# ΤΗΛ 301

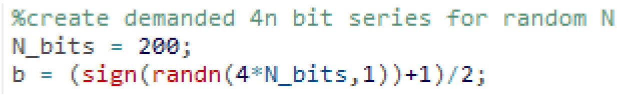
ΧΕΙΜΕΡΙΝΟ ΕΞΑΜΗΝΟ 2019-2020

# Εργαστηριακή Άσκηση 3

ΜΑΝΕΣΗΣ ΑΘΑΝΑΣΙΟΣ
ΑΜ:2014030061

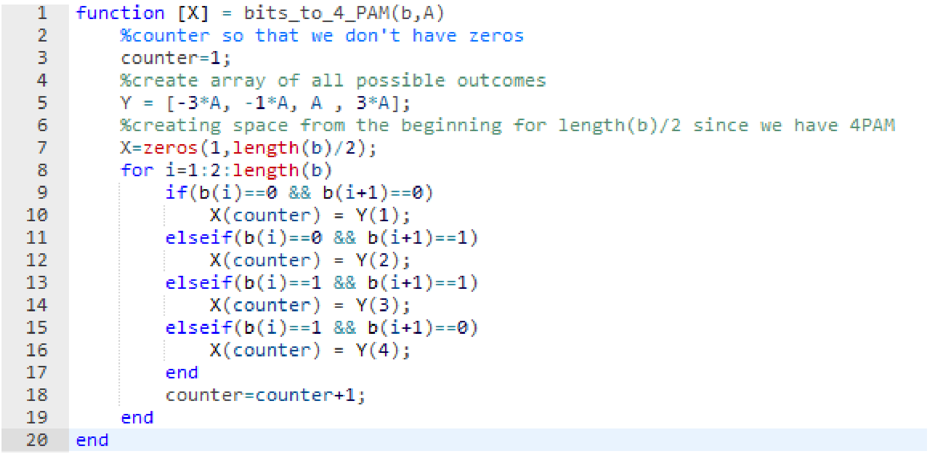
Σχολή
Ηλεκτρολόγων
Μηχανικών &
Μηχανικών
Υπολογιστών

## A1. Δημιουργία δυαδικής ακοκολουθίας N-bit (N=200 bits)



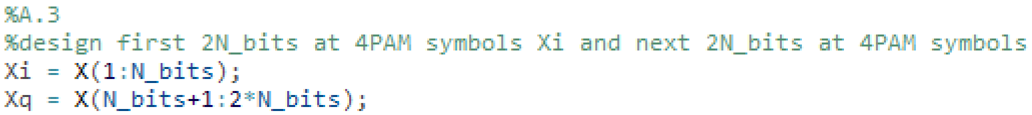
```
%create demanded 4n bit series for random N
N_bits = 200;
b = (sign(randn(4*N_bits,1))+1)/2;
```

## A2. Δημιουργία συνάρτησης bits to 4-PAM



```
function [X] = bits_to_4_PAM(b,A)
    %counter so that we don't have zeros
    counter=1;
    %create array of all possible outcomes
    Y = [-3*A, -1*A, A , 3*A];
    %creating space from the beginning for length(b)/2 since we have 4PAM
    X=zeros(1,length(b)/2);
    for i=1:2:length(b)
        if(b(i)==0 && b(i+1)==0)
            X(counter) = Y(1);
        elseif(b(i)==0 && b(i+1)==1)
            X(counter) = Y(2);
        elseif(b(i)==1 && b(i+1)==1)
            X(counter) = Y(3);
        elseif(b(i)==1 && b(i+1)==0)
            X(counter) = Y(4);
        end
        counter=counter+1;
    end
end
```

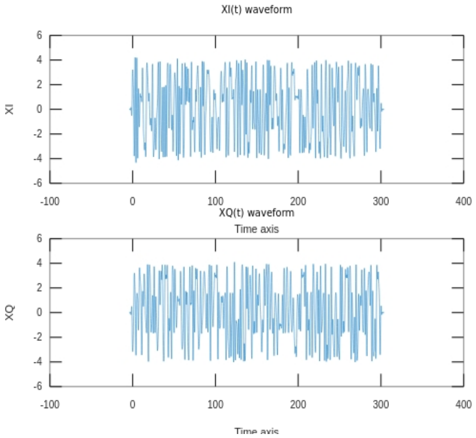
## A3. Δημιουργία $X_i$ , $X_q$

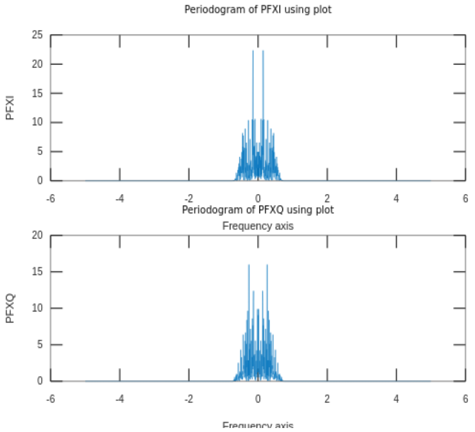


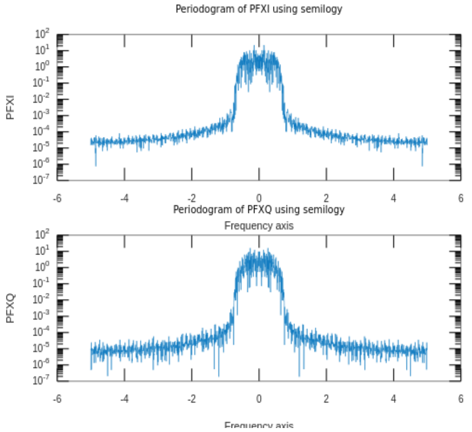
```
%A.3
%design first 2N_bits at 4PAM symbols Xi and next 2N_bits at 4PAM symbols
Xi = X(1:N_bits);
Xq = X(N_bits+1:2*N_bits);
```

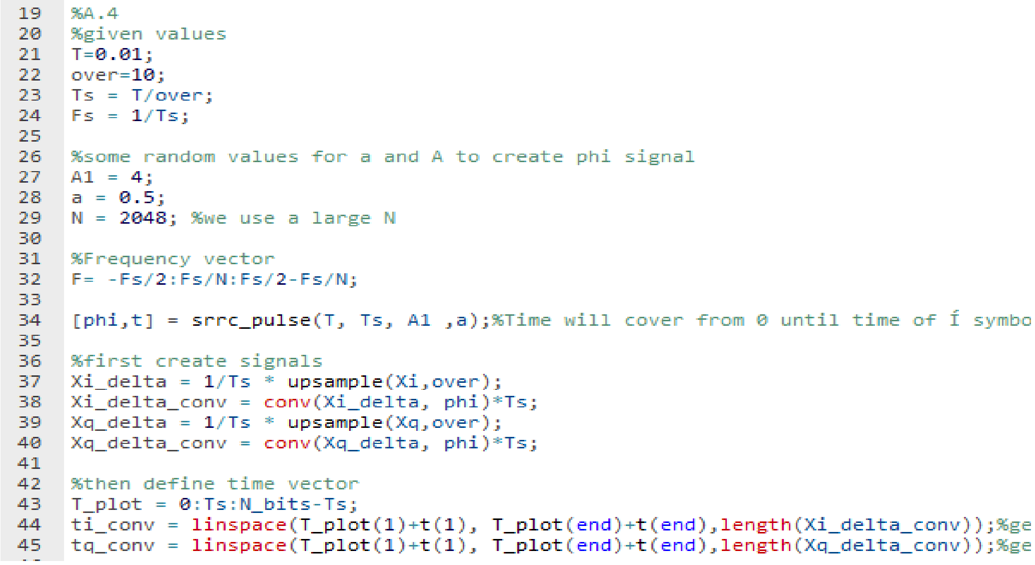
## A4. Κυματομορφές εξόδου $X_i$ , $X_q$

T=0.01sec, over=10





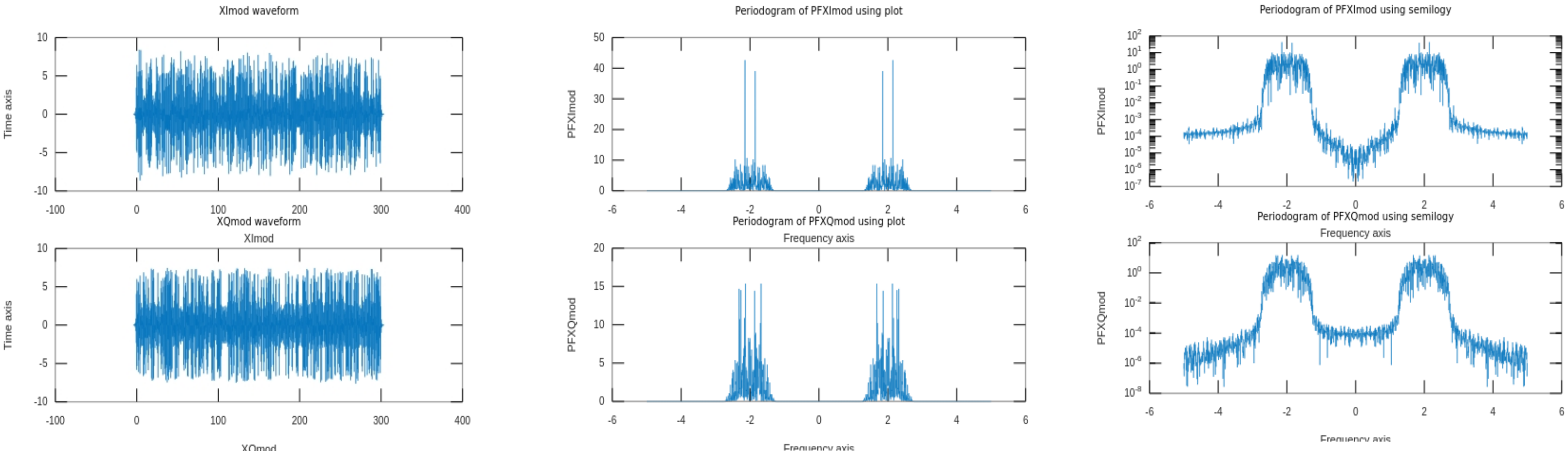


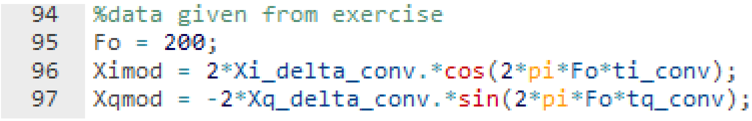


```
%A.4
%given values
T=0.01;
over=10;
Ts = T/over;
Fs = 1/Ts;

%some random values for a and A to create phi signal
A1 = 4;
a = 0.5;
N = 2048; %we use a large N

%Frequency vector
F= -Fs/2:Fs/N:Fs/2-Fs/N;

[phi,t] = srrc_pulse(T, Ts, A1 ,a);%Time will cover from 0 until time of Í symbo

%first create signals
Xi_delta = 1/Ts * upsample(Xi,over);
Xi_delta_conv = conv(Xi_delta, phi)*Ts;
Xq_delta = 1/Ts * upsample(Xq,over);
Xq_delta_conv = conv(Xq_delta, phi)*Ts;

%then define time vector
T_plot = 0:Ts:N_bits-Ts;
ti_conv = linspace(T_plot(1)+t(1), T_plot(end)+t(end),length(Xi_delta_conv));%ge
tq_conv = linspace(T_plot(1)+t(1), T_plot(end)+t(end),length(Xq_delta_conv));%ge
```

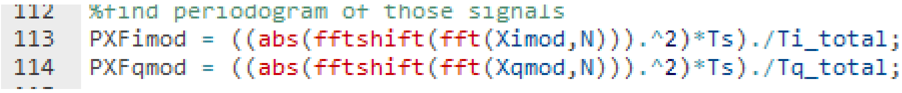
## A5. Κυματομορφές $X_i$mod , $X_q$mod

F0=200Hz



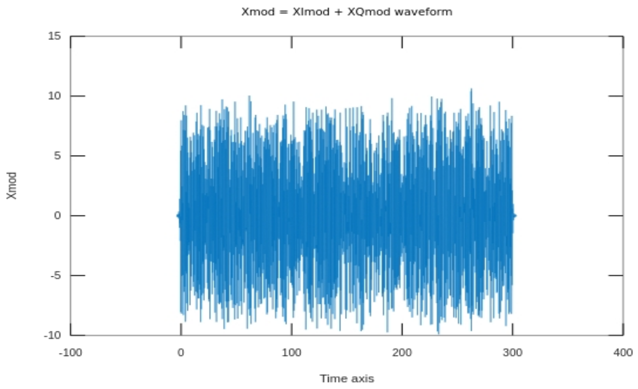


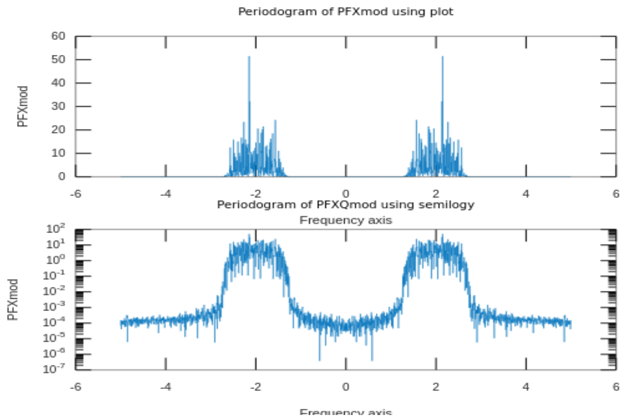
```
%data given from exercise
Fo = 200;
Ximod = 2*Xi_delta_conv.*cos(2*pi*Fo*ti_conv);
Xqmod = -2*Xq_delta_conv.*sin(2*pi*Fo*tq_conv);
```

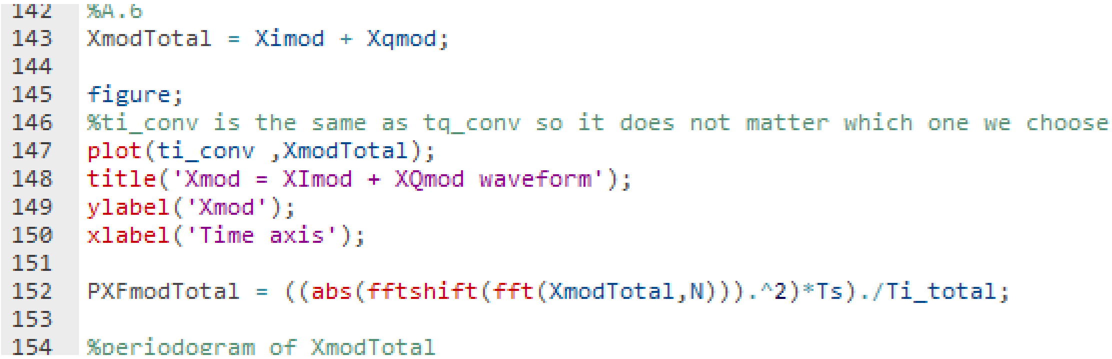


```
%find periodogram of those signals
PXFimod = ((abs(fftshift(fft(Ximod,N))).^2)*Ts)./Ti_total;
PXFqmod = ((abs(fftshift(fft(Xqmod,N))).^2)*Ts)./Tq_total;
```

## A6. Άθροιση κυματομορφών $X_i$mod , $X_q$mod

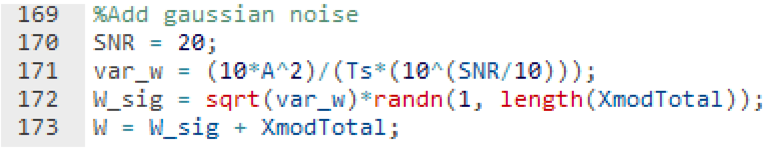






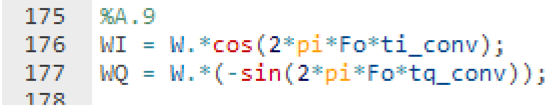
```
%A.6
XmodTotal = Ximod + Xqmod;

figure;
%ti_conv is the same as tq_conv so it does not matter which one we choose
plot(ti_conv ,XmodTotal);
title('Xmod = XImod + XQmod waveform');
ylabel('Xmod');
xlabel('Time axis');

PXFmodTotal = ((abs(fftshift(fft(XmodTotal,N))).^2)*Ts)./Ti_total;

%periodogram of XmodTotal
```

## A7. Προσθήκη Gaussian θορύβου

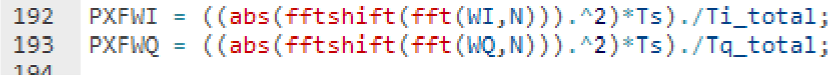


```
%Add gaussian noise
SNR = 20;
var_w = (10*A^2)/(Ts*(10^(SNR/10)));
W_sig = sqrt(var_w)*randn(1, length(XmodTotal));
W = W_sig + XmodTotal;
```

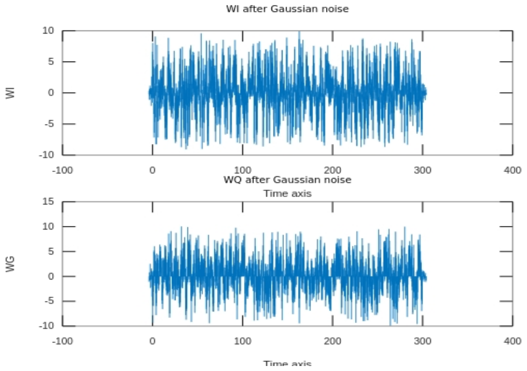
## A8. Διακλάδωση ενθόρυβης κυματομορφής

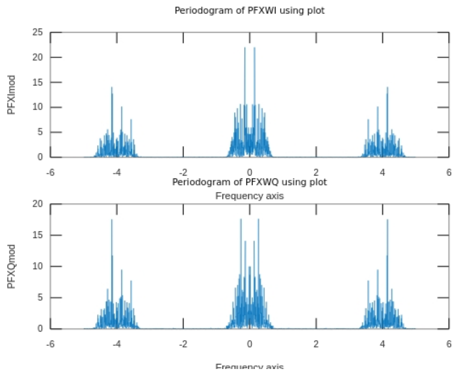


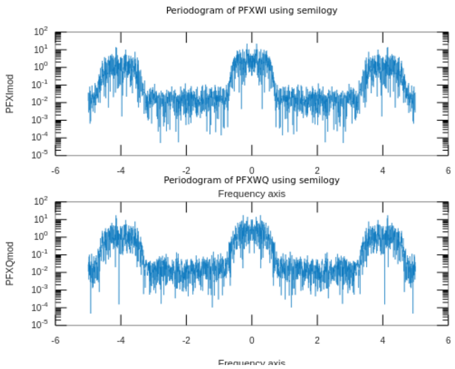
```
%A.9
WI = W.*cos(2*pi*Fo*ti_conv);
WQ = W.*(-sin(2*pi*Fo*tq_conv));
```



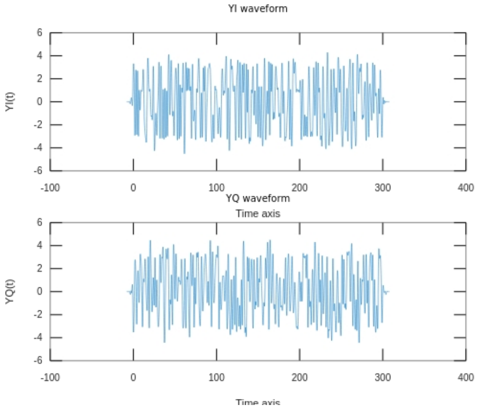
```
PXFWI = ((abs(fftshift(fft(WI,N))).^2)*Ts)./Ti_total;
PXFWQ = ((abs(fftshift(fft(WQ,N))).^2)*Ts)./Tq_total;
```

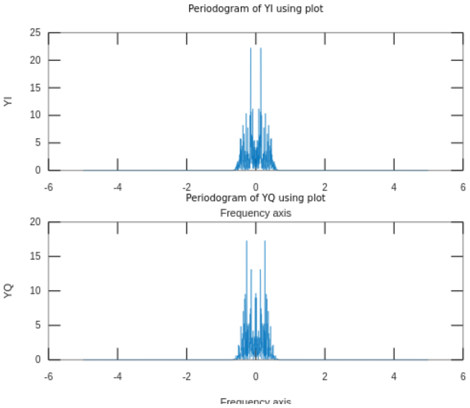


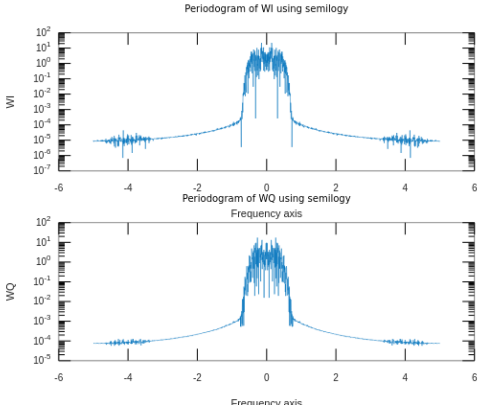


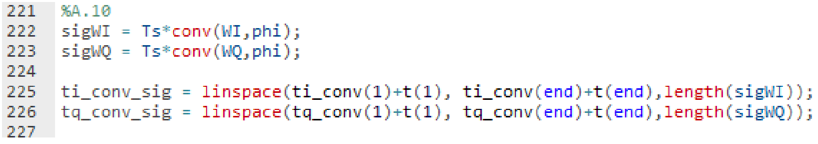


## A9. Φιλτράρισμα ενθόρυβων κυματομορφών με SRRC

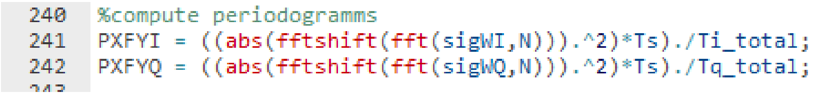






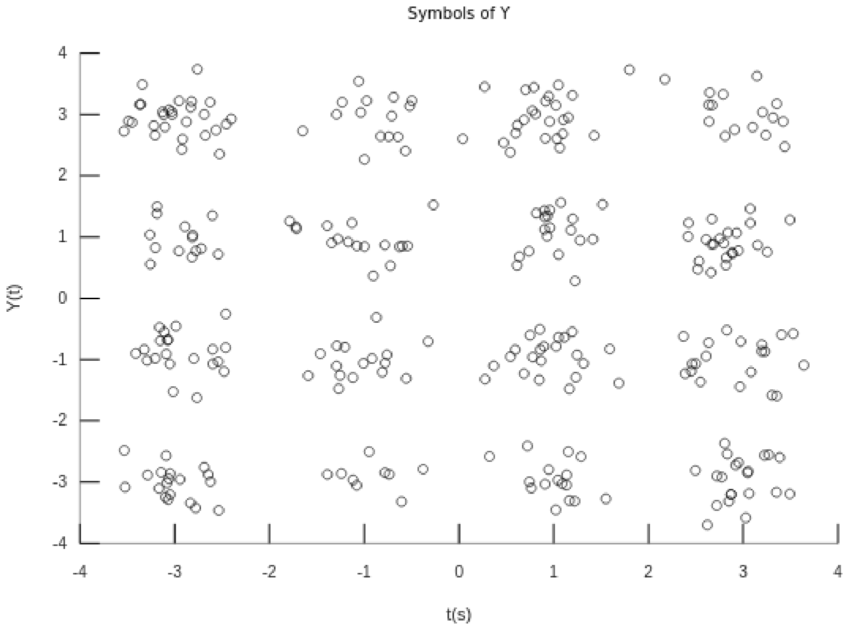


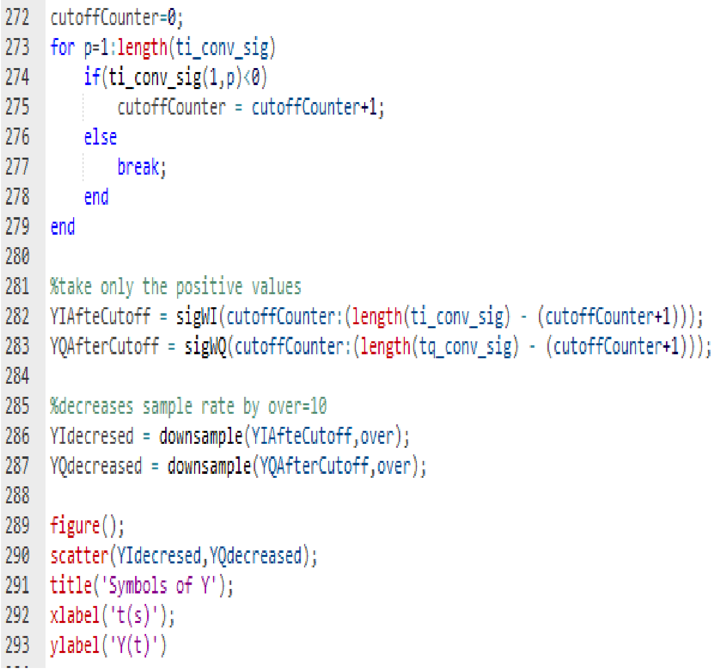
```
%A.10
sigWI = Ts*conv(WI,phi);
sigWQ = Ts*conv(WQ,phi);

ti_conv_sig = linspace(ti_conv(1)+t(1), ti_conv(end)+t(end),length(sigWI));
tq_conv_sig = linspace(tq_conv(1)+t(1), tq_conv(end)+t(end),length(sigWQ));

```



```
%compute periodogramms
PXFYI = ((abs(fftshift(fft(sigWI,N))).^2)*Ts)./Ti_total;
PXFYQ = ((abs(fftshift(fft(sigWQ,N))).^2)*Ts)./Tq_total;
```

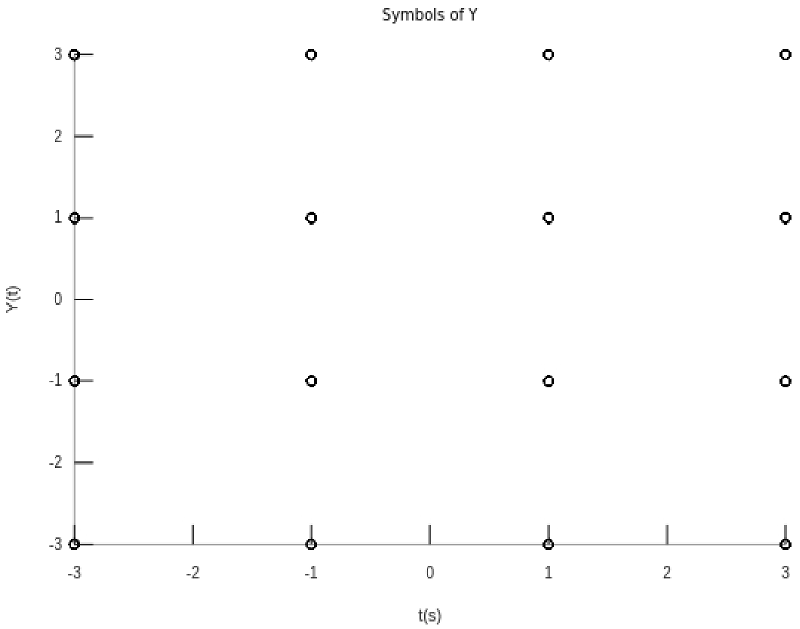
## A10. Δειγματοληψία εξόδου προσαρμοσμένων SRRC φίλτρων

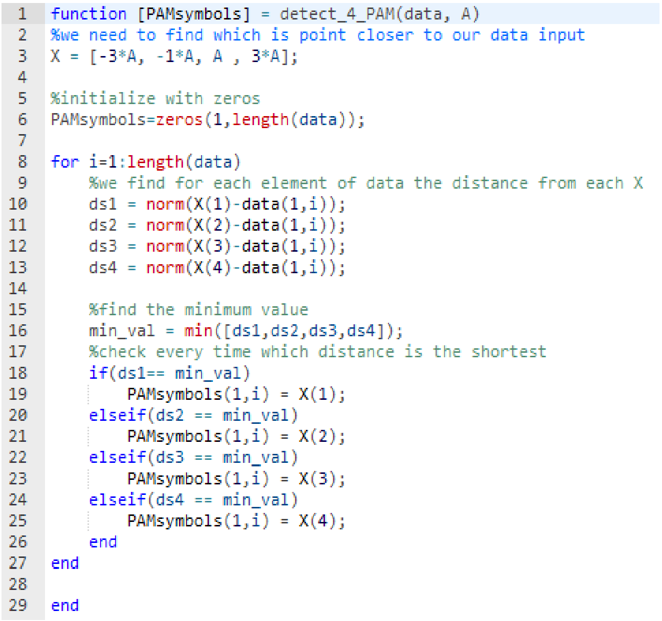




```
cutoffCounter=0;
for p=1:length(ti_conv_sig)
    if(ti_conv_sig(1,p)<0)
        cutoffCounter = cutoffCounter+1;
    else
        break;
    end
end

%take only the positive values
YIAfteCutoff = sigWI(cutoffCounter:(length(ti_conv_sig) - (cutoffCounter+1)));
YQAfterCutoff = sigWQ(cutoffCounter:(length(tq_conv_sig) - (cutoffCounter+1)));

%decreases sample rate by over=10
YIdecresed = downsample(YIAfteCutoff,over);
YQdecreased = downsample(YQAfterCutoff,over);

figure();
scatter(YIdecresed,YQdecreased);
title('Symbols of Y');
xlabel('t(s)');
ylabel('Y(t)')
```

## A11. Δημιουργία detect 4-PAM





```
function [PAMsymbols] = detect_4_PAM(data, A)
%we need to find which is point closer to our data input
X = [-3*A, -1*A, A , 3*A];

%initialize with zeros
PAMsymbols=zeros(1,length(data));

for i=1:length(data)
    %we find for each element of data the distance from each X
    ds1 = norm(X(1)-data(1,i));
    ds2 = norm(X(2)-data(1,i));
    ds3 = norm(X(3)-data(1,i));
    ds4 = norm(X(4)-data(1,i));

    %find the minimum value
    min_val = min([ds1,ds2,ds3,ds4]);
    %check every time which distance is the shortest
    if(ds1== min_val)
        PAMsymbols(1,i) = X(1);
    elseif(ds2 == min_val)
        PAMsymbols(1,i) = X(2);
    elseif(ds3 == min_val)
        PAMsymbols(1,i) = X(3);
    elseif(ds4 == min_val)
        PAMsymbols(1,i) = X(4);
    end
end

end
```