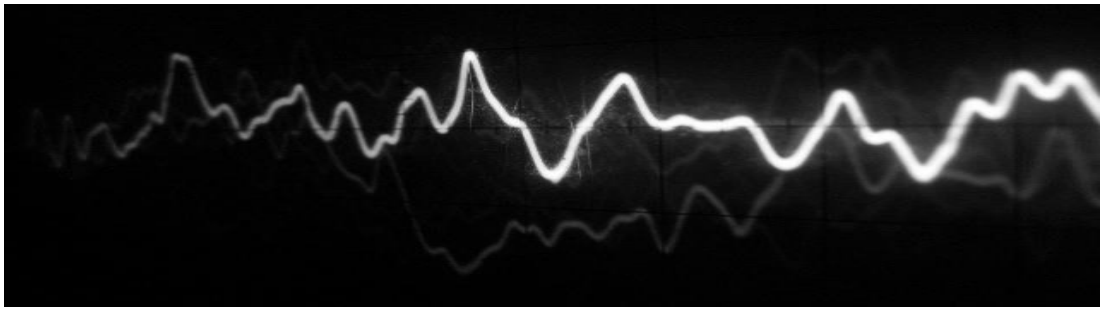
Ψηφιακές Τηλεπικοινωνίες

Ακ. Έτος 2017-18

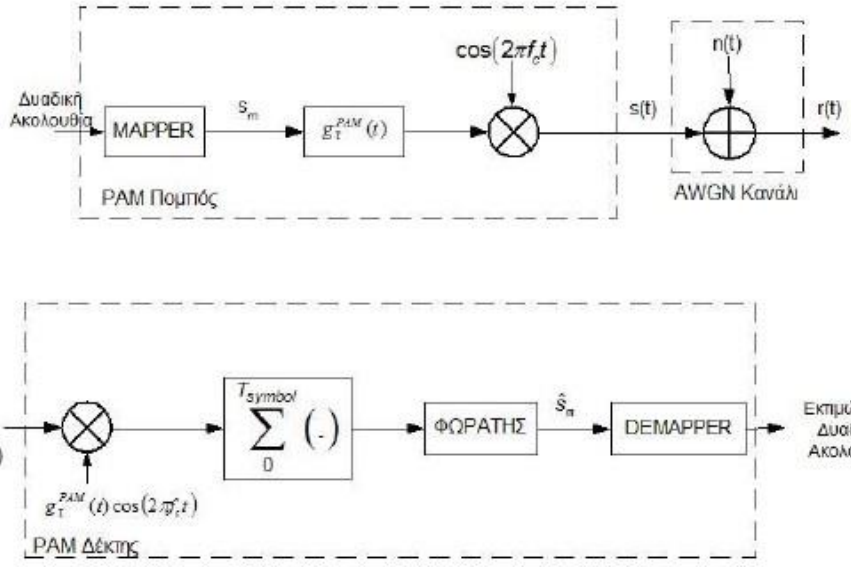
$1^{η}$ εργαστηριακή άσκηση



Σιαφλέκης Αναστάσιος
Α.Μ.4634
siaflekis@ceid.upatras.gr

Μέρος Α

Μελέτη Απόδοσης Ομόδυνου Ζωνοπερατού Συστήματος M-PAM



Ερώτημα 1

Η διαμόρφωση παλμών κατά πλάτος (Pulse Ampplitude Modulation-PAM) είναι μια μορφή διαμόρφωσης σήματος, όπου οι πληροφορίες που θέλουμε να στείλουμε μέσα από ένα ζωνοπερατό κανάλι, κωδικοποιούνται πολλαπλασιαζόμενες μ' ένα ημιτονοειδές φέρον της μορφής $\cos 2\pi f_c t$.

Το σύστημα M-PAM που υλοποιήθηκε περιλαμβάνει τα εξής τμήματα:

Τον **MAPPER**, ο οποίος αντιστοιχεί τα δυαδικά ψηφία της ακολουθίας εισόδου σε σύμβολα. Εφ' όσον ενεργοποιείται η κατάλληλη μεταβλητή, χρησιμοποιεί και κωδικοποίηση Gray.

Τον **MODULATOR** (διαμορφωτής), ο οποίος παράγει το ζωνοπερατό σήμα, που αντιστοιχεί στο σύμβολο που μεταδίδουμε.

Το **AWGN κανάλι**, το οποίο μοντελοποιεί ένα κανάλι επικοινωνίας, που προσθέτει λευκό και Gaussian κατανομής θόρυβο στο σήμα μας.

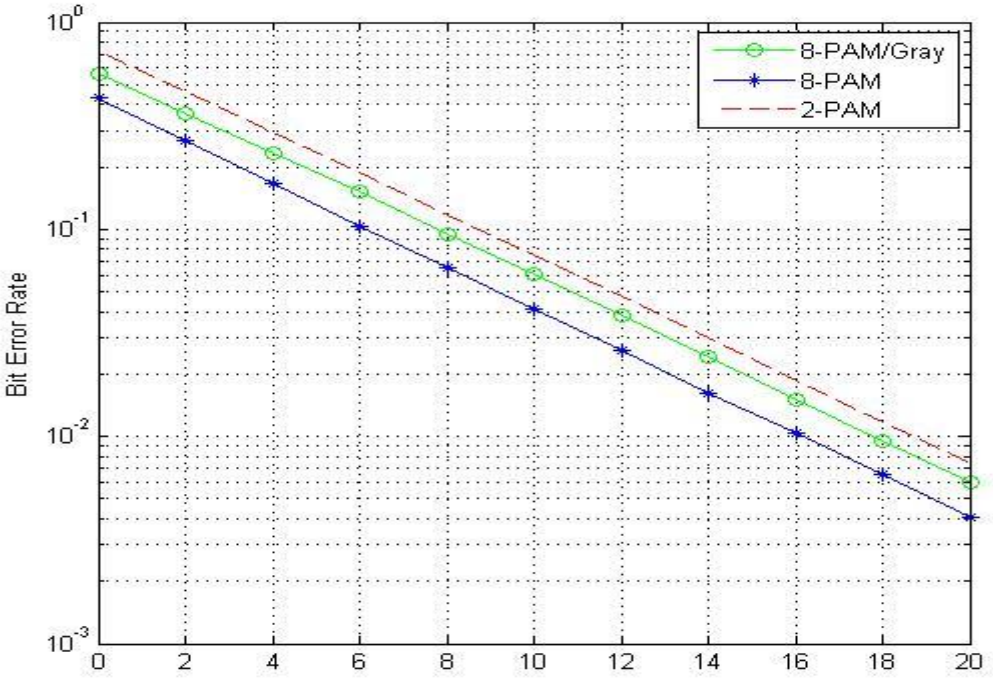
Τον **DEMODULATOR** (αποδιαμορφωτής), ο οποίος αποδιαμορφώνει το ζωνοπερατό σήμα και εξάγει σύμβολα.

Τους **DETECTOR** & **DEMAPPER**, οι οποίοι αντιστοιχούν τα σύμβολα που αποδιαμορφώθηκαν με τα σύμβολα που χρησιμοποιεί το συγκεκριμένο αλφάβητο.

Στο παράρτημα παρατίθενται αναλυτικά οι υλοποιήσεις των τμημάτων του συστήματος σε Matlab.

Ερώτημα 2

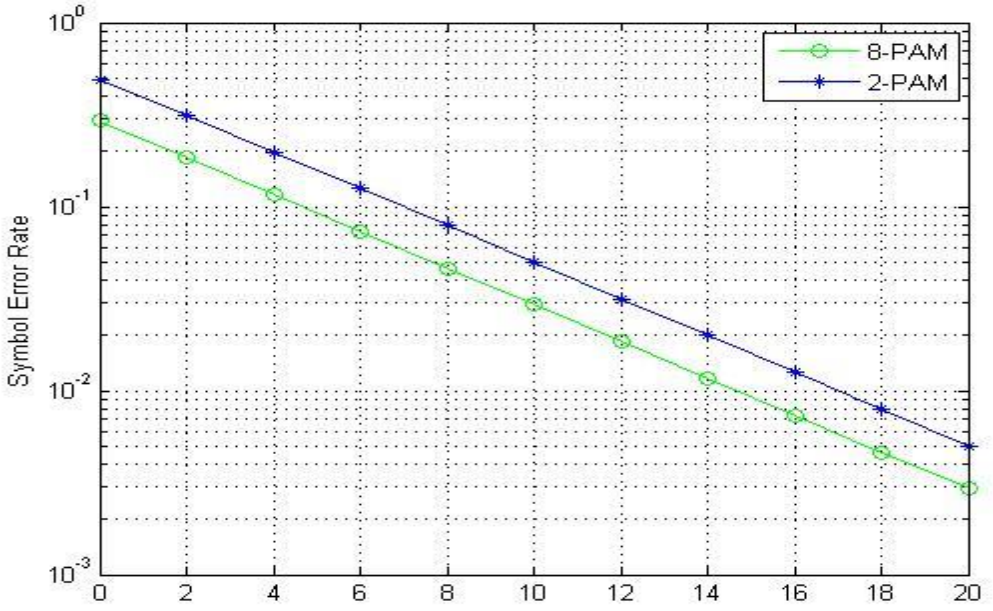
Η πειραματική απεικόνιση του BER για M=2 (απλή κωδικοποίηση) και M=8 (απλή και κατά Gray κωδικοποίηση) δίνεται στο παρακάτω σχήμα:



Παρατηρούμε ότι πως η πιθανότητα εμφάνισης σφάλματος bit (BER) μειώνεται όσο αυξάνεται το SNR. Αυτό συμβαίνει γιατί το SNR εκφράζει το λόγο της ισχύος εκπομπής του σήματος, προς τη μέση ισχύ του θορύβου. Άρα όσο μεγαλώνει αυτός ο λόγος, τόσο πιο ανθεκτικό θα γίνεται το σήμα προς τα σφάλματα.

Ερώτημα 3

Η πειραματική απεικόνιση του SER για M=2 & 8 (απλή κωδικοποίηση) δίνεται στο παρακάτω σχήμα:



Παρατηρούμε ότι κατ' αναλογία με πριν, η πιθανότητα εμφάνισης σφάλματος συμβόλου (SER) μειώνεται όσο αυξάνεται το SNR.

Μέρος Β

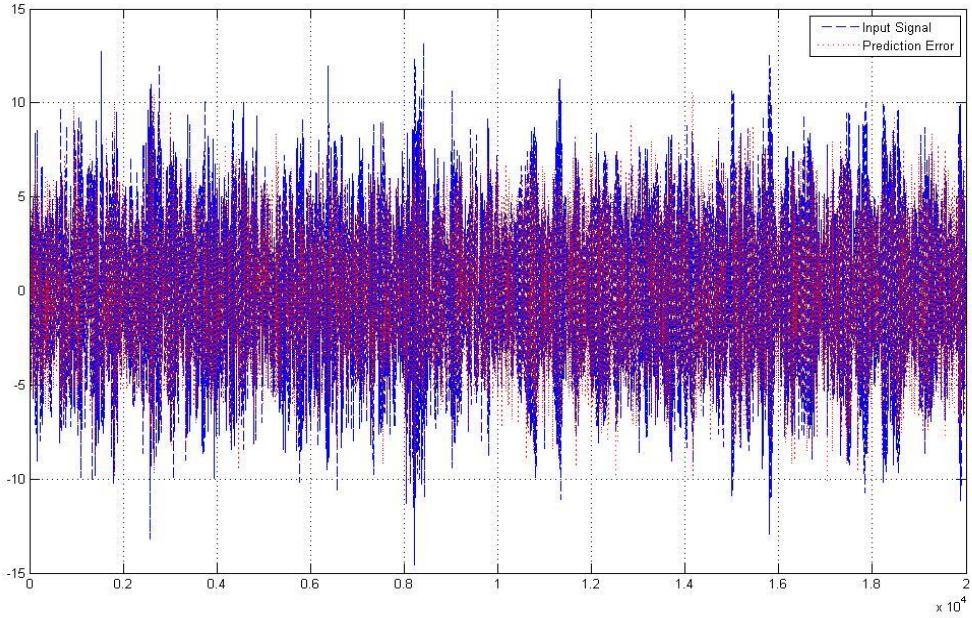
Κωδικοποίηση Διακριτής Πηγής με τη μέθοδο DPCM

Η υλοποίηση του συστήματος περιέχει έναν ομοιόμορφο κβαντιστή 2^N επιπέδων, καθώς και το φίλτρο πρόβλεψης με μια διάταξη μνήμης (κοινά για πομπό και δέκτη). **Αναλυτικά η υλοποίηση βρίσκεται στο τέλος της αναφοράς στους κώδικες των αρχείων my_quantizer.m, my_DPCM.m, erotima3.m.**

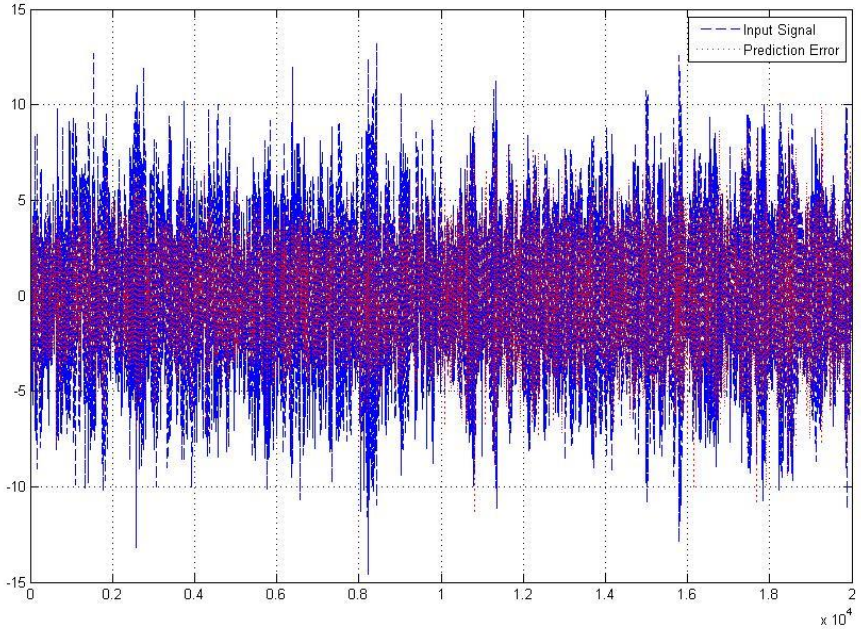
Ερώτημα 2

Παρουσιάζονται γραφήματα αρχικού σήματος και σφάλματος πρόβλεψης για p= 5,10 και για N=1,2,3. Με μπλε χρώμα αναπαρίσταται το αρχικό σήμα και με κόκκινο το σφάλμα πρόβλεψης.

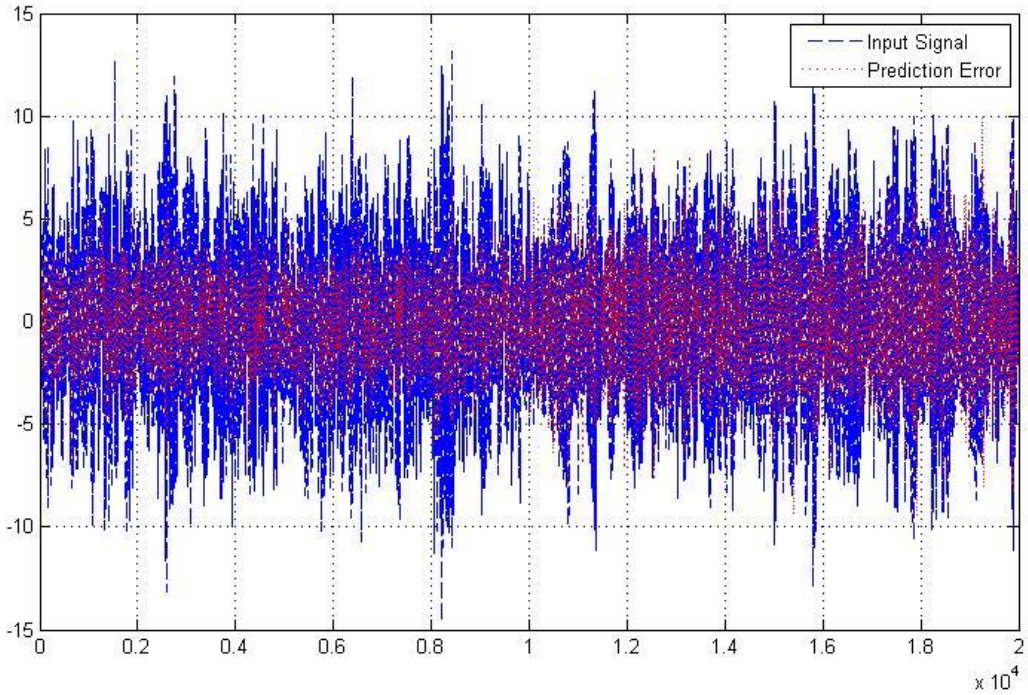
p= 5 & N=1



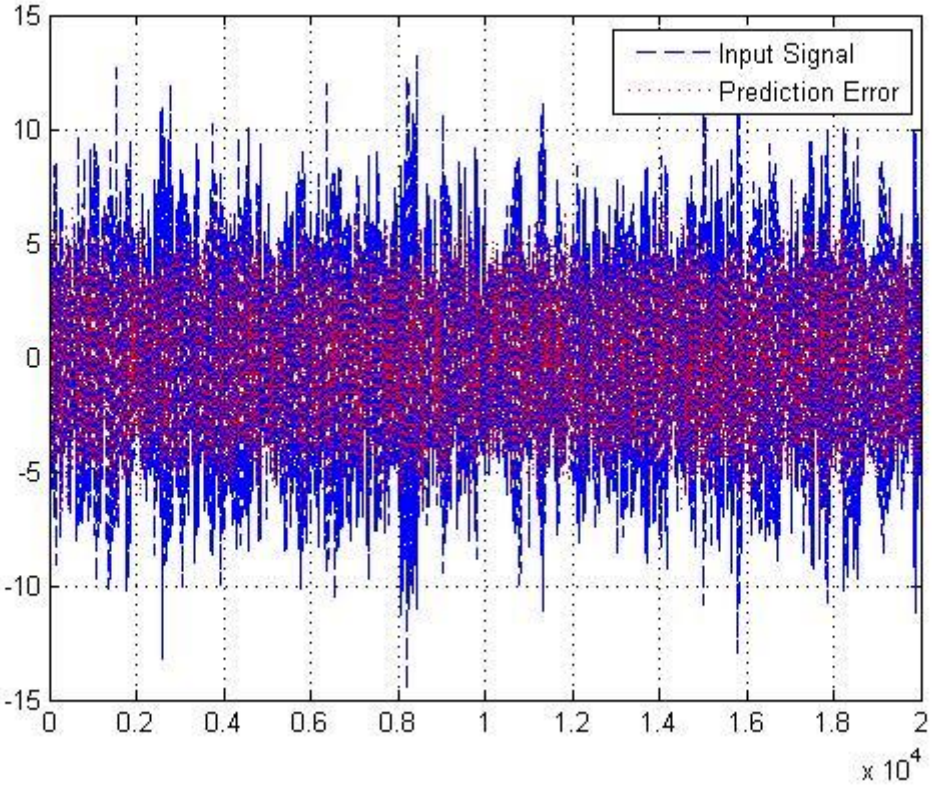
p= 5 & N=2



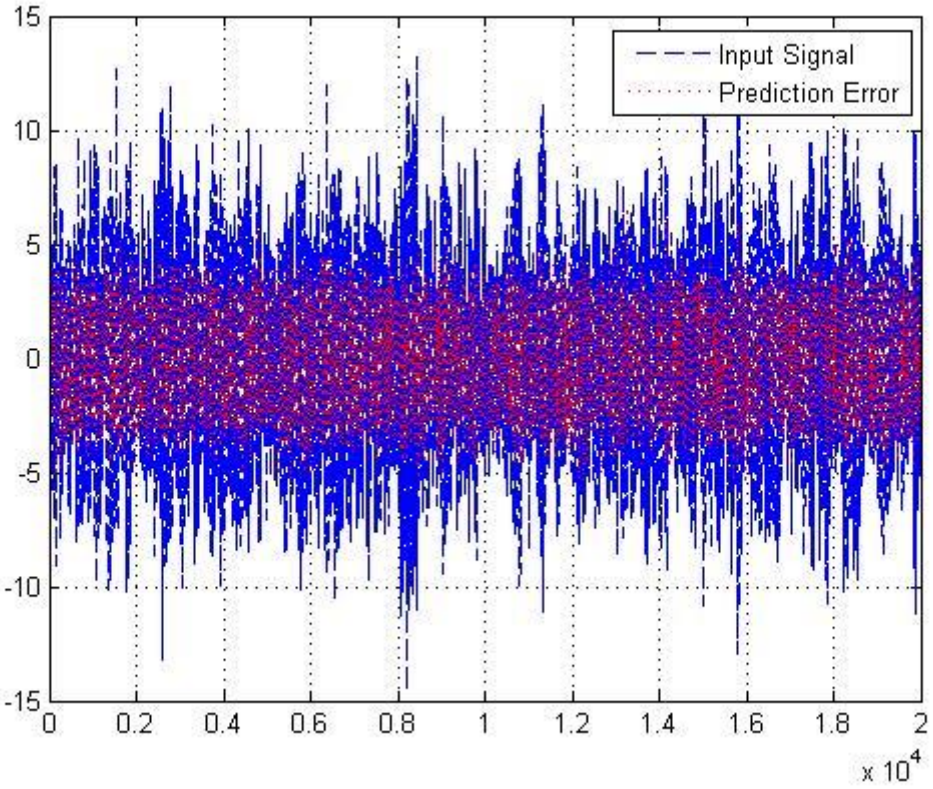
p= 5 & N=3


Input Signal
Prediction Error
15
10
5
0
-5
-10
-15
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
x 10^4

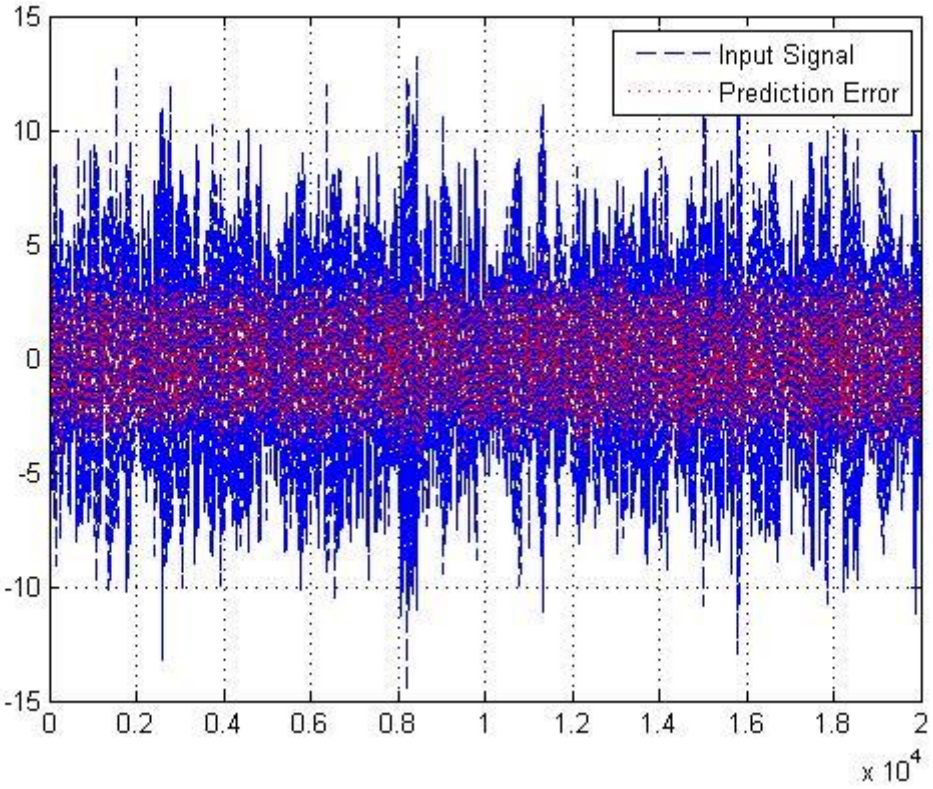
p= 10 & N=1


Input Signal
Prediction Error
15
10
5
0
-5
-10
-15
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
x 10^4

p= 10 & N=2



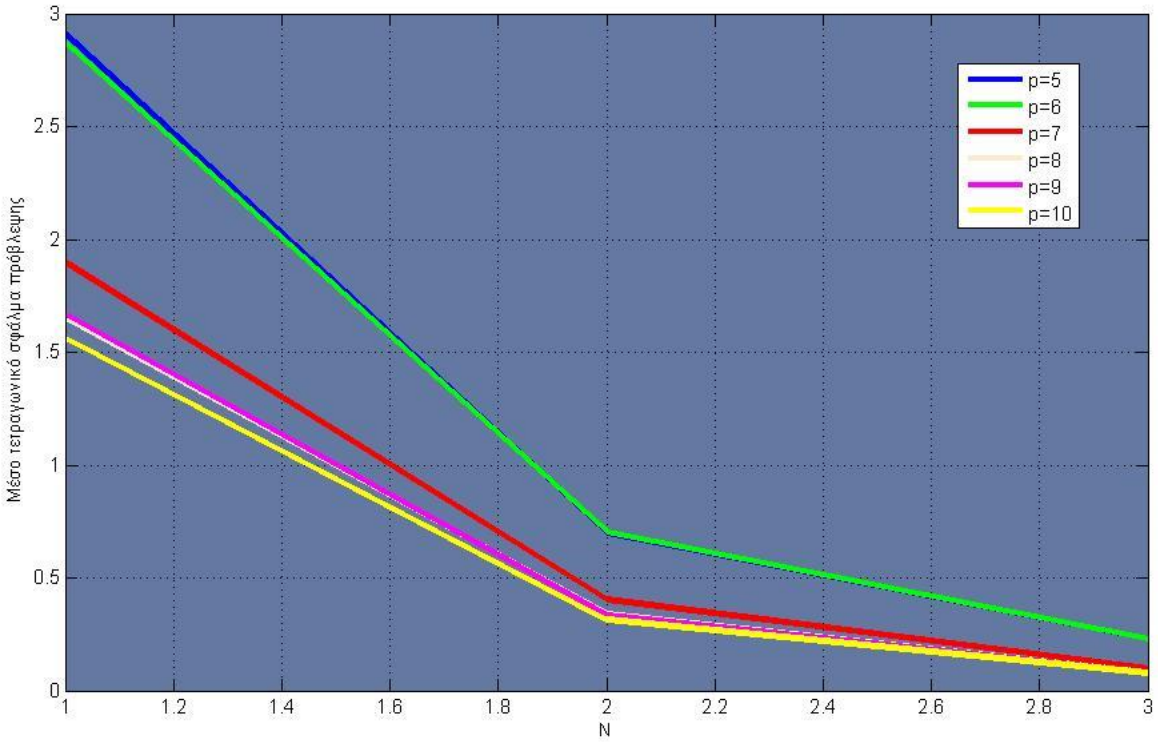
p= 10 & N=3



Από τα παραπάνω παρατηρούμε ότι αν και το p παίρνει μικρή τιμή ή μεγαλύτερη (δηλαδή είτε χρησιμοποιηθούν λιγότερα ή περισσότερα δείγματα από το φίλτρο πρόβλεψης), το N παίζει καθοριστικό ρόλο στον περιορισμό του σφάλματος πρόβλεψης. Όσο μεγαλύτερη η τιμή του N, τόσο λιγότερα σφάλματα έχουμε.

Ερώτημα 3

Για την αξιολόγηση της απόδοσης του συστήματος παρουσιάζεται γράφημα όπου φαίνεται το μέσο τετραγωνικό σφάλμα πρόβλεψης ως προς το N(=1,2,3 bits) για τις διάφορες τιμές του p(=5:10).



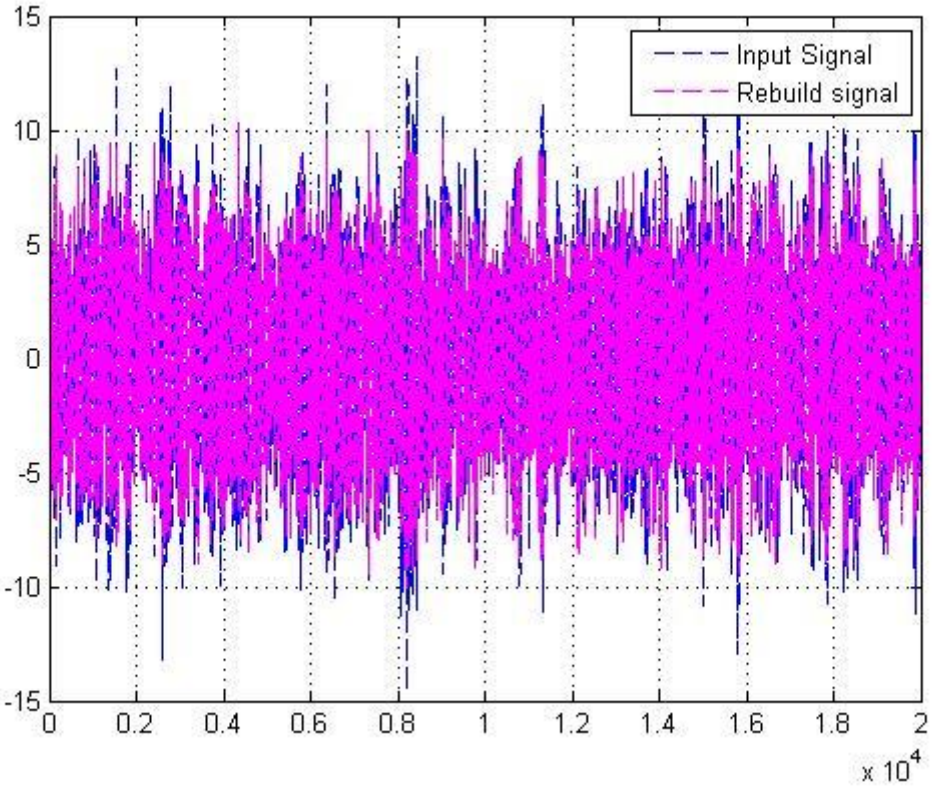
Γίνεται αντιληπτό ότι όσο μεγαλύτερο είναι το N, δηλαδή όσο περισσότερα bit κβάντισης χρησιμοποιούνται από το σύστημα, τόσο μειώνεται το σφάλμα πρόβλεψης. Αυτό έχει ως αποτέλεσμα να ανακατασκευάζεται με μεγαλύτερη ακρίβεια το αρχικό σήμα από το φίλτρο πρόβλεψης.

Ταυτόχρονα βλέπουμε ότι όσο το p αυξάνεται, δηλαδή όσο περισσότερα δείγματα χρησιμοποιούνται για την πρόβλεψη του επόμενου δείγματος, αυξάνεται και η απόδοση του συστήματος. Με άλλα λόγια για τον ίδιο αριθμό N, όσο μεγαλύτερο είναι το p, τόσο καλύτερα ανακατασκευάζεται το αρχικό σήμα.

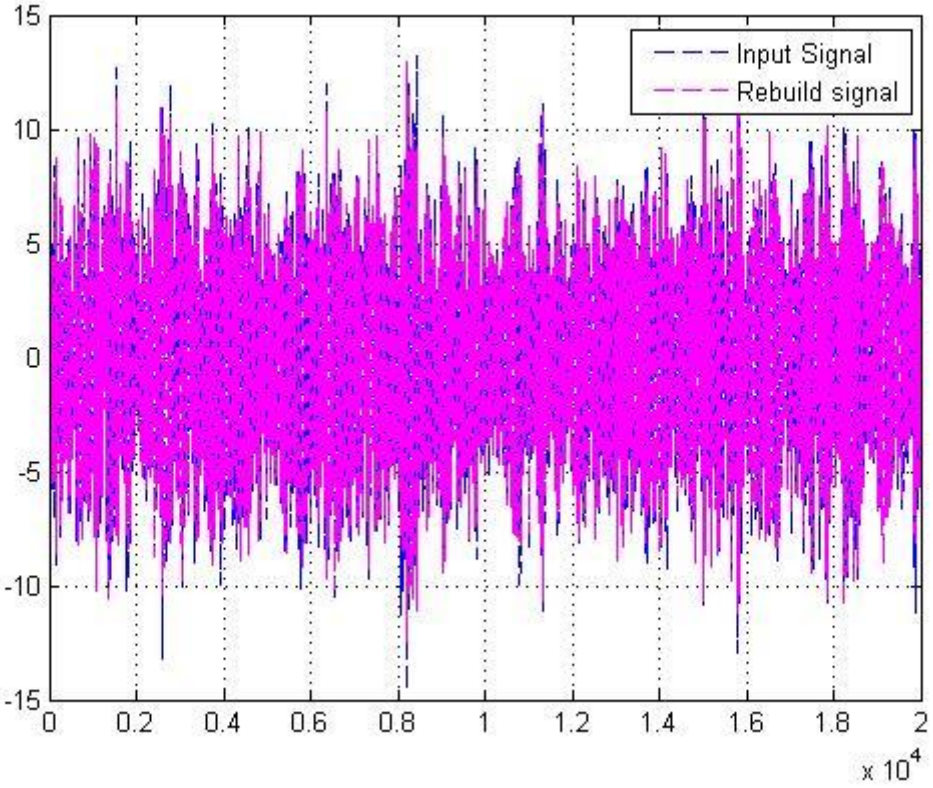
Ερώτημα 4

Παρουσιάζονται γραφήματα του αρχικού σήματος και του ανακατασκευασμένου σήματος στο δέκτη για p= 5,10 και για N=1,2,3.

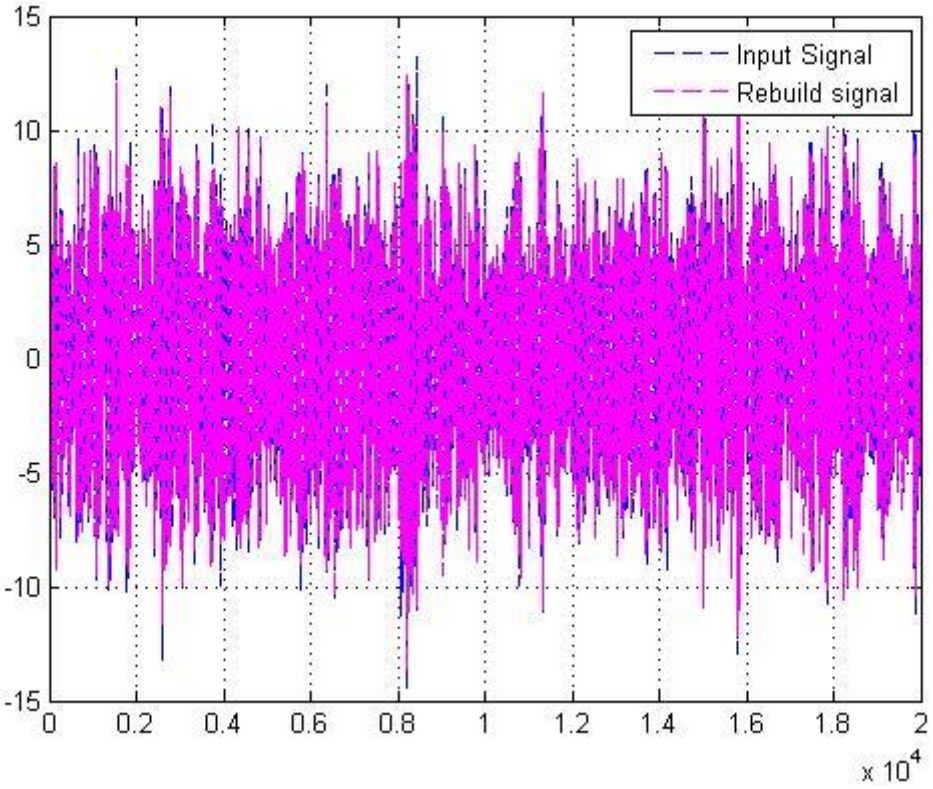
p=5 & N=1


Input Signal
Rebuild signal
15
10
5
0
-5
-10
-15
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
x 10^4

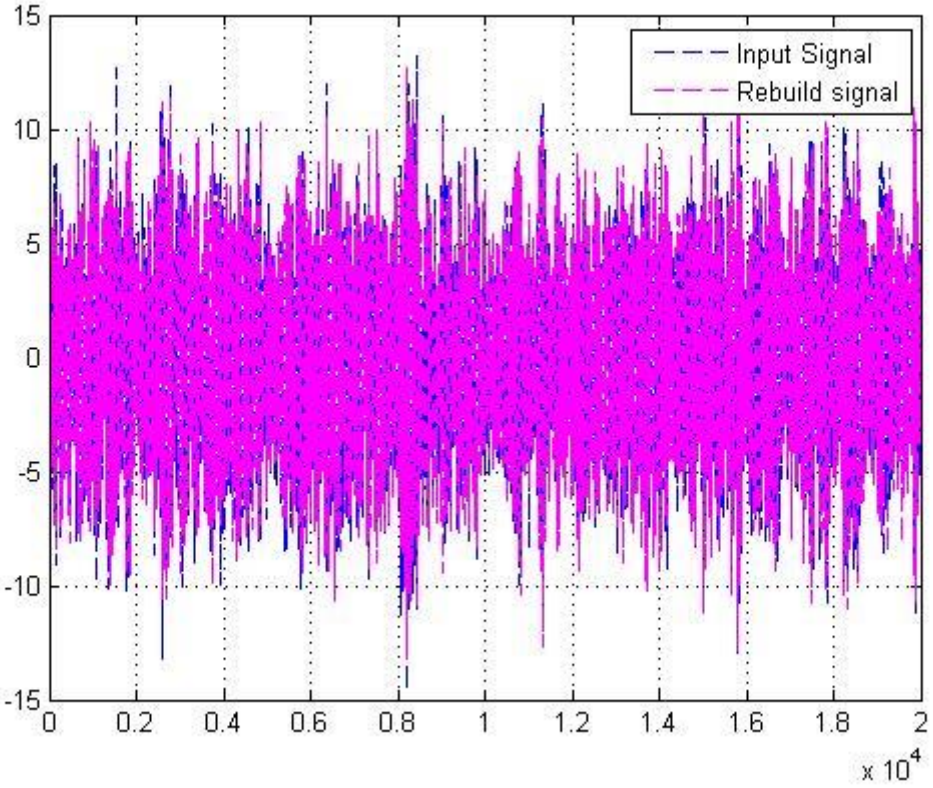
p=5 & N=2


Input Signal
Rebuild signal
15
10
5
0
-5
-10
-15
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
x 10^4

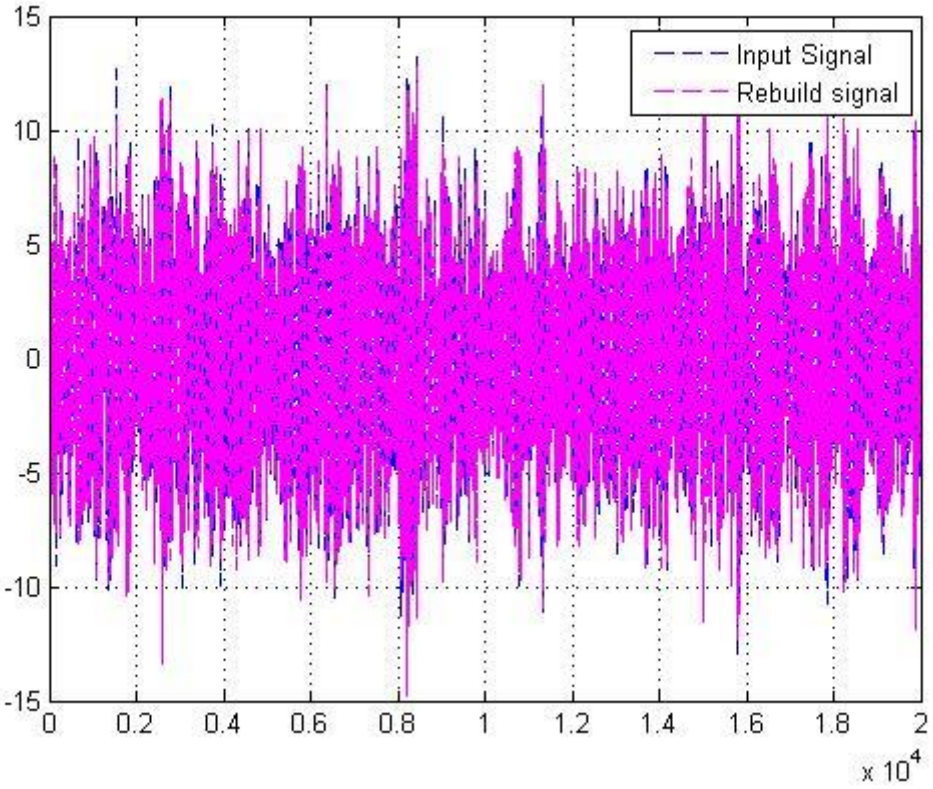
p=5 & N=3


Input Signal
Rebuild signal
15
10
5
0
-5
-10
-15
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
x 10^4

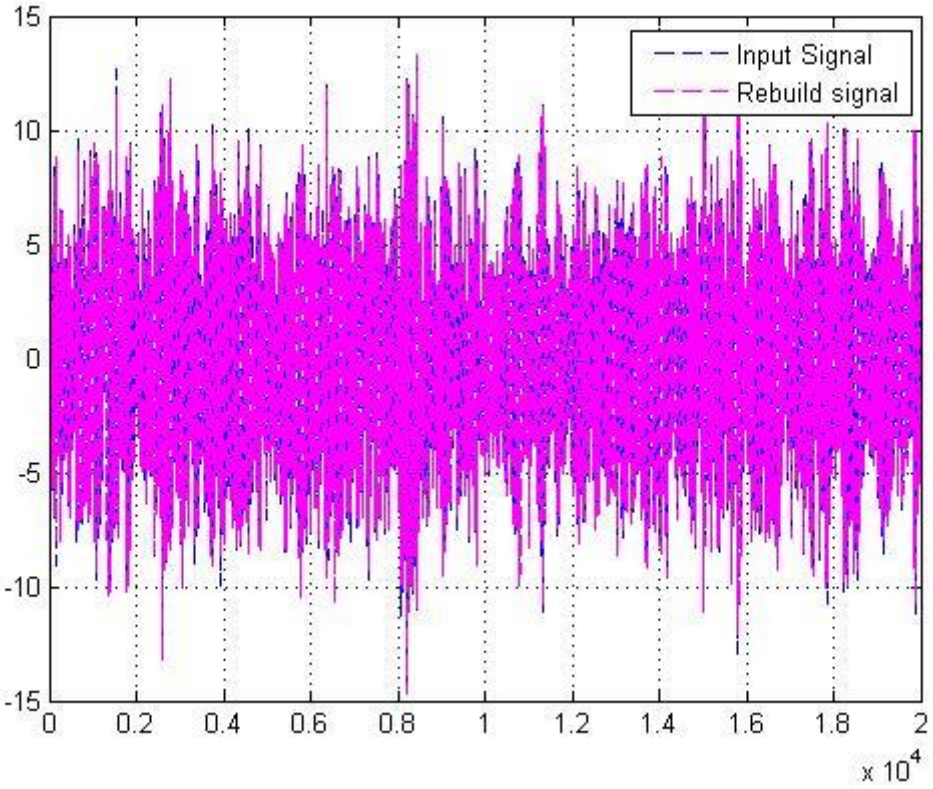
p=10 & N=1


Input Signal
Rebuild signal
15
10
5
0
-5
-10
-15
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
x 10^4

p=10 & N=2



p=10 & N=3



Παρατηρούμε ότι η τιμή του p παίζει σημαντικότερο ρόλο στην ακρίβεια του ανακατασκευασμένου σήματος από τον δέκτη σε σχέση με τα N bit κβάντισης. Βλέπουμε ότι για p=10, ακόμα και με N= 1 bit

κβάντισης το ανακατασκευασμένο σήμα είναι πολύ πιο κοντά στο αρχικό σε σχέση με p=5 και N= 1. Ταυτόχρονα ισχύει ότι όσο μεγαλύτερες τιμές έχουν το p και το N, το ανακατασκευασμένο σήμα θα αναπαριστά με μεγαλύτερη ακρίβεια το αρχικό σήμα.

ΠΑΡΑΡΤΗΜΑ-ΚΩΔΙΚΕΣ

ΜΕΡΟΣ Α

mapper.m

```
function symbols = mapper(bit_sequence, g, M)

if M==8
sym_seq_len = floor((length(bit_sequence)/3));
elseif M==2
  sym_seq_len = floor((length(bit_sequence)/2));
else
    msg='Not programmed to do anything else';
    error(msg)
end

symbol_sequence = zeros(sym_seq_len, 1);

j = 1;

if M==8
    for i=1:3:length(bit_sequence)
    symbol_sequence(j) = bit_sequence(i)+2*bit_sequence(i+1)+4*bit_sequence(i+2);

        j = j + 1;
    end
else

    for i=1:2:length(bit_sequence)
    symbol_sequence(j) = 2*bit_sequence(i) + bit_sequence(i + 1);
    j = j + 1;
    end
end

if g==1
symbols= bin2gray(symbol_sequence,'pam',M);
else symbols= symbol_sequence;
end




end

```

modulator.m

```
function s=modulator(symbols,M)

T_c=4;
f_c=1/T_c;
T_symbol= 40;

E_s=1;
g_t = sqrt((2 * E_s) / T_symbol);

A=sqrt(3/(M^2-1)); %Eaverage=Eg(M^2-1)/3=1 (proakis sel 392)
Am = (2*symbols-(M+1))*A;
s = Am*g_t*cos(2*pi*f_c*(1:T_symbol));

end
```

AWGN_channel.m

```
function r = AWGN_channel(s, SNR,M)

 [L_s, T_symbol] = size(s);
 variance= 1/(2*log2(M)*(10^(SNR/10))); %variance= σ^2

 noise = sqrt(variance)*randn(L_s, T_symbol);
 r = s + noise;

end
```

demodulator.m

```
function r_dem= demodulator(r,M)
T_c= 4;
f_c=1/T_c;
T_symbol=40;

E_s=1;
g_t = sqrt((2 * E_s) / T_symbol);

A=sqrt((M^2-1)/3);
Am_demodulator=((r + (M+1))/2)*A;


r_dem= Am_demodulator*(g_t*cos(2*pi*f_c*(1:T_symbol))');

end
```

detector.m

```
function detector_symbols= detector(r_dem,M)


if M==2
s_m = cos((2*pi*(0:3)')/4);
else
 s_m=(0:7)';
end

vect_dist = zeros(length(s_m), 1);
detector_symbols= zeros(length(r_dem),1);


for j=1:length(r_dem)
    for i=1:length(vect_dist)
        vect_dist(i) = norm(r_dem(j) - s_m(i));
    end
    [~, pos] = min(vect_dist);
    detector_symbols(j) = pos - 1;
end

```

demapper.m

```
function bit_sequence = demapper(detector_symbols,g,M)

    if M==8
        bit_seq_len = 3*length(detector_symbols);
    else
        bit_seq_len = 2*length(detector_symbols);
    end

    if g==1
        symbols = gray2bin(detector_symbols,'pam',M);
    else
        symbols=detector_symbols;
    end

bit_sequence = zeros(bit_seq_len, 1);

j = 1;
%~~~~~~~8-PAM~~~~~~~~~~
if M==8
    for i=1:3:bit_seq_len
        bit_sequence(i)= ceil(rem(symbols(j),2));
        bit_sequence(i+1)= floor((rem(symbols(j),4))/2);
        bit_sequence(i+2)=floor(symbols(j)/4);


        j = j + 1;
    end
 %~~~~~~~2-PAM~~~~~~~~~~
else
    for i=1:2:bit_seq_len
        bit_sequence(i) = floor(symbols(j)/2);
        bit_sequence(i + 1) = rem(symbols(j), 2);

        j = j + 1;
    end
end
end

```

erotimata_merosA_ab.m

(υλοποίηση συστήματος κι υπολογισμός BER)

```
L_b = 10002;
SNR = (0:2:20);
BER_temp_variable = zeros(length(SNR),3);
BER = zeros(length(SNR),3);
bit_tr = randsrc(L_b,1,[0 1]);
symbols = mapper(bit_tr,1,8);  %(bit_tr,gray_or_not,M)
signal_pam = modulator(symbols, 8); %(symbols,M)


%~~~~~~~Ypologismos BER gia 8-PAM Gray encoded symbols~~~~~~~
for i=1:length(SNR)
    signal_tr = AWGN_channel(signal_pam, SNR(i),8);

    signal_dem = demodulator(signal_tr,8);
    symbols_det = detector(signal_dem,8); %(signal_dem,M)

    bit_rc = demapper(symbols_det,1,8); %symbols_det,gray,8-PAM

    BER_temp_variable(i,1) = biterr(bit_tr, bit_rc)/L_b;

    BER(i, 1) =BER_temp_variable(i,1)*(1/(10^(SNR(i)/10)));


end

%~~~~~Ypologismos BER gia 8-PAM xoris Gray encoded symbols~~~~~
symbols=mapper(bit_tr,0,8);
signal_pam = modulator(symbols,8);
for i=1:length(SNR)
    signal_tr = AWGN_channel(signal_pam, SNR(i),8);

    signal_dem = demodulator(signal_tr,8);
    symbols_det = detector(signal_dem,8);

    bit_rc = demapper(symbols_det,0,8); %symbols_det,NO-gray,8-PAM

    BER_temp_variable(i,2) = biterr(bit_tr, bit_rc)/L_b;
    BER(i, 2) =BER_temp_variable(i,2) *(1/(10^(SNR(i)/10)));

end
```

```
41
42
43
44      %~~~~~~~Ypologismos BER gia 2-PAM~~~~~~~
45 -    symbols=mapper(bit_tr,0,2);
46 -    signal_pam = modulator(symbols,2);
47 -    for i=1:length(SNR)
48 -        signal_tr = AWGN_channel(signal_pam, SNR(i),2);
49 -        signal_dem = demodulator(signal_tr,2);
50 -        symbols_det = detector(signal_dem,2);
51
52 -        bit_rc = demapper(symbols_det,0,2); %symbols_det,NO-gray,8-PAM
53
54 -         BER_temp_variable(i,3) = biterr(bit_tr, bit_rc)/L_b;
55 -         BER(i, 3) =BER_temp_variable(i,3) *(1/(10^(SNR(i)/10)));
56
57
58 -    end
59
60
61      %aksonas x grafikis parastasis
62 -    x_axe=zeros(length(SNR),1);
63 -    for k=1:length(SNR)
64 -        x_axe(k)=10*log10(10^(SNR(k)/10));
65 -    end
66
67      %Sxediasmos grafikis anaparastasis
68 -    semilogy(x_axe, BER(:,1),'go-');hold on;
69 -    semilogy(x_axe, BER(:, 2), 'b*-'); hold on;
70 -    semilogy(x_axe, BER(:, 3), 'r--'); hold off;
71 -    grid on;
72
73 -    legend('8-PAM/Gray','8-PAM', '2-PAM');
74
75 -    xlabel('Eb/No (dB)');
76 -    ylabel('Bit Error Rate');
```

erotimata_merosA_c.m

(υλοποίηση συστήματος κι υπολογισμός SER)

```
L_b = 10002;
SNR = (0:2:20);
SER = zeros(length(SNR),2);
SER_temp_variable = zeros(length(SNR),2);
bit_tr = randsrc(L_b,1,[0 1]);


%~~~~~Ypologismos SER gia 8-PAM xoris Gray encoded symbols~~~~~
symbols=mapper(bit_tr,0,8);
signal_pam = modulator(symbols,8);
for i=1:length(SNR)
    signal_tr = AWGN_channel(signal_pam, SNR(i),8);

    signal_dem = demodulator(signal_tr,8);
    symbols_det = detector(signal_dem,8);

    bit_rc = demapper(symbols_det,0,8); %symbols_det,NO-gray,8-PAM

   SER_temp_variable(i,1) = symerr(symbols, symbols_det)/L_b;
   SER(i, 1) =SER_temp_variable(i,1)*(1/(10^(SNR(i)/10)));

end

%~~~~~~~Ypologismos SER gia 2-PAM~~~~~~~
symbols=mapper(bit_tr,0,2);
signal_pam = modulator(symbols,2);
for i=1:length(SNR)
    signal_tr = AWGN_channel(signal_pam, SNR(i),2);

    signal_dem = demodulator(signal_tr,2);
    symbols_det = detector(signal_dem,2);

    bit_rc = demapper(symbols_det,0,2); %symbols_det,NO-gray,8-PAM

       SER_temp_variable(i,2) = symerr(symbols, symbols_det)/L_b;
       SER(i, 2) =SER_temp_variable(i,2)*(1/(10^(SNR(i)/10)));

end

%aksonas x grafikis parastasis
x_axe=zeros(length(SNR),1);
for k=1:length(SNR)
    x_axe(k)=10*log10(10^(SNR(k)/10));
end

%Sxediasmos grafikis anaparastasis
semilogy(x_axe, SER(:,1),'go-');hold on;
semilogy(x_axe, SER(:, 2), 'b*-'); hold off;
grid on;

legend('8-PAM', '2-PAM');

ylabel('Symbol Error Rate');
```

ΜΕΡΟΣ Β

my_quantizer.m

```
function y_q= my_quantizer(y,N,min_value,max_value)

%Epipeda kvantismou
quant_levels=2^N;

%Arxikopoihsh tou y_q
y_q= zeros(length(y),1);

%vima kvantismou Δ
quant_step= (abs(min_value)+max_value)/quant_levels;

%ypologismos twn kentrwn
centers=zeros(quant_levels,1);


centers(1)=max_value-(quant_step/2);
centers(quant_levels)= min_value+(quant_step/2);

    for i= 2: (quant_levels-1)
        centers(i) = centers(i-1)-quant_step;
    end

%Ypologismos simatos eksodou kvantisti

for j=1:length(y)
    if y(j)<= min_value
        y_q(j)=quant_levels;
    elseif y(j)>= max_value
        y_q(j)=1;
    else
        if y(j)<0
            y(j) = max_value + abs(y(j));
        elseif y(j)>=0
            y(j) = max_value - y(j);
        end
        y_q(j) = floor(y(j)/quant_step)+1;
    end

y_q(j)=centers(y_q(j));

```

my_DPCM.m

```
function y_rec=my_DPCM(p,N)
%my_DPCM (p,N,e) To e einai o arithmos erotimatos
x1= load('source.mat');
x=x1.t;

len_x=length(x);

r=zeros(length(p),1);
%Dianysma aftosysxetisis
for i=1:p
    sum= 0;
    for n=p+1:len_x
        sum=sum+ x(n)*x(n-i);
    end
    r(i)=(1/(len_x-p))*sum;
end


%Pinakas aftosysxetisis
R=zeros(length(p),length(p));
for i=1:p
        for j=1:p
            sum=0;
            for n=p:len_x
            sum= sum+ x(n-j+1)*x(n-i+1);
            end
         R(i,j)=(1/(len_x-p+1))*sum;
        end
end

%syntelestes filtrou provlepsis
a=R\r';

%kvantisi syntelestwn
a_quantum=my_quantizer(a,8,-2,2)';


mem(1:p)=x(1:p)';

%filtro provlepsis
pred= zeros(len_x,1);   %provlepsi deigmatos
err=zeros(len_x,1);      %sfalma provlepsis
err_q=zeros(len_x,1);%kvantisi sfalmatos
%y_rec=zeros(N,1); %anakataskevasmeno sima
for j= p+1:len_x
    sum=0;
        for i=1:p
            sum= sum+ a_quantum(i)* mem(j-i);
        end

```

```
51          %Provlepsi deigmatos
52 -        pred(j)=sum;
53
54          %Ypologismos sfalmatos provlepsis
55 -        err(j)=x(j)- pred(j);
56
57          %Kvantopoihsh tou sfalmatos provlepsis
58 -        err_q(j)= my_quantizer(err(j),N,-3.5,3.5)';
59
60          %anakataskevi tou deigmatos sto dekti
61 -        y_rec(j)=err_q(j)+pred(j);
62 -        mem(j)=y_rec(j);
63 -      end
64
65
66      %plot~~~~~ erotima 2
67 -     plot(x,'b--'); hold on;
68 -     plot(err,'r:');hold off;
69 -     grid on;
70 -     legend('Input Signal','Prediction Error');
71
72      % plot~~~~~erotima 4
73 -    plot(x,'b--'); hold on;
74 -    plot( y_rec,'m--');hold off;
75 -    grid on;
76 -    legend('Input Signal','Rebuild signal');
77
78 -    end
```

erotima_3.m

```
x1= load('source.mat');
x=x1.t;
%~~~~~~~~~~~~~p=5~~~~~~~~~~~~~
ya=zeros(20000,3);
i=0;
for N=1:3
    i=i+1;
    ya(:,i)=my_DPCM (5,N);
end
mtsp1=zeros(3,1);
for k=1:3
    for j=1:length(x)
        ya(j,k)=((ya(j,k)-x(j))^2);
    end

end
 mtsp1=sum(ya,1)./length(x);

%~~~~~~~~~~~~~p=6~~~~~~~~~~~~~
 yb=zeros(20000,3);
 i=0;
for N=1:3
    i=i+1;
    yb(:,i)=my_DPCM (6,N);
end
mtsp2=zeros(3,1);
for k=1:3
    for j=1:length(x)
        yb(j,k)=((yb(j,k)-x(j))^2);
    end

end
 mtsp2=sum(yb,1)./length(x);

%~~~~~~~~~~~~~p=7~~~~~~~~~~~~~
 yc=zeros(20000,3);
 i=0;
for N=1:3
    i=i+1;
    yc(:,i)=my_DPCM (7,N);
end
mtsp3=zeros(3,1);
for k=1:3
    for j=1:length(x)
        yc(j,k)=((yc(j,k)-x(j))^2);
    end

end
 mtsp3=sum(yc,1)./length(x);

```

```
    %~~~~~~~~~~~~~~p=8~~~~~~~~~~~~~~
     yd=zeros(20000,3);
     i=0;
    for N=1:3
        i=i+1;
        yd(:,i)=my_DPCM (8,N);
    end
    mtsp4=zeros(3,1);
    for k=1:3
        for j=1:length(x)
            yd(j,k)=((yd(j,k)-x(j))^2);
        end

    end
     mtsp4=sum(yd,1)./length(x);

     %~~~~~~~~~~~~~~p=9~~~~~~~~~~~~~~
      ye=zeros(20000,3);
     i=0;
    for N=1:3
        i=i+1;
        ye(:,i)=my_DPCM (9,N);
    end
    mtsp5=zeros(3,1);
    for k=1:3
        for j=1:length(x)
            ye(j,k)=((ye(j,k)-x(j))^2);
        end

    end
     mtsp5=sum(ye,1)./length(x);

    %~~~~~~~~~~~~~~p=10~~~~~~~~~~~~~~
      yf=zeros(20000,3);
     i=0;
    for N=1:3
        i=i+1;
        yf(:,i)=my_DPCM (10,N);
    end
    mtsp6=zeros(3,1);
    for k=1:3
        for j=1:length(x)
            yf(j,k)=((yf(j,k)-x(j))^2);
        end

    end
     mtsp6=sum(yf,1)./length(x);

    %~~~~~~~~~~~~~~plot~~~~~~~~~~~~~~
     plot(mtsp1,'b-');hold on;
     plot(mtsp2,'g-');hold on;
     plot(mtsp3,'r-');hold on;
     plot(mtsp4,'k-');hold on;
     plot(mtsp5,'m-');hold on;
     plot(mtsp6,'y-');hold off;
     grid on;

    legend('p=5','p=6','p=7','p=8','p=9','p=10');
    xlabel('N');
    ylabel('Μέσο τετραγωνικό σφάλμα πρόβλεψης');

```