# ΤΗΛΕΠΙΚΟΙΝΩΝΙΑΚΑ ΣΥΣΤΗΜΑΤΑ Ι

## Αναφορά 3ης Εργαστηριακής Άσκησης

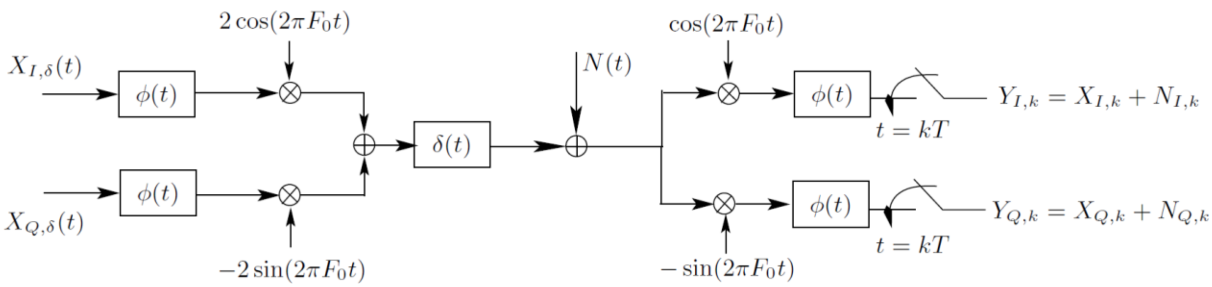
«ΜΕΛΕΤΗ ΤΗΛ. ΣΥΣΤΗΜΑΤΟΣ ΜΕ ΔΙΑΜΟΡΦΩΣΗ 8-PSK»

**Μιχάλης Γαλάνης** 201603003

# ΠΕΡΙΕΧΟΜΕΝΑ

# ΠΕΡΙΛΗΨΗ

Στη 3η εργαστηριακή άσκηση προσωμοιώνουμε ένα τηλεπικοινωνιακό σύστημα που χρησιμοποιεί διαμόρφωση 8-PSK και το μελετάμε ως προς την απόδοσή του και τα σφάλματα που παράγονται ανάλογα με το signal to noise ratio (SNR).




Για ακόμη μια φορά, χρησιμοποιούνται στον κώδικα οι παρακάτω ανώνυμες συναρτήσεις για την απλοποίηση του, οι οποίες έχουν εξηγηθεί σε προηγούμενες ασκήσεις:

## Τμήμα Κώδικα

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% BASIC INFORMATION                                   %
% Course: Telecommunication Systems I - Excersize 3   %
% Deadline: 20-Dec-18                                 %
% FullName: Mixalis Galanis                           %
% Academic ID: 2016030036                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Clearing things up
close all
clearvars

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Useful Functions                                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Plotting Preperation Functions
set_diag_labels = @(tit, xlab, ylab){title(tit); xlabel(xlab); ylabel(ylab);};
set_diag_layout = @(fig, rows, cols, selected){figure(fig); subplot(rows,cols,selected)};
%Some Useful Plotting Functions (continuous & discrete)
c_plot = @(t, f){ plot(t,f);};
c_semilogy = @(t, f){ semilogy(t,f);};
d_plot = @(t, f){ stem(t,f);};
%Some Useful Signal Processing Functions
t_conv = @(t1, t2, dt) (t1(1) + t2(1)):dt:(t1(end) + t2(end));  %Convolution time
gen_N_bits = @(N) ((sign(randn(N,1)) + 1)/2);                  %Generates N Random Bits
F_T = @(X,Nf,Ts) (abs(fftshift(fft(X,Nf)*Ts)));                %Fourier Transform of X
P_X = @(t_X, X,Nf,Ts) ((F_T(X,Nf,Ts).^2)/(length(t_X)*Ts));     %Periodogram of X
```

# A.1

Δημιουργούμε μια ακολουθία ισοπίθανων bits (**3N**) με **N = 50**;

### Τμήμα Κώδικα

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A1 - Creating N Bits Sequence                            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
N = 50;
bit_seq = gen_N_bits(3*N);
```

# A.2

Κατασκεύασα μια συνάρτηση bits_to_PSK_8 όπου κατασκευάζει μια ακολουθία συμβόλων 8-PSK από μια δυαδική ακολουθία. Αυτο επιτυγχάνεται ομαδοποιώντας τα bit ανά 3άδες και υπολογίζοντας την τιμή:

$$X_n = \begin{bmatrix} X_{I,n} \\ X_{Q,n} \end{bmatrix}$$

Με κάθε διάνυσμα $X_n$ να παίρνει τιμές από το αλφάβιτο 8-PSK $\{x_0, \dots, x_m\}$ με:

$$x_m = \begin{bmatrix} A\cos\dfrac{\pi m}{4} \\ X_{Q,n}\sin\dfrac{\pi m}{4} \end{bmatrix}$$

Επιλέχθηκε η ακόλουθη διαδρομή για την κωδικοποίηση Gray:

$$000 \to 001 \to 011 \to 010 \to 110 \to 111 \to 101 \to 100$$

Επιλέχθηκε και το πλάτος ίσο με 1 καθώς πρέπει να ισχύει η συνθήκη:

$$\mathcal{E}[\|\mathbf{X}_n\|_2^2] = \mathcal{E}[X_{I,n}^2] + \mathcal{E}[X_{Q,n}^2] = 1.$$

### Τμήμα Κώδικα (Κύριο Μέρος)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A2 - Generating 8-PSK bits                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A = 1; %TODO Figure out A's value
bit_8_psk = bits_to_PSK_8(bit_seq, A); %N length
```

## Τμήμα Κώδικα (Συνάρτηση)

```
function [psk_bits] = bits_to_PSK_8(b, A)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% psk_bits = bits_to_PSK_8(b)                                                   %
% OUTPUT                                                                        %
%      psk_bits: a vector of 8-PSK symbols                                      %
% INPUT                                                                         %
%      b: sequence (vector) of binary bits                                      %
%      A: Amplitude                                                             %
%                                                                               %
%    M. Galanis, Dec. 2018                                                      %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
L = length(b)/3;
psk_bits = zeros(L,2);
Xn = [0 0];
n = 1;

for i = 1:3:length(b)
    if ((b(i) == 0) && (b(i+1) == 0) && (b(i+2) == 0))
        m = 0;
    elseif ((b(i) == 0) && (b(i+1) == 0) && (b(i+2) == 1))
        m = 1;
    elseif ((b(i) == 0) && (b(i+1) == 1) && (b(i+2) == 1))
        m = 2;
    elseif ((b(i) == 0) && (b(i+1) == 1) && (b(i+2) == 0))
        m = 3;
    elseif ((b(i) == 1) && (b(i+1) == 1) && (b(i+2) == 0))
        m = 4;
    elseif ((b(i) == 1) && (b(i+1) == 1) && (b(i+2) == 1))
        m = 5;
    elseif ((b(i) == 1) && (b(i+1) == 0) && (b(i+2) == 1))
        m = 6;
    elseif ((b(i) == 1) && (b(i+1) == 0) && (b(i+2) == 0))
        m = 7;
    end
    Xn(1) = A*cos((pi*m)/4);
    Xn(2) = A*sin((pi*m)/4);
    psk_bits(n, 1) = Xn(1);
    psk_bits(n, 2) = Xn(2);
    n = n + 1;
end
return
```

# A.3

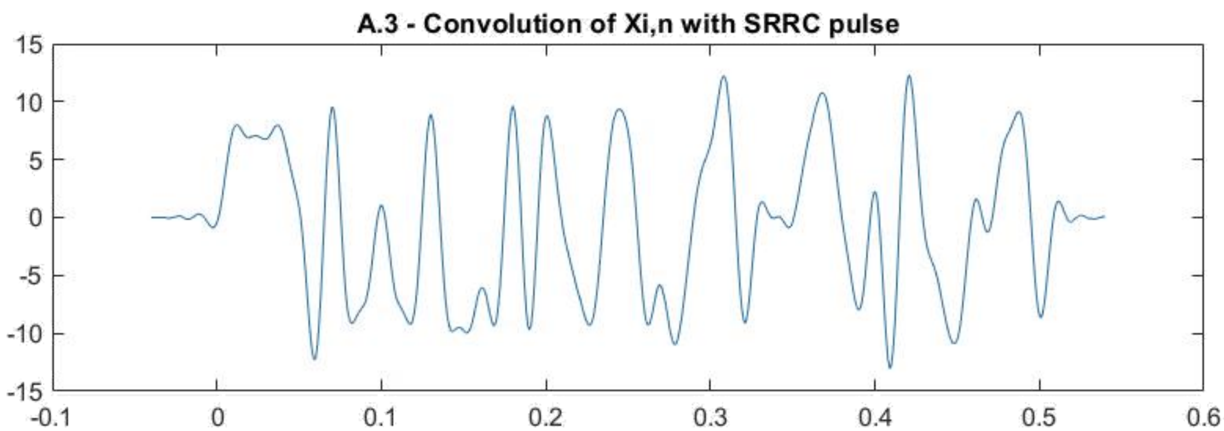
Σε αυτό το ερώτημα περνάμε τις ακολουθίες $X_{I,n}$ και $X_{Q,n}$ από SRRC φίλτρα μορφωποίησης, δηλαδή τις συνελίσσουμε με τη $\varphi(t)$ και ύστερα απεικονίζουμε τις κυματομορφές εξόδου και τα περιοδογράμματα τους.

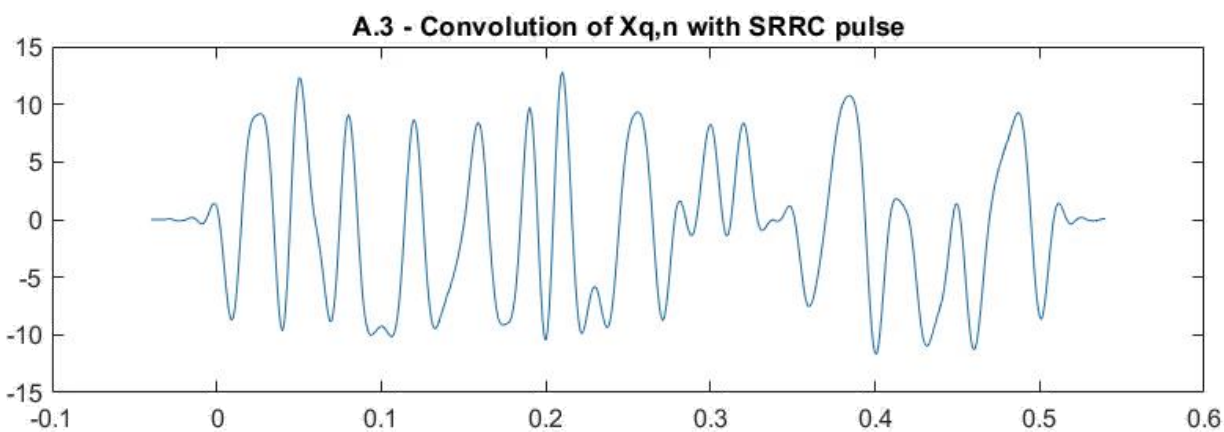
## Τμήμα Κώδικα

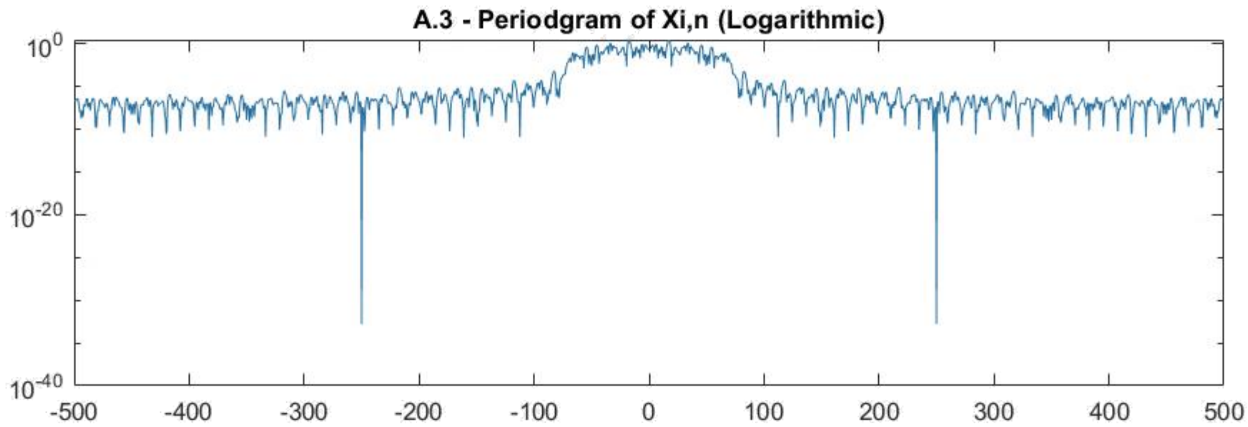
```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A3 - Filtering Xi & Xq with SRRC Pulses                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Settings for plotting the diagrams
r = 2; c = 2; fig = 1; sub_fig = 1;

Nf = 2^11;
DT = 1;

Xi = squeeze(bit_8_psk(:,1));
t_Xi = 0:length(Xi);
Xq = squeeze(bit_8_psk(:,2));
t_Xq = 0:length(Xq);

%Building up parameters for srrc_pulse function
T = 0.01;       %Nyquist parameter (>0)
over = 10;      %Oversampling factor (>0)
Ts = T/over;    %Sampling period (>0)
Fs = 1/Ts;      %Sampling frequency (>0)
Hd = 4;         %Half duration of the pulse (>0)
a = 0.5;        %Roll-off factor(0<a<1)

%Calling srrc_pulse function to store phi and t variables
[phi, t_phi] = srrc_pulse(T, Ts, Hd, a);

%Convoluting Xi, Xq with phi
Xi_delta = Fs * upsample(Xi,over);            %Inserts zeros in between bits
Xq_delta = Fs * upsample(Xq,over);            %Inserts zeros in between bits

Xi_conv = conv(Xi_delta, phi)*Ts;
Xq_conv = conv(Xq_delta, phi)*Ts;

t_Xi_conv = t_conv(t_phi, t_Xi, Ts);
t_Xq_conv = t_conv(t_phi, t_Xq, Ts);

%Calculating Periodgrams
F_ES = (-Fs/2) :(Fs/Nf): (Fs/2 - Fs/Nf);
PXi = P_X(t_Xi_conv, Xi_conv,Nf,Ts);
PXq = P_X(t_Xq_conv, Xq_conv,Nf,Ts);

%Plotting output diagrams
%Xi_conv
set_diag_layout(fig, r, c, sub_fig);
c_plot(t_Xi_conv,Xi_conv);
set_diag_labels("A.3 - Convolution of Xi,n with SRRC pulse","t(s)","");
grid on;
```

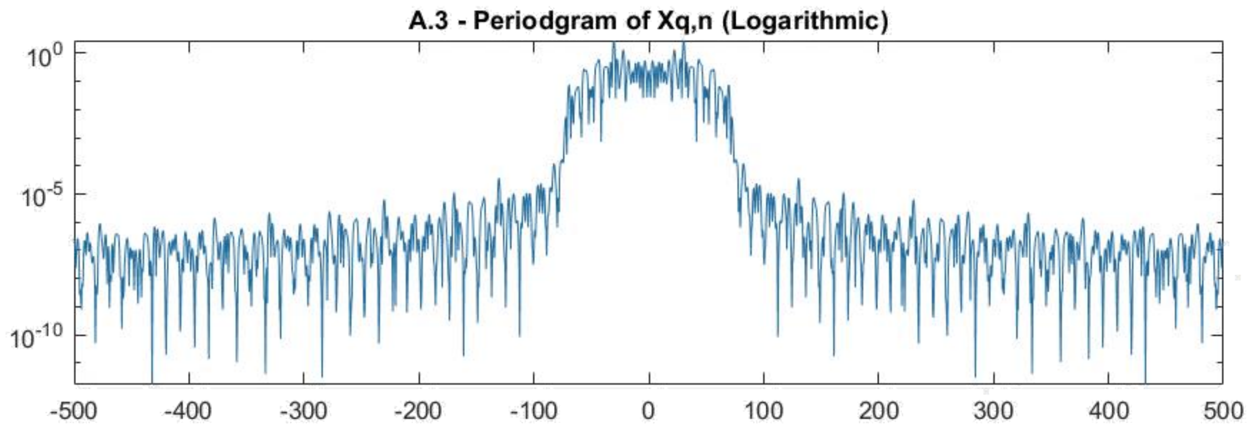
```
%Xq_conv
sub_fig = sub_fig + 1;
set_diag_layout(fig, r, c, sub_fig);
c_plot(t_Xq_conv,Xq_conv);
set_diag_labels("A.3 - Convolution of Xq,n with SRRC pulse","t(s)","");
grid on;
%PXi
sub_fig = sub_fig + 1;
set_diag_layout(fig, r, c, sub_fig);
c_semilogy(F_ES,PXi);
set_diag_labels("A.3 - Periodgram of Xi,n (Logarithmic)","F [Hz]","");
grid on;
%PXq
sub_fig = sub_fig + 1;
set_diag_layout(fig, r, c, sub_fig);
c_semilogy(F_ES,PXq);
set_diag_labels("A.3 - Periodgram of Xq,n (Logarithmic)","F [Hz]","");
grid on;
```

## Διαγράμματα






# A.4

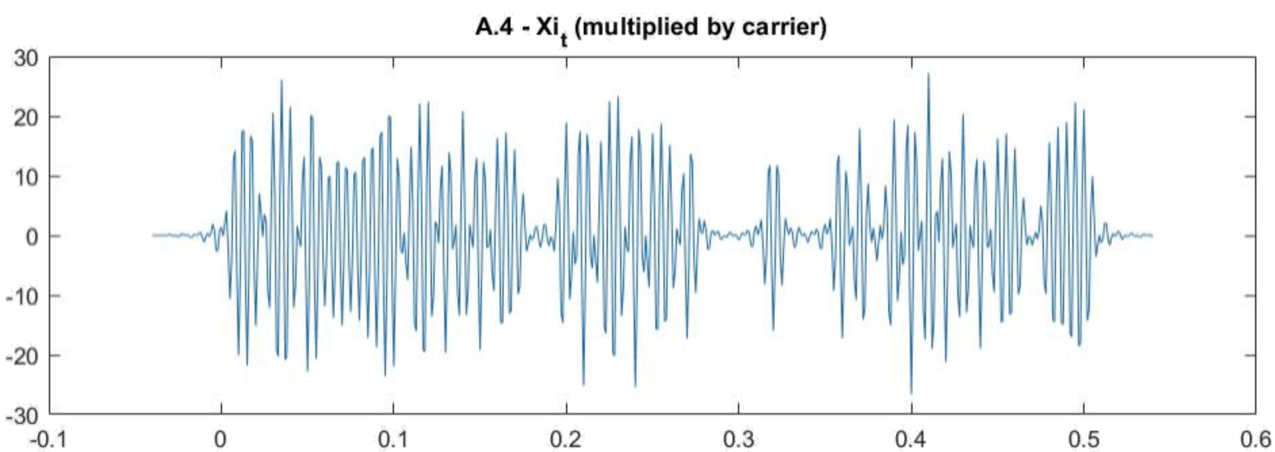
Εδώ μας ζητήθηκε να πολλαπλασιάσουμε τις κυματομορφές από τους αντίστοιχους φορείς, που όπως βλέπουμε από το αρχικό μας σχήμα, είναι $2\cos 2\pi F_0 t$ για το $X_{I,n}$ και $-2\sin 2\pi F_0 t$ για το $X_{Q,n}$. Με αυτή τη διαδικασία μετατοπίζουμε τα σήματά σε μια ζώνη μεγαλύτερων συχνοτήτων προκειμένου να μπορούν να μεταδωθούν.

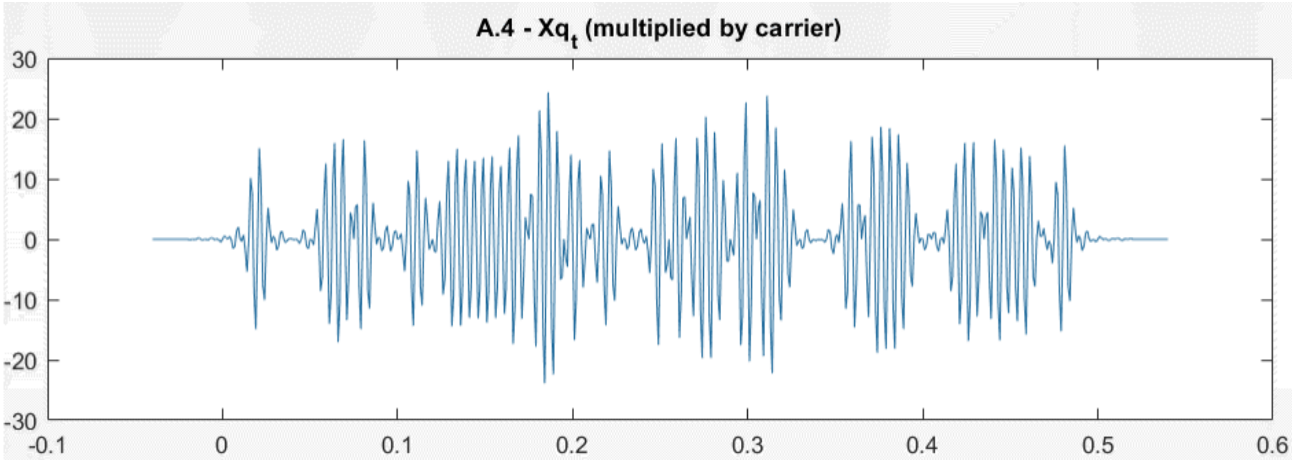
## Τμήμα Κώδικα

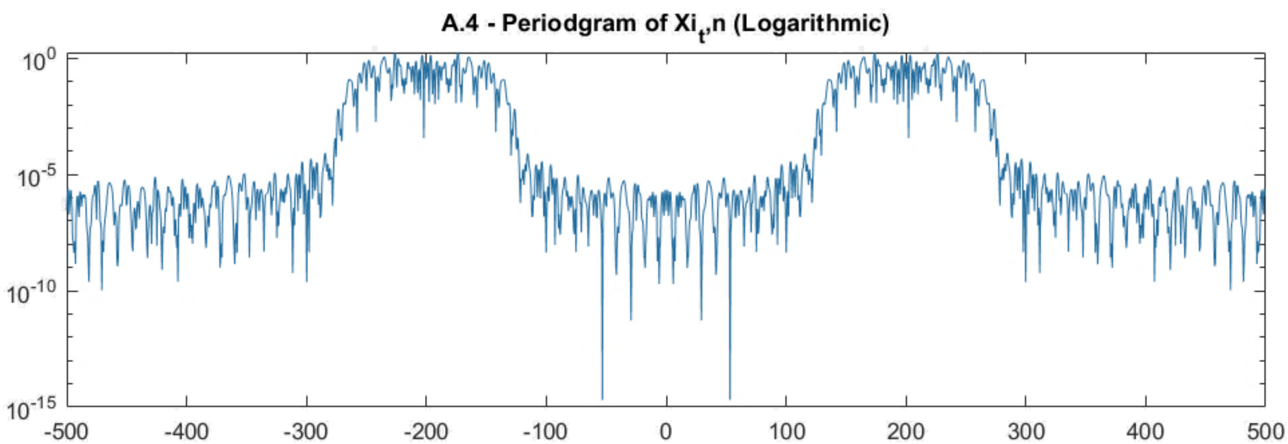
```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A4 - Multiplication of Xi, Xq by carrier                 %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fig = fig + 1; sub_fig = 1; %New Figure
F0 = 200;

%Calculating Multiplication by Carrier
Xi_t = Xi_conv .* (2*cos(2*pi*F0*t_Xi_conv));
Xq_t = Xq_conv .* ((-2)*sin(2*pi*F0*t_Xq_conv));
```

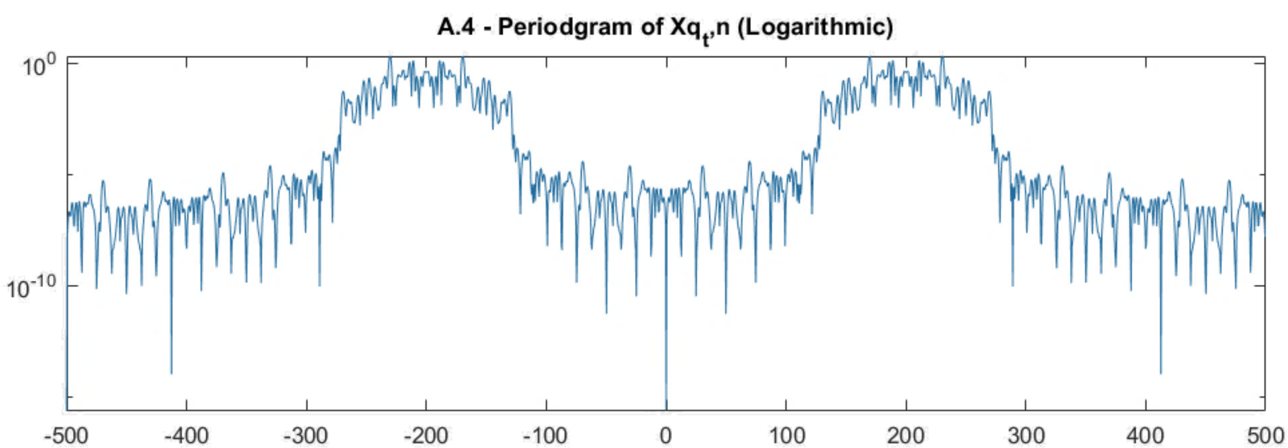
```
%Calculating Periodgrams
PXi_t = P_X(t_Xi_conv, Xi_t,Nf,Ts);
PXq_t = P_X(t_Xq_conv, Xq_t,Nf,Ts);

%Plotting output diagrams
%Xi_t
set_diag_layout(fig, r, c, sub_fig);
c_plot(t_Xi_conv,Xi_t);
set_diag_labels("A.4 - Xi_t (multiplied by carrier)","t(s)","");
grid on;
%Xq_t
sub_fig = sub_fig + 1;
set_diag_layout(fig, r, c, sub_fig);
c_plot(t_Xq_conv,Xq_t);
set_diag_labels("A.4 - Xq_t (multiplied by carrier)","t(s)","");
grid on;
%PXi
sub_fig = sub_fig + 1;
set_diag_layout(fig, r, c, sub_fig);
c_semilogy(F_ES,PXi_t);
set_diag_labels("A.4 - Periodgram of Xi_t,n (Logarithmic)","F [Hz]","");
grid on;
%PXq
sub_fig = sub_fig + 1;
set_diag_layout(fig, r, c, sub_fig);
c_semilogy(F_ES,PXq_t);
set_diag_labels("A.4 - Periodgram of Xq_t,n (Logarithmic)","F [Hz]","");
grid on;
```

## Διαγράμματα




A.4 - $Xq_t$ (multiplied by carrier)
30
20
10
0
-10
-20
-30
-0.1
0
0.1
0.2
0.3
0.4
0.5
0.6


A.4 - Periodgram of $Xi_t$,n (Logarithmic)
$10^0$
$10^{-5}$
$10^{-10}$
$10^{-15}$
-500
-400
-300
-200
-100
0
100
200
300
400
500


A.4 - Periodgram of $Xq_t$,n (Logarithmic)
$10^0$
$10^{-10}$
-500
-400
-300
-200
-100
0
100
200
300
400
500

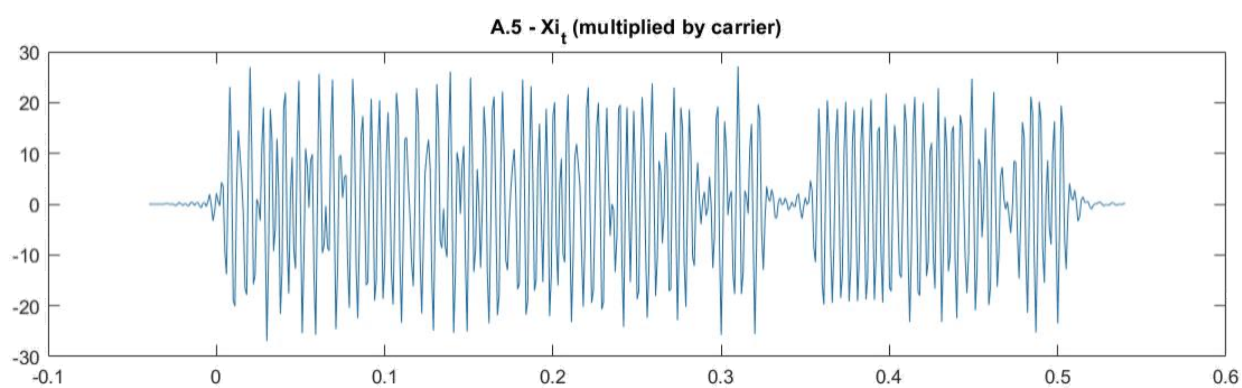
# A.5

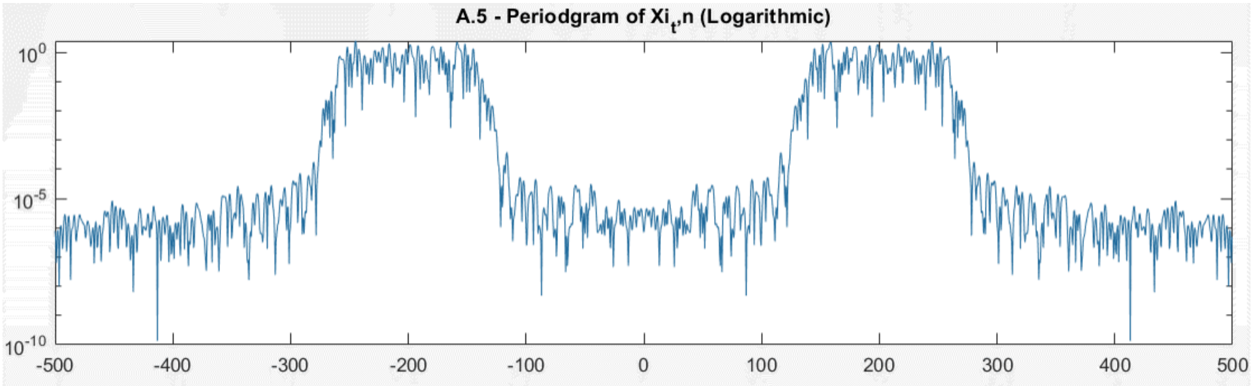
Συνεχίζουμε υπολογίζοντας την είσοδος του καναλιού $X(t)$ η οποία ισοδυναμεί με το άθροισμα των σημάτων **Xi_t** και **Xq_t**. Συγκεκριμένα έχουμε:

## Τμήμα Κώδικα

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A5 - Multiplication of Xi, Xq by carrier                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fig = fig + 1; sub_fig = 1; r = 2; c = 2; %New Figure

X = Xi_t + Xq_t;

PX = P_X(t_Xi_conv, X,Nf,Ts);

%Plotting output diagrams
%X
set_diag_layout(fig, r, c, sub_fig);
c_plot(t_Xi_conv,X);
set_diag_labels("A.5 - Xi_t (multiplied by carrier)","t(s)","");
grid on;
%PX
sub_fig = sub_fig + 1;
set_diag_layout(fig, r, c, sub_fig);
c_semilogy(F_ES,PX);
set_diag_labels("A.5 - Periodgram of Xi_t,n (Logarithmic)","F [Hz]","");
grid on;
```

## Διαγράμματα






## A.6

Υποθέτουμε ότι το κανάλι είναι ιδανικό, συνεπώς το σήμα παραμένει αμετάβλητο (χωρίς αλλοιωση). Το σύστημα έχει δηλαδή κρουστική απόκριση $\delta(t)$.

## A.7

Προσθέτουμε τωρα λευκό θόρυβο Gaussian, έτσι ώστε να προσωμοιώσουμε «ρεαλιστικά» το σύστημα, το οποίο έχει διασπορά:

$$\sigma_w^2 = F_s \cdot 10^{\left(-\frac{SNR}{10}\right)}$$

Αναθέτουμε την ενδεικτική τιμή 10 για SNR:

### Τμήμα Κώδικα

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A7 - White Gassuan Noise                                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
SNR = 10; %Setting up signal to noise ratio
sw = Fs * (10^(-(SNR/10)));
wn= randn(1,length(X)) .* sqrt(sw);
X_wn = X + wn;
```

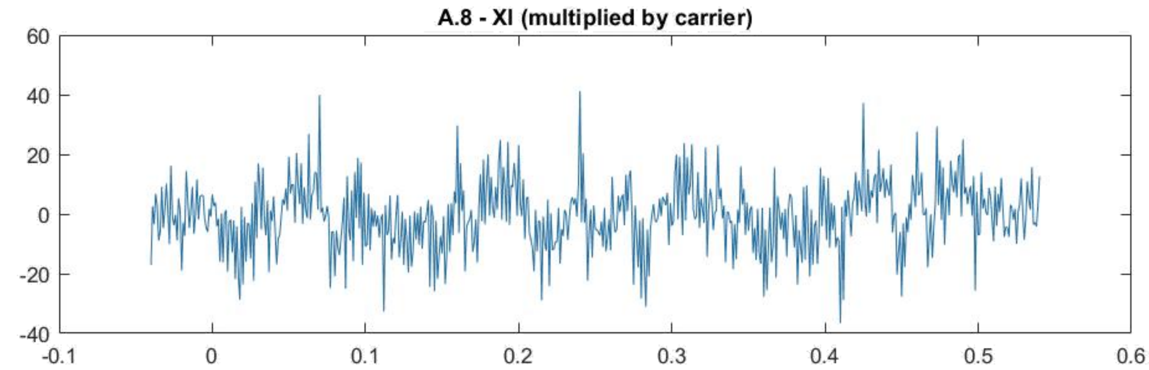
# A.8

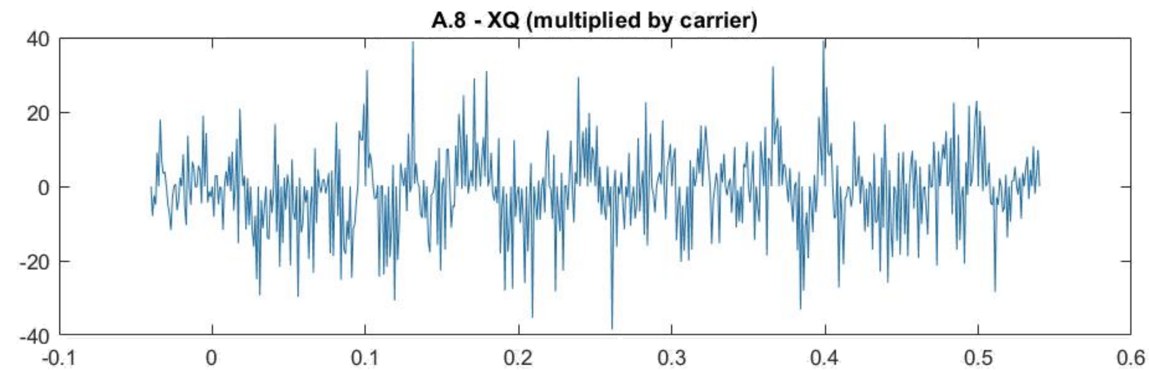
Παρόμοια με το ερώτημα A.4, αποδιαμορφώνουμε το σήμα πολλαπλασιάζοντας την ενθόρυβη κυματομορφή με τους αντίστοιχους φορείς και απεικονίζοντας τα αποτελέσματα:

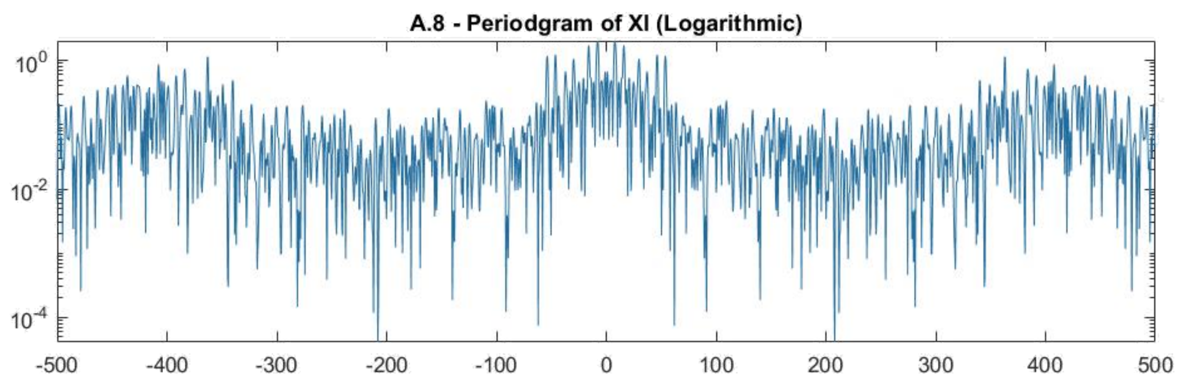
## Τμήμα Κώδικα

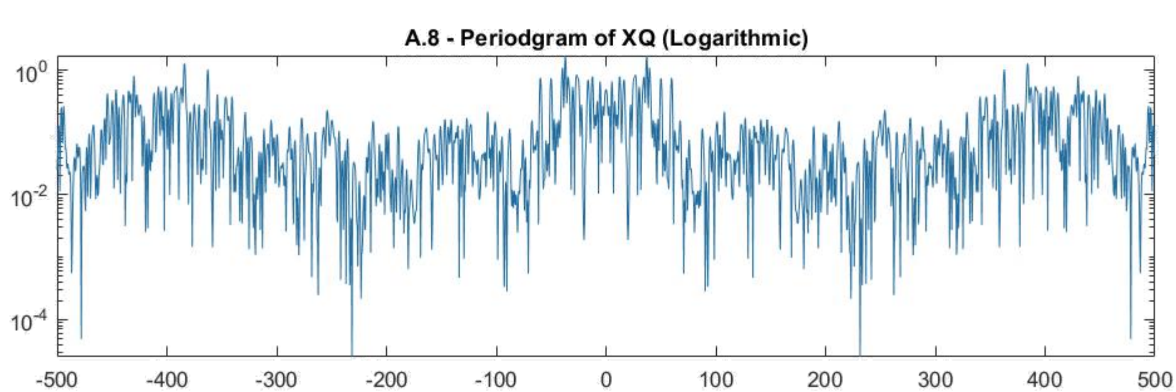
```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A8 - Multiplication of X by carrier                      %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fig = fig + 1; sub_fig = 1; r = 2; c = 2; %New Figure

XI = X .* cos(2*pi*F0*t_Xi_conv);
XQ = X .* cos(2*pi*F0*t_Xq_conv);

%Calculating Periodgrams
PXI = P_X(t_Xi_conv, XI,Nf,Ts);
PXQ = P_X(t_Xq_conv, XQ,Nf,Ts);

%Plotting output diagrams
%XI
set_diag_layout(fig, r, c, sub_fig);
c_plot(t_Xi_conv,XI);
set_diag_labels("A.8 - XI (multiplied by carrier)","t(s)","");
grid on;
%XQ
sub_fig = sub_fig + 1;
set_diag_layout(fig, r, c, sub_fig);
c_plot(t_Xq_conv,XQ);
set_diag_labels("A.8 - XQ (multiplied by carrier)","t(s)","");
grid on;
%PXI
sub_fig = sub_fig + 1;
set_diag_layout(fig, r, c, sub_fig);
c_semilogy(F_ES,PXI);
set_diag_labels("A.8 - Periodgram of XI (Logarithmic)","F [Hz]","");
grid on;
%PXQ
sub_fig = sub_fig + 1;
set_diag_layout(fig, r, c, sub_fig);
c_semilogy(F_ES,PXQ);
set_diag_labels("A.8 - Periodgram of XQ (Logarithmic)","F [Hz]","");
grid on;
```

## Διαγράμματα



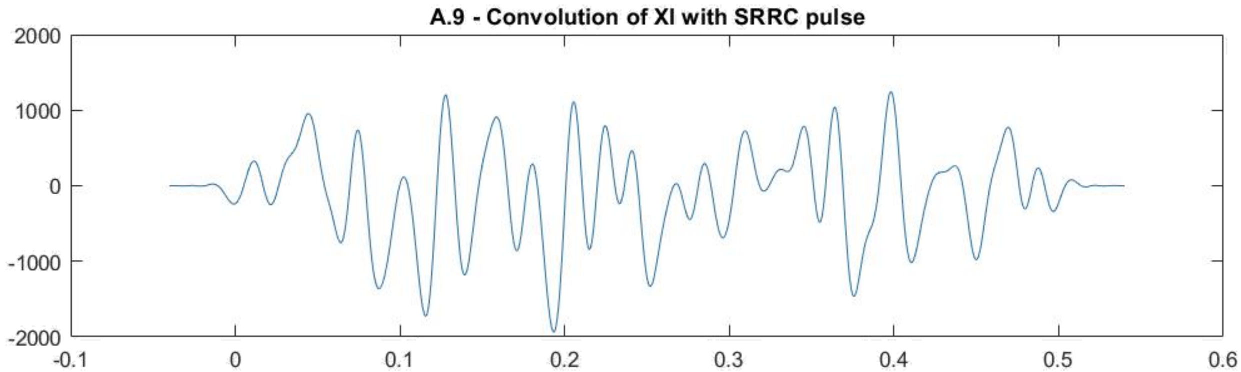
## A.9

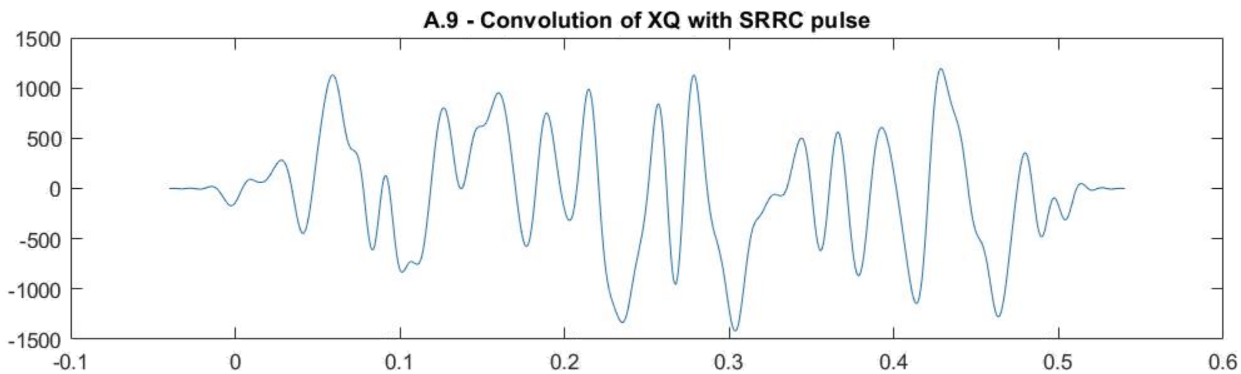
Φιλτράρουμε και πάλι τα σήματα με παλμούς SRRC και οι κυματομορφές και τα περιοδογράμματα φαίνονται στα επόμενα σχήματα:

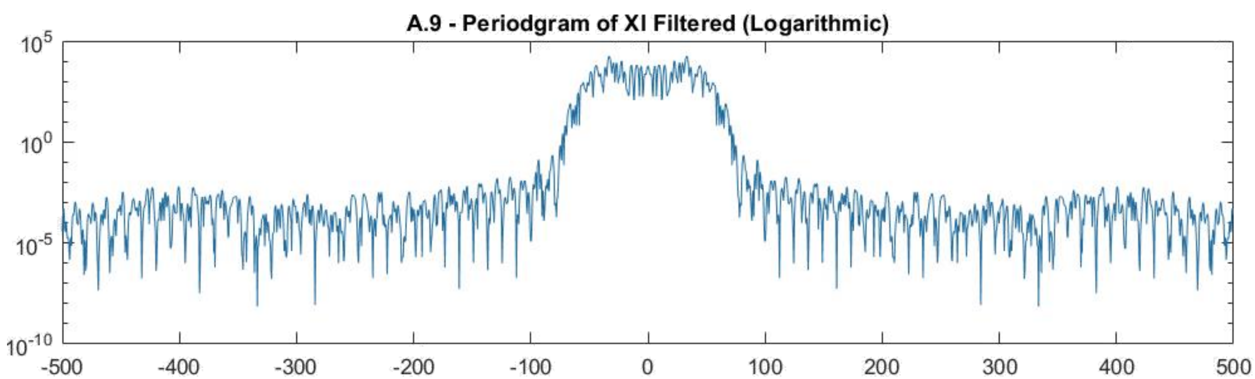
### Τμήμα Κώδικα

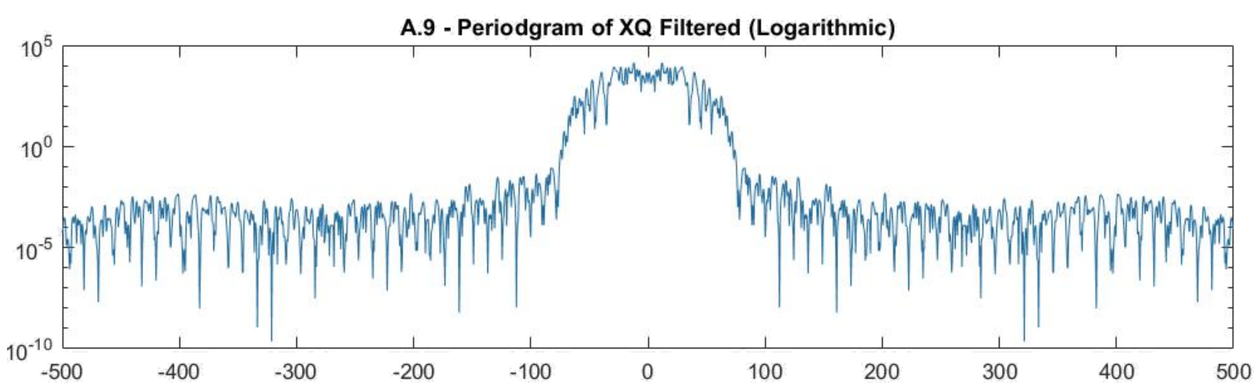
```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A9 - Filtering XI & XQ with SRRC Pulses                  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fig = fig + 1; sub_fig = 1; r = 2; c = 2; %New Figure

XI_conv = conv(XI, phi)*Ts;
XQ_conv = conv(XQ, phi)*Ts;

t_XI_conv = t_conv(t_phi, t_Xi, Ts);
t_XQ_conv = t_conv(t_phi, t_Xq, Ts);

%Calculating Periodgrams
F_ES = (-Fs/2) :(Fs/Nf): (Fs/2 - Fs/Nf);
PXI_2 = P_X(t_XI_conv, XI_conv,Nf,Ts);
PXQ_2 = P_X(t_XQ_conv, XQ_conv,Nf,Ts);

%Plotting output diagrams
%XI_conv
set_diag_layout(fig, r, c, sub_fig);
c_plot(t_XI_conv,XI_conv);
set_diag_labels("A.9 - Convolution of XI with SRRC pulse","t(s)","");
grid on;
%XQ_conv
sub_fig = sub_fig + 1;
set_diag_layout(fig, r, c, sub_fig);
c_plot(t_XQ_conv,XQ_conv);
set_diag_labels("A.9 - Convolution of XQ with SRRC pulse","t(s)","");
grid on;
%PXI_2
sub_fig = sub_fig + 1;
set_diag_layout(fig, r, c, sub_fig);
c_semilogy(F_ES,PXI_2);
set_diag_labels("A.9 - Periodgram of XI Filtered (Logarithmic)","F [Hz]","");
grid on;
%PXQ_2
sub_fig = sub_fig + 1;
set_diag_layout(fig, r, c, sub_fig);
c_semilogy(F_ES,PXQ_2);
set_diag_labels("A.9 - Periodgram of XQ Filtered (Logarithmic)","F [Hz]","");
grid on;
```

## Διαγράμματα



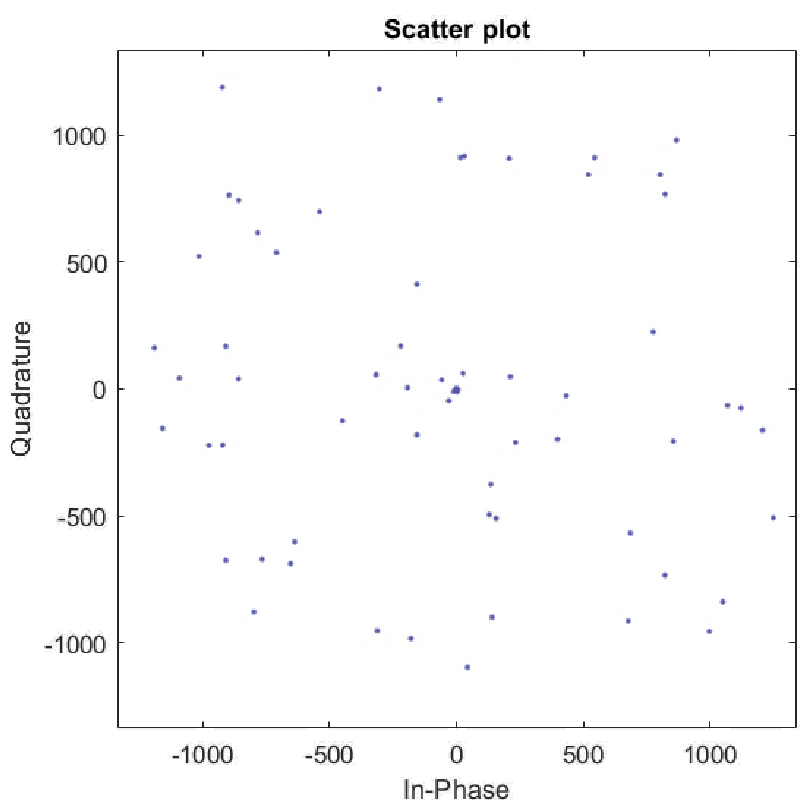
# A.10

Χρησιμοποιώντας τη συνάρτηση **`scatterplot`** του matlab, απεικονίζουμε την ακολοθία εξόδου Y. Παρατηρούμε οτι έχουμε σχετικά μεγάλες αποκλίσεις αλλα αυτό κυρίως οφείλεται στη μικρή σχετικά τιμή του SNR και στις λίγα δείγματα των σημάτων (Μικρό N).

## Τμήμα Κώδικα

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A10 - Using Scatterplot to sample plotted signals        %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
cout = 1;

for i = 1:over:length(t_XI_conv)
    X_wn(count,1) = XI_conv;
    X_wn(count,2) = XQ_conv;
    count = count + 1;
end
Y = [XI_conv XQ_conv];
scatterplot(Y');
```

## Διαγράμματα



# A.11

Κατασκευάστηκε τη συνάρτηση **`[est_X,est_bit_seq] = detect_PSK_8(Y,A)`** η οποία κάνει χρήση του nearest neighbor algorithm έτσι ώστε να παράξει μια εκτιμώμενη ακολουθία PSK.

## Τμήμα Κώδικα (Κύριο Μέρος)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A11 - Detecting PSK Sequence                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[est_X, est_bit_seq] = detect_PSK_8(Y,A);
```

## Τμήμα Κώδικα (Συνάρτηση)

```
function [est_X,est_bit_seq] = detect_PSK_8(Y,A)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% [est_X,est_bit_seq] = detect_PSK_8(Y,A)                                      %
% OUTPUT                                                                       %
%      est_X: estimated symbols                                                %
%      est_bit_seq: estimated bit sequence                                     %
% INPUT                                                                        %
%      Y: PSK Sequence                                                         %
%      A: Amplitude                                                            %
%                                                                              %
%    M. Galanis, Dec. 2018                                                     %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
est_X = zeros(1, length(Y));
m_length = 2^3;

possible_values_i = zeros(1,m_length);
for m = 0: m_length - 1
   possible_values_i(m + 1) = A*cos((m*pi)/4);
end

possible_values_q = zeros(1,m_length);
for m = 0 : m_length - 1
   possible_values_q(m + 1) = A*sin((m*pi)/4);
end

for i = 1 : length(Y)
distances = zeros(1,m_length);
    for m = 1 : m_length
        if (i < lenth(Y) /2)
            distances(m) = sqrt((Y(i,1) - possible_values_i(m))^2 + (Y(i,1) - possible_values_i(m))^2);
        else
            distances(m) = sqrt((Y(i,1) - possible_values_q(m))^2 + (Y(i,1) - possible_values_q(m))^2);
        end
    end

    index = 1;
    for m = 1 : m_length
        if (distances(m) == min(distances))
            est_X(i) = distances(m);
            index = m;
        end
    end
```

```
    b = de2bi(est_X(index));
    est_bit_seq(3 * i) = b(1);
    est_bit_seq(3 * i + 1) = b(2);
    est_bit_seq(3 * i + 2) = b(3);
end
return
```