρατηρείτε;

   Για τον σχεδιασμό της εισόδου του καναλιού, $X(t)$ προσθέσαμε τις φιλτραρισμένες ακολουθίες X_I και Q_Q. Έπειτα κάναμε μετασχηματισμό Fourier αυτής και υπολογίσαμε το περιοδόγραμμά της σε γραμμική και ημιλογαριθμική κλίμακα.Παρατηρούμε πως το σήμα μας έχει μετατοπιστεί γύρω απο την συχνότητα F_0 και $-F_0$ ώς απόρροια του πολλαπλασιασμού με τους φορείς.

Σήμα εισόδου μαζί με περιοδόγραμμα:

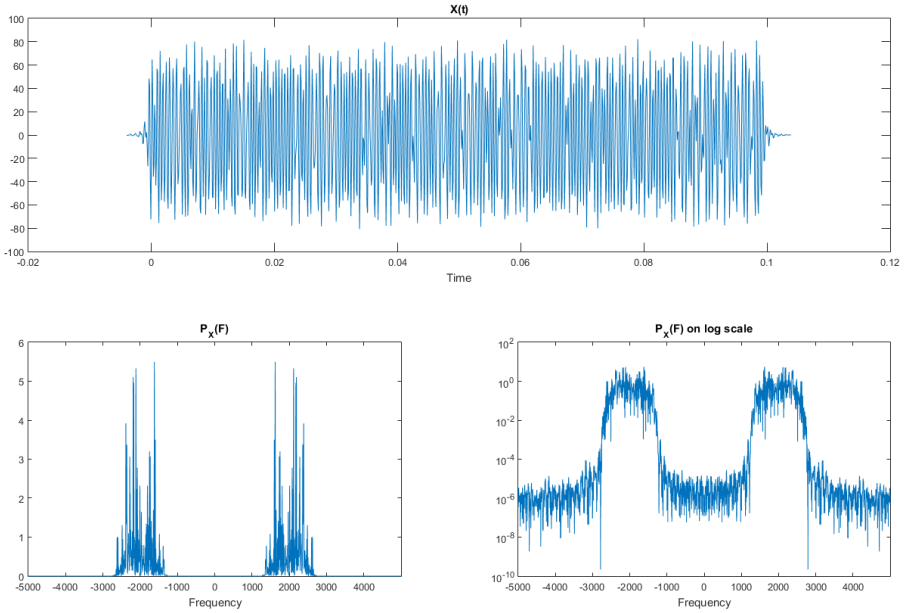

Figure 7: $X(t)$, $P_X(F)$

Κώδικας για την υλοποίηση του ερωτήματος:

```
%% 5.X(t) design
% Entry Signal
X_t = Xi_c + Xq_c;


X_t_F = fftshift(fft(X_t, Nf)*Ts);
P_X_t_F = power(abs(X_t_F),2)./(time(end)-time(1));
```

6. Υποθέτουμε ότι το κανάλι είναι ιδανικό.

7. Στην έξοδο του καναλιού, να προσθέσετε λευκό Gaussian θόρυβο $W(t)$ με διασπορά ίση με

$$\boxed{\sigma_w^2 = \frac{1}{T_S \cdot 10^{\frac{SNR_{db}}{10}}}}$$

Δημιουργήσαμε τον λευκό Gaussian θόρυβο $W(t)$ με χρήση της συνάρτησης randn μέσω της οποίας έχουμε μέση τιμή $m = 0$ και διασπορά 1.Για να έχουμε όμως την τιμή της διασποράς που θέλουμε πολλαπλασιάζουμε με το $\sigma_w^2 = \frac{1}{T_S \cdot 10^{\frac{SNR_{db}}{10}}}$. Επίσης θέσαμε ότι $SNR_{db} = 20$.

```
%% 7. Generate/add White Noise
% SNR = Ex/En = 1/2*sigma^2_N
SNRdb = 20;
sq_sigma = 1/(Ts * 10^(SNRdb/10));
mu = 0;

% White gaussian noise
W_t = randn(length(X_t),1)*sqrt(sq_sigma) + mu;

Y_t = X_t + W_t;
```

8. Να πολλαπλασιάσετε την ενθόρυβη κυματομορφή $Y(t)$ στο δέκτη με τους κατάλληλους φορείς και να σχεδιάσετε τις κυματομορφές που προκύπτουν και τα περιοδογράμματά τους. Τι παρατηρείτε;

Πολλαπλασιάσαμε της $Y_I(t)$ με τον φορέα $cos(2\pi F_0 t)$ και της $Y_Q(t)$ με $-sin(2\pi F_0 t)$. Στην συνέχεια κάνουμε μετασχηματισμό Fourier στην κάθε ακολουθία ξεχωριστά και στην συνέχεια αποτυπώνουμε τις κυματομορφές σε γραμμική και το περιοδόγραμμα σε γραμμική και ημιλογαριθμική κλίμακα.Παρατηρούμε ότι το σήμα μας έχει μετατοπιστεί γύρω απο την συχνότητα $-2F_0$, 0 και $2F_0$ σαν απόρροια του πολλαπλασιασμού με τους φορείς.Ουσιαστικά γίνεται μετατόπιση

των λοβών που ήταν στο $-F_0$ και F_0 κατά F_0 δεξιά και αριστερά με αποτέλεσμα να βγει ένα πιο ενισχυμένο σήμα στο 0.

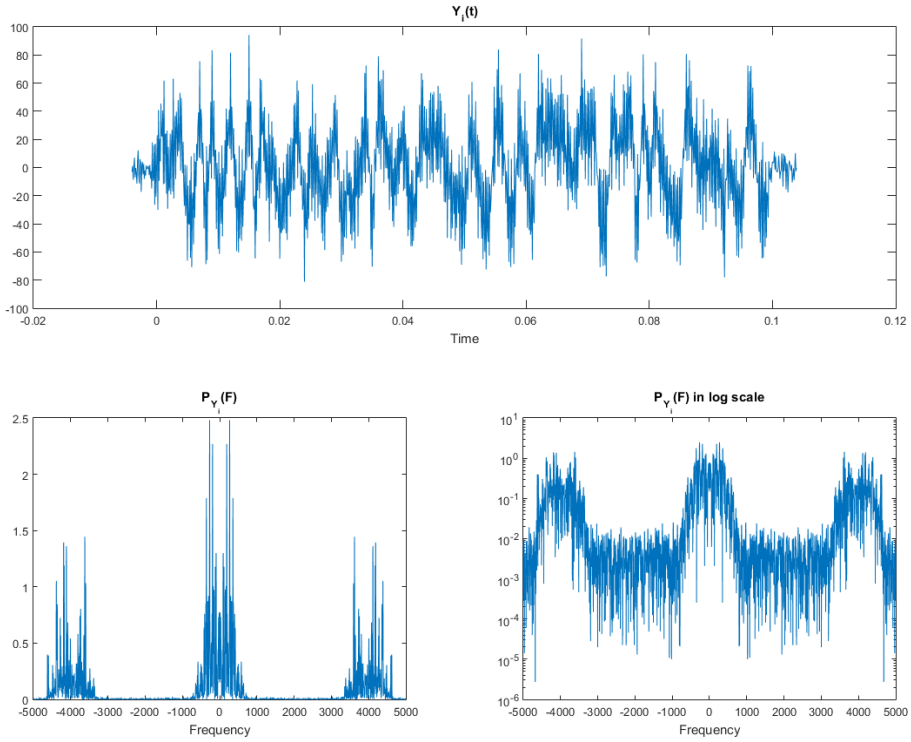

Figure 8: $Y_I(t)$ multiply whith carrier and Periodogramm

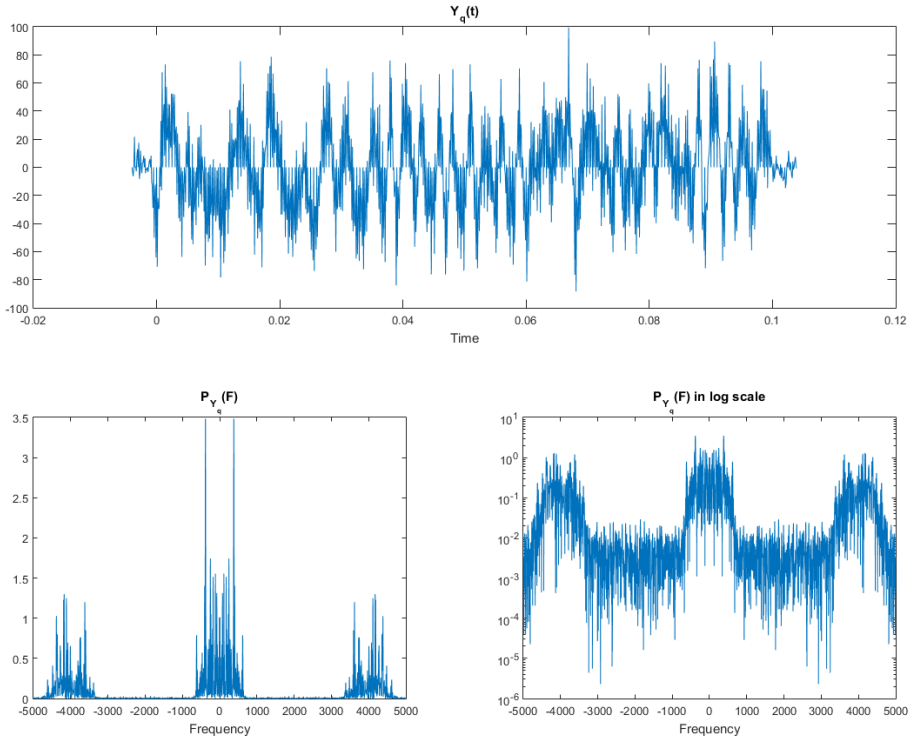

Figure 9: $Y_Q(t)$ multiply whith carrier and Periodogramm

Κώδικας:

```
%% 8.Y(t) * Carries
Ci = (cos(2*pi*F0*time))';
Cq = ((-sin(2*pi*F0*time)))';

Y_t_i = Y_t.*Ci;
Y_t_q = Y_t.*Cq;

Y_F_i = fftshift(fft(Y_t_i, Nf)*Ts);
P_Y_F_i = power(abs(Y_F_i),2)./(time(end)-time(1));

Y_F_q = fftshift(fft(Y_t_q, Nf)*Ts);
P_Y_F_q = power(abs(Y_F_q),2)./(time(end)-time(1));
```

9. Να περάσετε τις κυματομορφές που υπολογίσατε στο προηγούμενο βήμα από τα προσαρμοσμένα φίλτρα. Να σχεδιάσετε τις κυματομορφές που προκύπτουν και τα περιοδογράμματά τους (να θέσετε το σωστό άξονα χρόνου). Τι παρατηρείτε;

Αφού περαστούν οι προηγούμενες κυματομορφές που υπολογίσαμε στο προηγούμενο βήμα απο τα προσαρμοσμένα φίλτρα δηλαδή των SRRC παλμό παρατηρούμε ότι αποκόπτονται οι συχνότητες γύρω απο το $-2F_0$ και $2F_0$. Αυτό συμβαίνει επειδή το φίλτρο μας ειναι χαμηλοπερατό άρα ¨αποκόπτει' τις υψηλές συχνότητες και μένουν μόνο οι συχνότητες γύρω απο το 0 όπως φαίνεται και στο σχήμα.

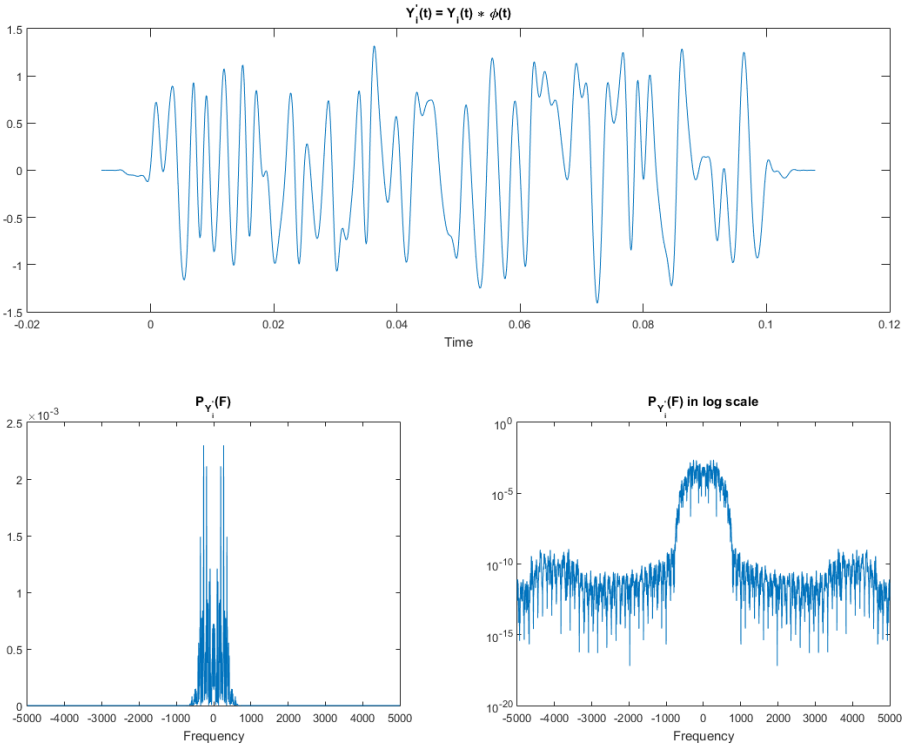

Figure 10: $Y_I(t)$ modified

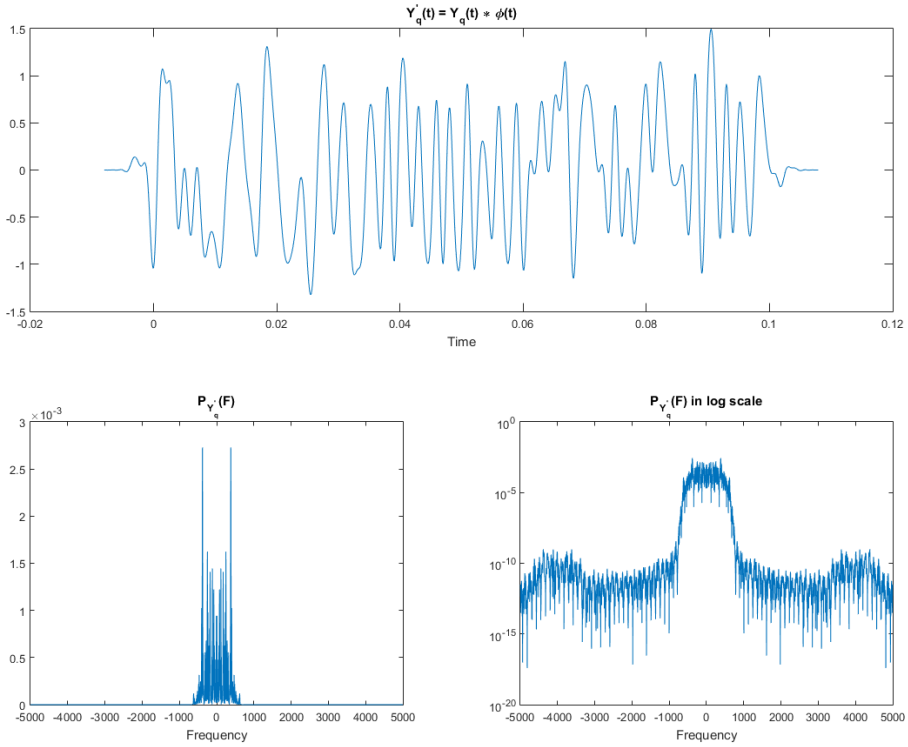

Figure 11: $Y_Q(t)$ modified

Κώδικας:

```
%% 9. Y(t) conv phi(t)
% Convlolution
Y_conv_i = conv(Y_t_i,phi)*Ts;
Y_conv_q = conv(Y_t_q,phi)*Ts;
time = time(1) + t(1) : Ts : time(end) + t(end);
% periodogramms
Y_F_conv_i = fftshift(fft(Y_conv_i , Nf)*Ts);
P_Y_F_i = power(abs(Y_F_conv_i),2)./(time(end)-time(1));

Y_F_conv_q = fftshift(fft(Y_conv_q , Nf)*Ts);
P_Y_F_q = power(abs(Y_F_conv_q),2)./(time(end)-time(1));
```

10. Να δειγματοληπτήσετε την έξοδο των προσαρμοσμένων φίλτρων τις κατάλληλες χρονικές στιγμές και να σχεδιάσετε την ακολουθία εξόδου Y χρησιμοποιώντας την εντολή scatterplot.

Για να δειγματοληπτήσουμε την έξοδο πρώτα βρήκαμε την αρχή και το τέλος κάθε παλμού δηλαδή από 0 μέχρι $N \cdot T$. Στην συνέχεια βρήκαμε το σήμα εξόδου και το κάναμε downsample κατά οερ και το αποτυπώσαμε με την εντολή scatterplot.

Το δειγματοληπτιμένο σήμα εξόδου:

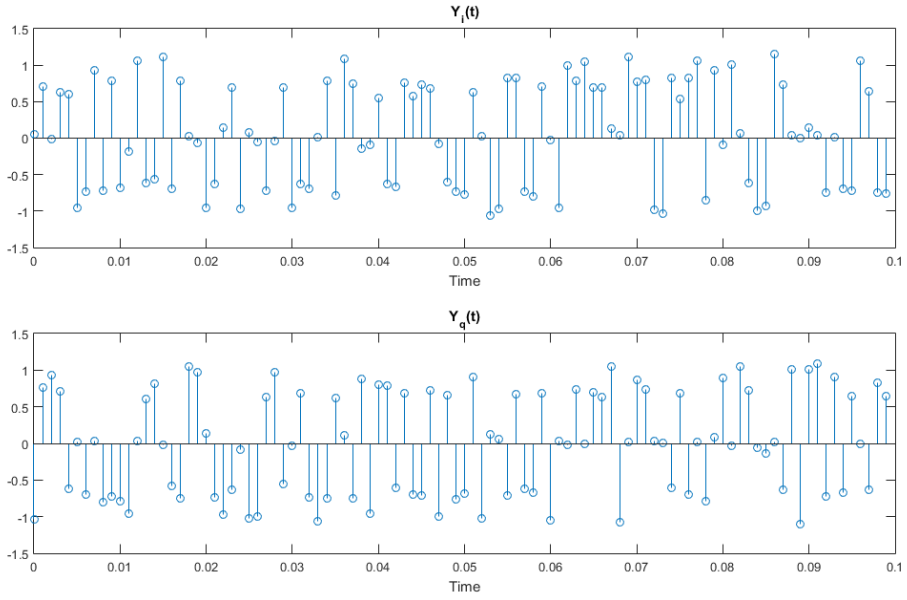

Figure 12: $Y_I[t]$ and $Y_Q[t]$

Το ενθόρηβο σήμα εξόδου:

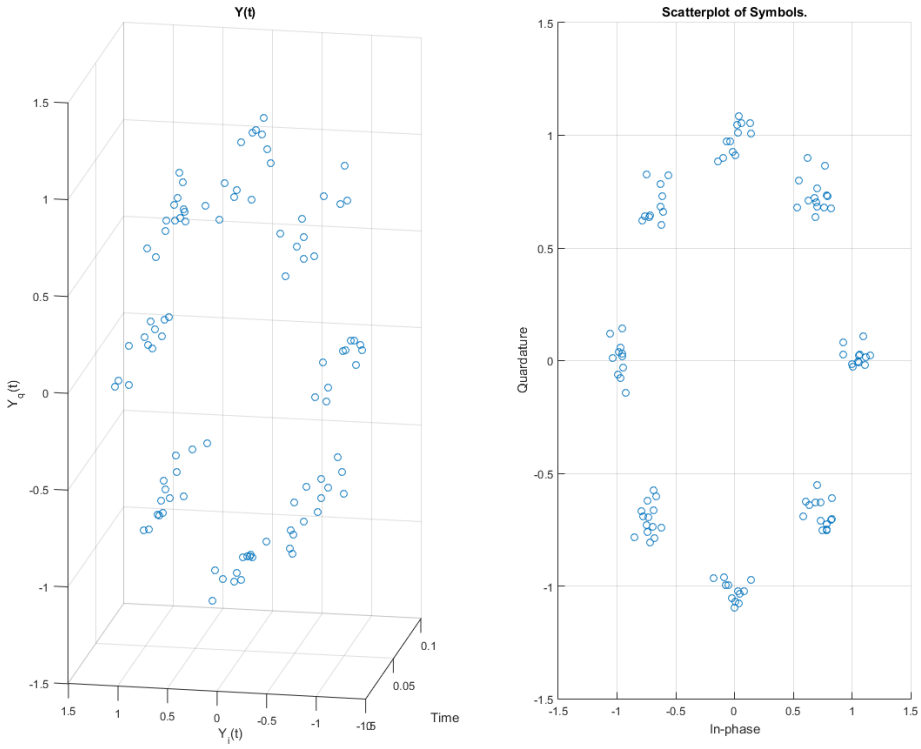

Figure 13: 3D Y(t) and Statter plot of Y(t)

Υλοποίηση:

```
%% 10.Sample the output
% Find the start and end of each pulse
% Yi'(t)
index_first = find(abs(time) < Ts/over);
index_last = find(time >= ((N*T)-Ts), 1);

% The upscaled "tranmeted info"
Yi_t = Y_conv_i(index_first:index_last);
Yq_t = Y_conv_q(index_first:index_last);

Yi_t = downsample(Yi_t, over);
Yq_t = downsample(Yq_t, over);

% Output Signal
```

```
Y(:,1) = Yi_t;
Y(:,2) = Yq_t;

time = 0:T:(N*T)-T;
% Output figures
figure('Name','Output Coordinates');
subplot(2,1,1);
stem(time,Yi_t);
hold on;
% stem(time,Xi);
title('Y_i(t)');
xlabel('Time');
hold on;
subplot(2,1,2);
stem(time,Yq_t);
hold on;
% stem(time,Xi);
title('Y_q(t)');
xlabel('Time');

figure('Name','Output Signal');
subplot(1,2,1);
plot3(time,Y(:,1),Y(:,2),'o');
hold on;
grid on;
hold on;
% stem(time,Xi);
title('Y(t)');
xlabel('Time');
ylabel('Y_i(t)');
zlabel('Y_q(t)');
hold on;
% plot3(time,X(:,1), X(:,2), 'o');
subplot(1,2,2);
scatter(Y(:,1), Y(:,2),'o');
hold on;
grid on;
```

```
xlabel('In-phase');
ylabel('Quardature');
title('Scatterplot of Symbols.');
```

11. Να γράψετε την συνάρτηση

$$function[est\_X, est\_bit\_seq] = detect\_PSK\_8(Y)$$

η οποία

(α) Χρησιμοποιεί τον κανόνα εγγύτερου γείτονα και αποφασίζει για την ακολουθία εισόδου 8-PSK σύμβολο-προς -σύμβολο,

(β) Χρησιμοποιεί την αντίστροφη απεικόνιση Gray, δηλαδή, από σύμβολα σε τριάδες bits, και από την εκτιμώμενη ακολουθία συμβόλων εισόδου υπολογίζει την εκτιμώμενη δυαδική ακολουθία εισόδου.

Συνάρτηση: $detect\_PSK\_8()$

```
function [est_X, est_bit_seq] = detect_PSK_8(Y)
%DETECT_PSK_8: Detect and returns the estimated Symbol
    Sequence
%
%   Estimated X Sequence:
%   Each symbol of the given symbol sequence Y, is stored and
    measured
%   the distance of each Symbol "Center" S.
%   The estimated output Symbol est_X(i) is the minimum
    counted distance.
%
%   Estimated Bit Sequnce:
%   The est_b_seq(i:i+2) is a translated 8_PSK Symbol from
    the estimated
%   X sequence to a 3 bit Gray Code.
%
%   m = 0 -> [000]
%   m = 1 -> [001]
%   m = 2 -> [011]
```

```
%   m = 3 -> [010]
%   m = 4 -> [110]
%   m = 5 -> [111]
%   m = 6 -> [101]
%   m = 7 -> [100]
S = zeros(8,2);
% Set Symbol Centers
for i = 0:1:7
    S((i+1),:) = Coordinates(i);
end

est_X = zeros(length(Y),2);
est_bit_seq = zeros(length(Y)*3,1);


for i = 1:1:length(Y)
    dist = zeros(8,1);

    % store all distances for the given
    for j=1:8
        dist(j) = norm(S(j,:) - Y(i,:));
    end

    [~, index] = min(dist);

    est_X(i,:) = S(index,:);


    if est_X(i,:) == S(1,:)
        est_bit_seq(i*3-2:i*3) = [0;0;0];
    elseif est_X(i,:) == S(2,:)
        est_bit_seq(i*3-2:i*3) = [0;0;1];
    elseif est_X(i,:) == S(3,:)
        est_bit_seq(i*3-2:i*3) = [0;1;1];
    elseif est_X(i,:) == S(4,:)
        est_bit_seq(i*3-2:i*3) = [0;1;0];
    elseif est_X(i,:) == S(5,:)
```

```
        est_bit_seq(i*3-2:i*3) = [1;1;0];
    elseif est_X(i,:) == S(6,:)
        est_bit_seq(i*3-2:i*3) = [1;1;1];
    elseif est_X(i,:) == S(7,:)
        est_bit_seq(i*3-2:i*3) = [1;0;1];
    else
        est_bit_seq(i*3-2:i*3) = [1;0;0];
     end
end

function a = Coordinates(m)
    a = [cos((2*pi*m)/8) sin((2*pi*m)/8)];
end
end
```

12. Να γράψετε συνάρτηση

$$num\_of\_symbol\_errors = symbol\_errors(est\_X, X)$$

η οποία υπολογίζει το πλήθος των σφαλμάτων εκτίμησης συμβόλου.

Συνάρτηση: $symbol\_errors()$

```
function num_of_symbol_errors = symbol_errors(est_X, X)
%   Check the estimated sequence with the given input
%   retern the number of errors

    % Symbol Error Counter
    num_of_symbol_errors=0;
    for i=1:length(X)
        if (est_X(i,:)~= X(i,:))
            num_of_symbol_errors = num_of_symbol_errors + 1;
        end
    end
return;
```

13. Να γράψετε συνάρτηση

$$num\_of\_bit\_errors = bit\_errors(est\_bit\_seq, bit\_seq)$$

η οποία υπολογίζει το πλήθος των σφαλμάτων εκτίμησης bit.

Συνάρτηση: $bit\_errors()$

```
function num_of_bit_errors = bit_errors(est_bit_seq, b_seq)
%BIT_ERRORS Summary of this function goes here
%   Detailed explanation goes here
    num_of_bit_errors=0;
    for i=1:length(est_bit_seq)
        if est_bit_seq(i)~= b_seq(i)
            num_of_bit_errors = num_of_bit_errors + 1;
        end
    end

end
```

B. Θα εκτιμήσουμε την πιθανότητα σφάλματος συμβόλου και bit με χρήση της μεθόδου Monte Carlo.

- Για τα διάφορα SNRdb, τα οποία είναι στο διάστημα $[-2 : 2 : 16]$ και για $K = 1000$ επαναλήψεις, δημιουργούμε τις ακολουθίες από bit και ακολουθόντας τα βήματα του πρώτου μέρους, δημιουργήσουμε το σήμα εξόδου στο οποίο, στην κάθε περίπτωση, έχει προστεθεί Gaussian White Noise με διαφορετική τυπική απόκλιση κάθε φορά.

- Για να υπόλογίσουμε τις πειραματικές πιθανότητες σφάλματος συμβόλου και bit κάαμε τα εξής:

```
% B
clear;
clc;
close all;

% Set starcting info
N = 100;
T = 0.001;
over = 10;
Ts = T/over;
a = 0.5;
A = 4;
Nf = 4096;
Fs = 1/Ts; % Sampling Frequency
f_axis = linspace(-Fs/2,(Fs/2-Fs/Nf), Nf);
K = 1000;
F0 = 2000;
% Generate phi[t]
[phi, t] = srrc_pulse(T, over, A, a);
% Symbol transmit duration
time = 0:T:((N*T)-T);

SNRDB = -2 : 2 : 16;
P_Symbol = zeros(1,length(SNRDB));
```

```
P_Bit = zeros(1,length(SNRDB));
Upper_bound_symbol = zeros(1,length(SNRDB));
Upper_bound_bit = zeros(1,length(SNRDB));
% For each SRNdb
for i = 1:length(SNRDB);
    SNRdb = SNRDB(i);
    max_num_of_symbol_errors = 0;
    max_num_of_bit_errors = 0;

    sq_sigma = 1/(Ts * (10^(SNRdb/10)));
    sq_sigma_N = (Ts*sq_sigma)/2;
%   Upper_bound_symbol(i) = 2*Q(sqrt(2*(10^(SNRdb/10)))*sin(
   pi/8));
    Upper_bound_symbol(i) = 2*Q((1/sqrt(sq_sigma_N))*sin(pi
   /8));
    Upper_bound_bit(i) = Upper_bound_symbol(i)/3;
    for k = 1:K
        % 3N bits generation
        b_seq = (sign(randn((3*N), 1)) + 1)/2;
        % 8PSK Symbols
        X = bits_to_PSK_8(b_seq);
        Xi = X(:,1);
        Xq = X(:,2);
        % Upsample coordinates
        X_upsample=(1/Ts) .* upsample(X, over);
        Xi_upsample = X_upsample(:,1);
        Xq_upsample = X_upsample(:,2);
        X_time = 0:Ts:(N*T)-Ts;
        % Signal formatting
        Xi_conv = conv(Xi_upsample, phi)*Ts;
        Xq_conv = conv(Xq_upsample, phi)*Ts;
        time = t(1) + X_time(1):Ts:t(end) + X_time(end);
        % Modulate with carries
        Ci = (2*cos(2*pi*F0*time))';
        Cq = ((-2)*sin(2*pi*F0*time))';

        Xi_c = Xi_conv .* Ci;
```

```
Xq_c = Xq_conv .* Cq;
% Transmeted signal
X_t = Xi_c + Xq_c;
% Set and generate Noise
mu = 0;
W_t = randn(length(X_t),1)*sqrt(sq_sigma) + mu;
% Output signal
Y_t = X_t + W_t;

time = t(1) + X_time(1):Ts:t(end) + X_time(end);
% Modulate with carries
Ci = (cos(2*pi*F0*time))';
Cq = ((-sin(2*pi*F0*time)))';

Y_t_i = Y_t.*Ci;
Y_t_q = Y_t.*Cq;
% Signal formatting
Y_conv_i = conv(Y_t_i,phi)*Ts;
Y_conv_q = conv(Y_t_q,phi)*Ts;
time = time(1) + t(1) : Ts : time(end) + t(end);
% Sample the output
index_first = find(abs(time) < Ts/over);
index_last = find(time >= ((N*T)-Ts), 1);

Yi_t = Y_conv_i(index_first:index_last);
Yq_t = Y_conv_q(index_first:index_last);
% Sample the output
Yi_t = downsample(Yi_t, over);
Yq_t = downsample(Yq_t, over);
% Output Sampled Signal
Y(:,1) = Yi_t;
Y(:,2) = Yq_t;
time = 0:T:(N*T)-T;
% Estinated Symbol and bit Sequenced
[est_X, est_bit_Seq] = detect_PSK_8(Y);

num_of_symbol_errors = symbol_errors(est_X, X);
```

```
        max_num_of_symbol_errors = max_num_of_symbol_errors +
    num_of_symbol_errors;

        num_of_bit_errors = bit_errors(est_bit_Seq, b_seq);
        max_num_of_bit_errors = max_num_of_bit_errors +
   num_of_bit_errors;
    end
%     disp(SNRdb)
    P_Symbol(i) = max_num_of_symbol_errors/(K*100); % K->
   peiramata, N->Symbols
%     disp(P_Symbol)
    P_Bit(i) = max_num_of_bit_errors/(K*(300));
%     disp(P_Bit)


end

figure('Name', 'Symbol_Error - SNRdb')
semilogy(SNRDB,P_Symbol);
hold on;
semilogy(SNRDB,Upper_bound_symbol);
hold on;
grid on;
xlabel('SNR in db');
ylabel('Symbol Error Rate for 8_PSK')
legend('Experimental', 'Upper Bound')

figure('Name', 'Bit_Error - SNRdb')
semilogy(SNRDB,P_Bit);
hold on;
semilogy(SNRDB,Upper_bound_bit);
hold on;
grid on;
xlabel('SNR in db');
ylabel('Bit Error Rate for 8_PSK')
legend('Experimental', 'Upper Bound')
```

- Για τον υπολογισμό του Upper bound για την πιθανότητητα σφάλματος συμβόλου κάναμε χρήση του τύπου:

$$\boxed{P(E_{symbol}) \leq 2 \cdot Q(\frac{1}{\sigma_N} \cdot \sin(\frac{\pi}{8})) \text{ με } \sigma_N = \frac{T_S \cdot \sigma_W^2}{2}}$$

Το οποίο συμμένει πως η πειραματική πιθανότητα θα πρέπει να βρίσκετε ¨κάτω' από το παραπάνω όριο.

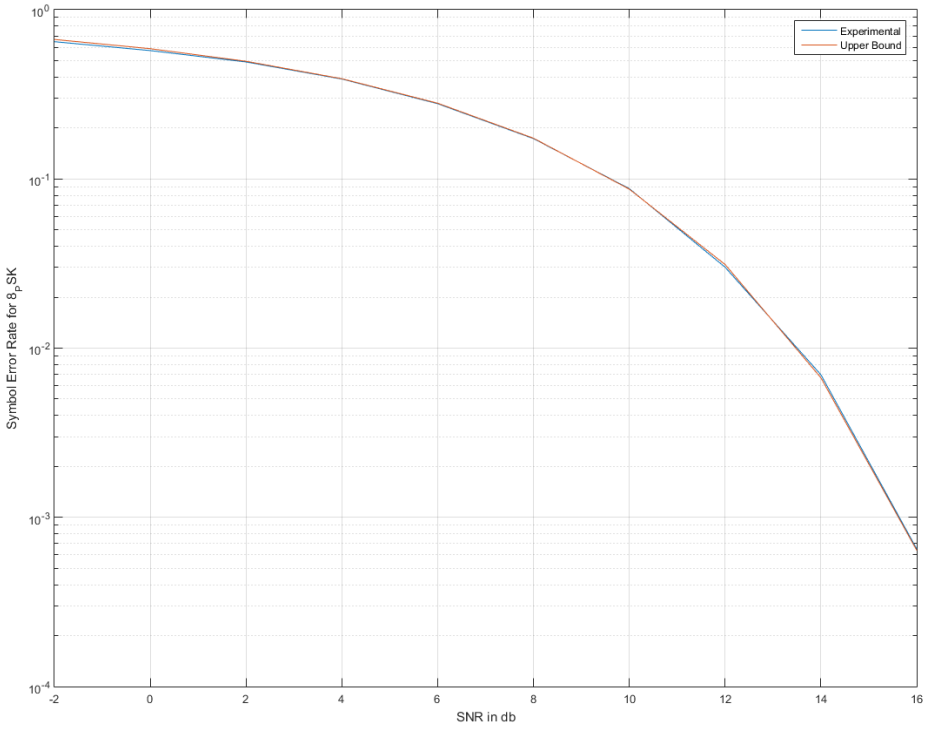

Figure 14: Experimental and Theoretical Error rates

Στο παραπάνω κοινό Semilogy παρατηρούμε ότι όσο αυξάνετε η τιμή του SNRdb, δηλάδή του SNR η πειραματική πιθανότητηα σφάλματος συμβόλου συμπίπτει μετ την θεωριτική τιμή καθώς μειώνεται η πιθανότητα λάθους.

- Για τον υπολογισμό του Upper bound για την πιθανότητητα σφάλματος bit κάναμε χρήση του τύπου:

$$\boxed{P(E_{bit}) \leq \frac{P(E_{symbol})}{log_{10}(M)} = \frac{P(E_{symbol})}{log_{10}(8)} = \frac{P(E_{symbol})}{3}}$$

Όμοια η πειραματική θα πρέπει να βρίσκετε ¨κάτω' από το όριο.

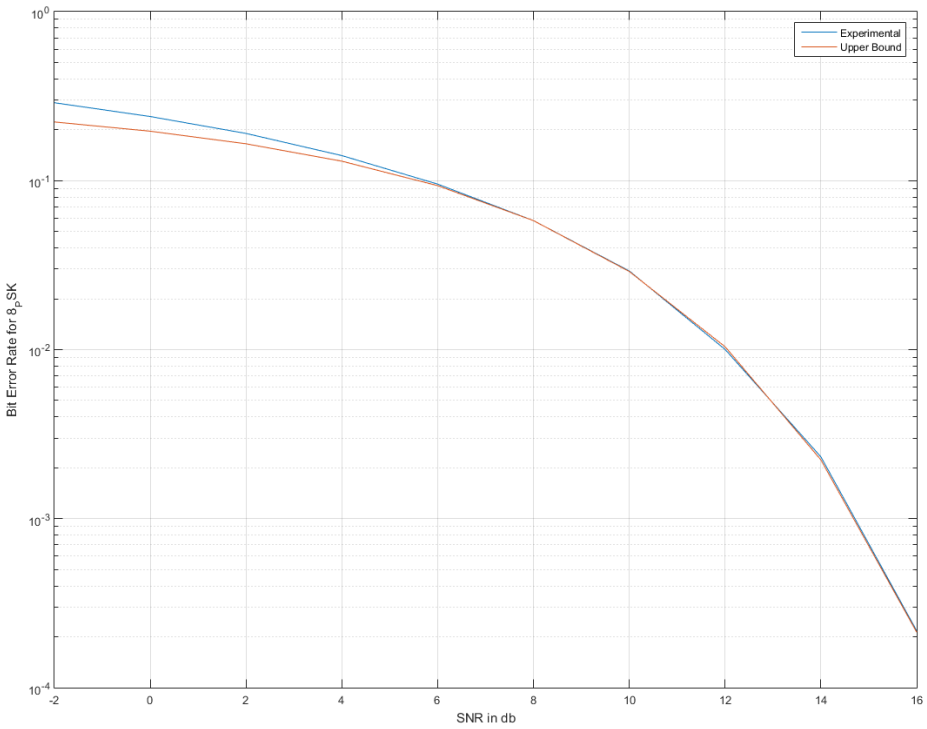

Figure 15: Experimental and Theoretical Error rates

Βέβαια, όπως παρατηρούμε για μικρές τιμές του SNR, το πειραματικό Rate ξεπερνά το θεωρητικό όριο. Αυτό συμβάινει μιας και ο τύπος που χρησιμοποιήσαμε επαληθεύεται για μεγάλες τιμές SNR.