ΨΗΦΙΑΚΕΣ ΤΗΛΕΠΙΚΟΙΝΩΝΙΕΣ

2η Εργαστηριακή Ασκηση

Καλάργαρης Χαράλαμπος, ΑΜ:3929

27/2/2011

Contents

1 M-PSK

Ο πομπός του συστήματος PSK δέχεται ως είσοδο μια δυαδική ακολουθία, τη μετατρέπει σε σύμβολα, την 1 πολλαπλασιάζει με τον ορθογώνιο παλμό, και κατόπιν το σήμα μεταφέρεται στη ζώνη μετάδοσης μέσω του διαμορφωτή. Το σήμα διέρχεται μέσα από κανάλι AWGN και φθάνει στο δέκτη του συστήματος. Εκεί αποδιαμορφώνεται και προκύπτει ένα δισδιάστατο διάνυσμα, το οποίο εισάγεται στο φωρατή όπου και αποφασίζεται ποιο σύμβολο στάλθηκε. Τέλος, ο demapper κάνει την αντίστροφη αντιστοίχηση από σύμβολα σε bits. Τα συστήματα αυτά περιγράφονται στη συνέχεια.

1.1 Mapper

Ο Mapper χρησιμοποιείται για να αντιστοίχίζει bits σε σύμβολα.Οπως γινετε εύκολα αντιληπτό για M=2 χρειαζόμαστε 1 bit για κάθε σύμβολο, ενώ για M=4 χρειαζόμαστε 2 bits για κάθε σύμβολο. Επομένος για M=2 δεν χρειαζεται Mapper αφου υπαρχει αντιστοίχηση συμβόλου με bits.Το παρακάτω πρόγραμμα δειχνει την λειτουργια του Mapper για M=4 με επιλογή της κωδικοποίσης που θα κάνουμε.Η κωδικωποίηση μπορεί να είναι είτε Gray είτε κάποια άλλη.

```
function [symbol] = Mapper(bits, choose)
% Gray choose=1
% other tyoe of coding choose=0
if choose == 1
    if ( bits(1)==0 && bits(2)==0 )
        symbol = 0;
    elseif ( bits(1)==0 && bits(2)==1 )
        symbol = 1;
    elseif ( bits(1)==1 && bits(2)==1 )
        symbol = 2;
    elseif ( bits(1)==1 && bits(2)==0 )
        symbol = 3;
    end
elseif choose == 2
    if ( bits(1)==0 && bits(2)==0)
        symbol = 0;
    elseif ( bits(1)==0 && bits(2)==1 )
        symbol = 1;
    elseif ( bits(1)==1 && bits(2)==0 )
        symbol = 2;
    elseif ( bits(1)==1 && bits(2)==1 )
        symbol = 3;
    end

end

end
```

1.2 Διαμορφωτής

Η λειτουργία του διαμορφωτή είναι να δημιουργεί το ζωνοπερατό σημα για κάθε σύμβολο.Ο τύπος που μας δίνει το ζωνοπερατό σημα φαινεται παρακατω($E_s = 1$):

$$s_m(t) = \cos(\tfrac{2\pi m}{M}) g_T(t) \cos(2\pi f_c t) + \sin(\tfrac{2\pi m}{M}) g_T(t) \sin(2\pi f_c t), 0 \leq t \leq T_{symbol}$$

Να σημειωθει οτι $g_t(t)$ ειναι ο παλμός που χρησιμοποιούμε.Είναι ορθογώνιος παλμός και οριζεται ως ($E_s = 1$):

$$g_T(t) = \begin{cases} \sqrt{\frac{2E_s}{T_{symbol}}} = \sqrt{\frac{2}{T_{symbol}}} & : \quad 0 \leq t \leq T_{symbol} \\ 0 & : \quad other \end{cases}$$

Παρακάτω βρίσκετε ο κώδικας Matlab που υλοποιεί τον διαμορφωτή.

```
%symbol coming from fuction Mapper.m
% M the type of "M"-PSK system
% M = 2 => 2-PSK
% M = 4 => 4-PSK
function [mod] = PSK_Modulator(symbol, M)

%--------Initialization-----------------------

%time-period-frequency
T_sample = 1;
T_c = 4;
T_s = 40;
f_c = 1/T_c;
t = [0 : T_sample : T_s*T_sample-1];
%normalized energy
E_s = 1;
%basic pulse
g_T = sqrt(2*E_s/T_s);

%-----------------------------------------------

% --------Modulator-----------------------------

if M == 2
   % 2-PSK base-fuction
    base(1) = sqrt(E_s)*cos(2*pi*symbol/2);
    base(2) = sqrt(E_s)*sin(2*pi*symbol/2);
else
   % 4-PSK base fuction
    base(1) = sqrt(E_s)*cos(2*pi*symbol/4);
    base(2) = sqrt(E_s)*sin(2*pi*symbol/4);
end



% base-fuction * base-pulse
baseband_signal = zeros(T_s,2);
for i = 1 : T_s
    baseband_signal(i,1) = g_T * base(1);
    baseband_signal(i,2) = g_T * base(2);
end



% 1st signal.
carrier(:,1) = cos(2*pi*f_c.*t);
% 2nd signal
carrier(:,2) = sin(2*pi*f_c.*t);
% final singal
car = carrier.*baseband_signal;
mod(:,1)=car(:,1) + car(:,2);

```

```
55 %-------------------------------------------------------------------
56
57 end
```

1.3 AWGN Κανάλι

Τα ζωνοπετατό σήμα που εκπέμπει ο πομπός των συστημάτων διέρχεται μέσα από ένα ιδανικό κανάλι προσθετικού θορύβου. Ο θόρυβος είναι λευκός και ακολουθεί Gaussian κατανομή μηδενικής μέσης τιμής και διασποράς σ2=N0/2.

Να σημειωθει οτι χρησιμοποιείθηκε η συνάρτηση normrnd() της Matlab για την Gauss κατανομή.

```
1  %input signal coming from Modulator
2  %SNR= [0:2:16]
3  % M the type of "M"-PSK system
4  % M = 2 => 2-xSK
5  % M = 4 => 4-xSK
6  function [output_signal] = AWGN_channel(input_signal, SNR, M )
7
8  %symbol energy normalized
9  E_s = 1;
10
11 if M == 2
12     %2-xSK
13     E_b = E_s;
14 else
15     %4-xSK
16     E_b = E_s/2;
17 end
18
19 %Gaussian noise
20 N_0 = E_b / (10^(SNR*0.1));
21 Noise = normrnd(0,sqrt(N_0/2),40,1) ;
22
23 % output singal with Gaussian noise
24 output_signal(:,1) = input_signal(:,1) + Noise(:,1);
25 end
```

1.4 Αποδιαμορφωτής

Ο αποδιαμορφωτής του συστήματος PSK συσχετίζει (δηλαδή πολλαπλασιάζει και ολοκληρώνει-αθροίζει) το ληφθέν σήμα με τη φέρουσα και τον ορθογώνιο παλμό. Η συσχέτιση γίνεται στα χρονικά πλαίσια μιας περιόδου συμβόλου. Κατά την προσομοίωση υποθέτουμε ότι τόσο το PSK, όσο και το FSK είναι ομόδυνα (coherent). Αυτό σημαίνει ότι ο δέκτης γνωρίζει τη φάση της φέρουσας και τα χρονικά πλαίσια κάθε συμβόλου, δηλαδή είναι πλήρως συγχρονισμένος με τον πομπό. Ο αποδιαμορφωτής συσχετίζει το ληφθέν σήμα με τις δύο συνιστώσες της φέρουσας, οπότε προκύπτουν δύο τιμές, δηλαδή ένα διάνυσμα r που είναι η εκτιμηθείσα τιμή του τρέχοντος συμβόλου πάνω στον αστερισμό του PSK.

Παρακάτω βρίσκεται ο κώδικας Matlab που υλοποιεί τον αποδιαμορφωτή. Λαμβάνει το σήμα απο το AWGN Κανάλι επιστρέφει το αντίστοιχο διάνυσμα r.

```
1  % received_signal coming from AWGN channel
```

```
function [vector_r] = PSK_Demodulator(received_signal)

%--------Initialization-----------------------------

%time-period-frequency
T_sample = 1;
T_c = 4;
T_s = 40;
f_c = 1/T_c;
t = [0 : T_sample : T_s*T_sample-1];
%normalized energy
E_s = 1;
%basic pulse
g_T = sqrt(2*E_s/T_s);

%-------------------------------------------------


% received singal * basic pulse
r_t = g_T .* received_signal;
% multiply with basic fuctions
r(:,1) = r_t(:,1) .* cos(2*pi*f_c.*t');
r(:,2) = r_t(:,1) .* sin(2*pi*f_c.*t');

% final vector
vector_r(1) = 0;
vector_r(2) = 0;
for k = 1 : T_s
    vector_r(1) = vector_r(1) + r(k,1);
    vector_r(2) = vector_r(2) + r(k,2);
end

end
```

1.5 Φωρατής

Ο φωρατής δέχεται ως είσοδο το διάνυσμα r, και αποφασίζει σε ποιο σύμβολο (όπως αυτά ορίστηκαν διανυσματικά παραπάνω) βρίσκεται εγγύτερα. Το διάνυσμα s_m που θα έχει τη μικρότερη απόσταση από το r, αντιστοιχεί και στο σύμβολο που στάλθηκε.

Παρακάτω βρίσκεται ο κώδικας Matlab που υλοποιεί τον φωρατή.Αρχικά υπολογίζουμε ολα τα s_m και μετα επιλέγουμε αυτο με την μικρότερη απόσταση απο το διάνυσμα r.

```
%vector_r is coming from demodulator_PSK fuction
% M the type of "M"-PSK system
% M = 2 => 2-PSK
% M = 4 => 4-PSK
function [symbol_decision] = PSK_Detector(vector_r,M)

%--------Initialization-----------------------------

%time-period-frequency
T_sample = 1;
T_c = 4;
T_s = 40;
f_c = 1/T_c;
%normalized energy
```

```
E_s = 1;


%-------------------------------------------------

%where is the symbol
for k = 0 : M-1
    symbols(k+1,1) = sqrt(E_s) * cos(2*pi*k/M );
    symbols(k+1,2) = sqrt(E_s) * sin(2*pi*k/M );
end

% symbol distance
for k = 1 : M
    Dist(k,1) = sqrt( (vector_r(1,1) - symbols(k,1))^2 + (vector_r(1,2) - symbols(k,2))^2
end

% smallest symbol distance
temp = Dist(1,1);
symbol_decision = 1;
for k = 2 : M
    if temp > Dist(k,1)
        temp = Dist(k,1);
        symbol_decision = k;
    end
end

%final symbol
symbol_decision = symbol_decision - 1;

end
```

1.6 Demapper

Ο Demapper κανει την αντίστροφη λειτουργία απο τον Mapper. Δηλαδή μετατρέπει τα σύμβολα σε bits. Παρακάτω φαινεται ο κώδικας Matlab που το υλοποεί.

```
% Gray choose=1
% other tyoe of coding choose=0
function [bits] = Demapper(symbol, choose)

if choose == 1
    if symbol == 0
        bits = [0 0];
    elseif symbol == 1
        bits = [0 1];
    elseif symbol == 2
        bits = [1 1];
    elseif symbol == 3
        bits = [1 0];
    end

elseif choose == 2
    if symbol == 0
        bits = [0 0];
    elseif symbol == 1
        bits = [0 1];
    elseif symbol == 2
        bits = [1 0];
    elseif symbol == 3
```

```
24            bits = [1  1];
25        end
26
27    end
28
29    end
```

1.7 Matlab PSK σύστημα

Για να προσομοιώσουμε το PSK σύστημα χρησιμοποιούμε σαν εσωτερικές συναρτησεις οτι σχεδιασαμε πιο πανω για τις εσωτερικές του μονάδες (Mapper,Διαμορφωτής,AWGN channel,Αποδιαμορφωτής,Φωρατης, Demapper).Επίσης για να είναι πιο παραμετροποιήσημο το προγραμμα αυτό παιρνει σαν είσοδο το M και το choose οπου:

- M : Ορίζουμε το είδος του συστήματος
  - 2 : 2-PSK
  - 4 : 4-PSK
- choose : Είδος κωδικοποίησης.
  - 1 : Κωδικοποίηση Gray
  - 2 : Αλλη κωδικοποίηση

Επίσης αρχικά το σύστημα δημιουργεί σαν είσοδο μια ακολουθία bits, όπου οι τιμές 0 και 1 εμφανίζονται ισοπίθανα.Το πλήθος των bits πρέπει είναι της τάξης των Lb=10000-100000 bits (διάλεξα τυχαία 75000 bits).Τέλος υπολογίζουμε και το BER που θα χρειαστει για το 3ο ερώτημα για $SNR = [0 : 2 : 16]dB$.

```
1  % M the type of "M"-PSK system
2  % M = 2 => 2-PSK
3  % M = 4 => 4-PSK
4  % Gray choose=1
5  % other tyoe of coding choose=0
6  function [BER] = PSK_System(choose, M )
7
8  %normalized energy
9  E_s = 1;
10
11 % bits per symbol
12 if M == 2
13     N = 1;
14 else
15     N = 2;
16 end
17
18 % for SNR = 0, 2, 4, 6, 8, 10, 12, 14, 16 dB.
19 for SNR = 0:2:16
20
21 % total bits between 10000-100000
22 total_bits = 75000;
23 total_symbols = total_bits / N;
24
25 % create random bits
26 input_bits = randsrc(1,total_bits,[0,1]);
```

```
27
28 % bits which transmited (counter)
29 input_bits_counter = 1;
30 % bits which received (counter)
31 output_bits_counter = 1;
32
33 for j = 1 : total_symbols
34     %group for bits per symbol
35     for i = 1 : N
36         symbol_bits(1,i) = input_bits(1,input_bits_counter );
37         input_bits_counter =  input_bits_counter  + 1;
38     end
39
40 %-------------PSK System-------------------------------
41
42     % -------Mapper-----------
43     if M  == 4
44         symbol = Mapper(symbol_bits, choose);
45     else
46         symbol = symbol_bits(1,1);
47     end
48     input_symbol(j) = symbol;
49     %-------------------------
50
51     %-----Modulator PSK--------
52     [transmitted_signal] = PSK_Modulator(symbol, M );
53     %-------------------------
54
55     %-----AWGN Channel--------
56     [received_signal] = AWGN_channel(transmitted_signal, SNR, M );
57     %-------------------------
58
59     %-----Demodulator PSK-----
60     [vector_r] = PSK_Demodulator(received_signal);
61     %-------------------------
62
63     %-----Detector PSK--------
64     [symbol_decision] = PSK_Detector(vector_r, M );
65     output_symbol(j) = symbol_decision;
66     %-------------------------
67
68     %-----Demapper-----------
69     if M  == 4
70         [bits_decision] = Demapper(symbol_decision, choose);
71     else
72         bits_decision = symbol_decision;
73     end
74     %-------------------------
75
76 %-----------------------------------------------------
77
78
79     % final output of bits
80     for i = 1 : N
81         output_bits(1,output_bits_counter) = bits_decision(1,i);
82         output_bits_counter = output_bits_counter + 1;
83     end
84 end
85
86
87     % error bits counter
88     error_bits = 0;
```

```
89      for j = 1 : total_bits
90          if input_bits(j) ≠ output_bits(j)
91              error_bits = error_bits + 1;
92          end
93      end
94
95      % Bit Error Rate - BER
96      BER(SNR/2 +1) = error_bits / total_bits;
97
98  end
99
100 end
```

2 M-FSK

2.1 Mapper

Ομοια με τον Mapper του PSK .

2.2 Διαμορφωτής

Η λειτουργία του διαμορφωτή είναι να δημιουργεί το ζωνοπερατό σημα για κάθε σύμβολο.Ο τύπος που μας δίνει το ζωνοπερατό σημα φαινεται παρακατω:

$$s_m(t) = \cos(2\pi(f_c + \frac{m}{T_{symbol}})t)g_T(t), m = 0, ...M-1$$

Παρακάτω βρίσκετε ο κώδικας Matlab που υλοποιεί τον διαμορφωτή.Να σημειωθει οτι $g_t(t)$ ειναι ο παλμός που χρησιμοποιούμε και φαίνεται **ΕΔΩ**.

```
1  %symbol coming from fuction Mapper.m
2  function [mod] = FSK_Modulator(symbol)
3
4  %--------Initialization ----------------------------
5
6  %time-period-frequency
7   T_sample = 1;
8   T_c = 4;
9   T_s = 40;
10  f_c = 1/T_c;
11  t = [0 : T_sample : T_s*T_sample-1];
12  Df = 1 / T_s; %diafora metaksi geitwniko f
13  %normalized energy
14  E_s = 1;
15  %basic pulse
16  g_T = sqrt(2*E_s/T_s);
17
18  %-------------------------------------------------
19
20
21  % symbol frequency
22    f = f_c + symbol * Df;
23
24  % ---------Modulator-----------------------------
25   carrier = cos(2*pi*f.*t');
26  mod = g_T .* carrier;
27  %-------------------------------------------------
28  end
```

2.3 AWGN Κανάλι

Ομοια με τον AWGN Κανάλι του PSK .

2.4 Αποδιαμορφωτής

Παρακάτω βρίσκεται ο κώδικας Matlab που υλοποιεί τον αποδιαμορφωτή. Λαμβάνει το σήμα απο το AWGN Κανάλι επιστρέφει το αντίστοιχο διάνυσμα r.

```
% received_signal coming from AWGN channel
% M the type of "M"-FSK system
% M = 2 => 2-FSK
% M = 4 => 4-FSK
function [vector_r] = FSK_Demodulator(received_signal, M)


%----------Initialization----------------------------

%time-period-frequency
T_sample = 1;
T_c = 4;
T_s = 40;
f_c = 1/T_c;
t = [0 : T_sample : T_s*T_sample-1];
Df = 1 / T_s; %diafora metaksi geitwniko f
%normalized energy
E_s = 1;
%basic pulse
g_T = sqrt(2*E_s/T_s);

%----------------------------------------------------


pulse = g_T .* received_signal;

%----------------Demodulator-------------------------
temp = 0;
for m = 0 : M-1
    f = f_c + Df*m;
    tmp = (cos(2*pi*f.*t')).*pulse;
    tmp = tmp';
    r2(m+1,:) = tmp(1,:);
end

for i=1:M
    for j = 1:40
        temp = temp + r2(i,j);
    end
    vector_r(i,1) = temp;
    temp = 0;
end
%----------------------------------------------------
end
```

2.5 Φωρατής

Καθένας από τους δύο ή τέσσερεις κλάδους συσχέτισης του δέκτη παράγει μια τιμή ri. Αυτές δίνονται στο φωρατή, και όποια έχει τη μεγαλύτερη τιμή, επιλέγεται

το αντίστοιχο σύμβολο ως εκτιμηθέν.

Παρακάτω βρίσκεται ο κώδικας Matlab που υλοποιεί τον φωρατή.

```
%vector_r is coming from demodulator_FSK fuction
% M the type of "M"-FSK system
% M = 2 => 2-FSK
% M = 4 => 4-FSK
function [symbol_decision] = FSK_Detector(vector_r, M)

% smallest symbol distance
temp = vector_r(1,1);
count = 0;
for i = 2 : M
    if temp < vector_r(i,1)
        temp = vector_r(i,1);
        count = i - 1;
    end
end
%final symbol
symbol_decision = count;

end
```

2.6 Demapper

Ομοια με τον Demapper του PSK .

2.7 Matlab FSK σύστημα

Για να προσομοιώσουμε το FSK σύστημα χρησιμοποιούμε σαν εσωτερικές συναρτησεις οτι σχεδιασαμε πιο πανω για τις εσωτερικές του μονάδες (Mapper,Διαμορφωτής,AWGN channel,Αποδιαμορφωτής,Φωρατης, Demapper).Επίσης για να είναι πιο παραμετροποιήσημο το προγραμμα αυτό παιρνει σαν είσοδο το M:

- M : Ορίζουμε το είδος του συστήματος
  - 2 : 2-PSK
  - 4 : 4-PSK

Στην περίπτωση των συστημάτων FSK δεν παίζει ρόλο ο τρόπος κωδικοποίησης των συμβόλων, επειδή όλα τα σύμβολα στο χώρο σημάτων έχουν ίσες αποστάσεις μεταξύ τους.Ϊρα δεν χρεαζεται μεταβλητη για είδος κωδικοποίησης.

Επίσης αρχικά το σύστημα δημιουργεί σαν είσοδο μια ακολουθία bits, όπου οι τιμές 0 και 1 εμφανίζονται ισοπίθανα.Το πλήθος των bits πρέπει είναι της τάξης των Lb=10000-100000 bits (διάλεξα τυχαία 75000 bits).Τέλος υπολογίζουμε και το BER που θα χρειαστει για το 3ο ερώτημα για $SNR = [0 : 2 : 16]dB$.

```
% M the type of "M"-FSK system
% M = 2 => 2-FSK
% M = 4 => 4-FSK
function [BER] = FSK_System(M)

%normalized energy
E_s = 1;
```

```
8
9  % bits per symbol
10 if M == 2
11    N = 1;
12 else
13    N = 2;
14 end
15
16 % for SNR = 0, 2, 4, 6, 8, 10, 12, 14, 16 dB.
17 for SNR = 0:2:16
18
19 % total bits between 10000-100000
20 total_bits = 75000;
21 total_symbols = total_bits / N;
22
23 % create random bits
24 input_bits = randsrc(1,total_bits,[0,1]);
25
26 % bits which transmited (counter)
27 input_bits_counter = 1;
28 % bits which received (counter)
29 output_bits_counter = 1;
30 for j = 1 : total_symbols
31     %group for bits per symbol
32     for i = 1 : N
33         symbol_bits(1,i) = input_bits(1,input_bits_counter );
34         input_bits_counter =  input_bits_counter  + 1;
35     end
36
37 %--------------PSK System-------------------------------
38
39     %--------Mapper----------------
40     if M == 4
41     symbol = Mapper(symbol_bits,1);
42     else
43         symbol = symbol_bits(1,1);
44     end
45     input_symbol(j) = symbol;
46     %--------------------------------
47
48
49     %--------Modulator-------------
50     [transmitted_signal] = FSK_Modulator(symbol);
51     %--------------------------------
52
53     %--------AWGN channel----------
54     [received_signal] = AWGN_channel(transmitted_signal, SNR, M);
55     %--------------------------------
56
57     %--------Demodulator-----------
58     [vector_r] = FSK_Demodulator(received_signal, M);
59     %--------------------------------
60
61     %--------Detector--------------
62     [symbol_decision] = FSK_Detector(vector_r, M);
63     output_symbol(j) = symbol_decision;
64     %--------------------------------
65
66     %--------Demapper--------------
67     if M == 4
68        [bits_decision] = Demapper(symbol_decision, 1);
69     else
```

```
70          bits_decision = symbol_decision;
71      end
72      %-------------------------
73
74
75  %-------------------------------------------------
76
77      % final output of bits
78      for i = 1 : N
79          output_bits(1,output_bits_counter) = bits_decision(1,i);
80          output_bits_counter = output_bits_counter + 1;
81      end
82  end
83
84      % error bits counter
85      error_bits = 0;
86      for j = 1 : total_bits
87          if input_bits(j) ≠ output_bits(j)
88              error_bits = error_bits + 1;
89          end
90      end
91
92      % Bit Error Rate - BER
93      BER(SNR/2 +1) = error_bits / total_bits;
94
95  end
96
97
98  end
```

3 Ερώτημα 3 - BER

Για καθένα από τα δύο ζεύγη συστημάτων, μετρήστε την πιθανότητα σφάλματος και σχεδιάστε τις καμπύλες BER για τιμές του SNR=[0:2:16]dB. Οι καμπύλες BER θα πρέπει να σχεδιαστούν στο ίδιο γράφημα. Σχολιάστε τα αποτελέσματα. Ποιο σύστημα είναι καλύτερο ως προς την πιθανότητα σφάλματος για το ίδιο SNR; Πόσο παραπάνω SNR απαιτείται για να έχει το χειρότερο την ίδια πιθανότητα σφάλματος με το καλύτερο;

Για να υπολογίσουμε το BER χρησιμοποισούμε τα προγράμματα που σχεδιασαμε πιο πανω (Matlab PSK σύστημα Matlab FSK σύστημα).

Τα διαγράμματα με τις καμπύλες BER δημιουργούνται με τον παρακάτω κώδκα Matlab.

```
1  %Erwthma 3o
2  function Results_3()
3
4  SNR= [0:2:16];
5
6  % 2-PSK kampili
7  [BER1] = PSK_System(1,2)
8  figure(1);
9  semilogy(SNR, BER1,'gs-','LineWidth',2,'MarkerSize',8)
10 %diagramm's title
11 title('BER-dB : 2-PSK, 4-PSK, 2-FSK, 4-FSK');
12
13 % 4-PSK kampili
```

```
14 [BER2] = PSK_System(1,4)
15 hold on;
16 semilogy(SNR, BER2,'cd-','LineWidth',2,'MarkerSize',8)
17
18 % 2-FSK kampili
19 [BER3] = FSK_System(2)
20 hold on;
21 semilogy(SNR, BER3,'ko-','LineWidth',2,'MarkerSize',8)
22
23 % 4-FSK kampili
24 [BER4] = FSK_System(4)
25 hold on;
26 semilogy(SNR, BER4,'bx-','LineWidth',2,'MarkerSize',8)
27
28
29 xlabel('SNR (dB)');
30 ylabel('Bit Error Rate (BER)');
31 legend('2-PSK', '4-PSK','2-FSK', '4-FSK');
32
33 end
```

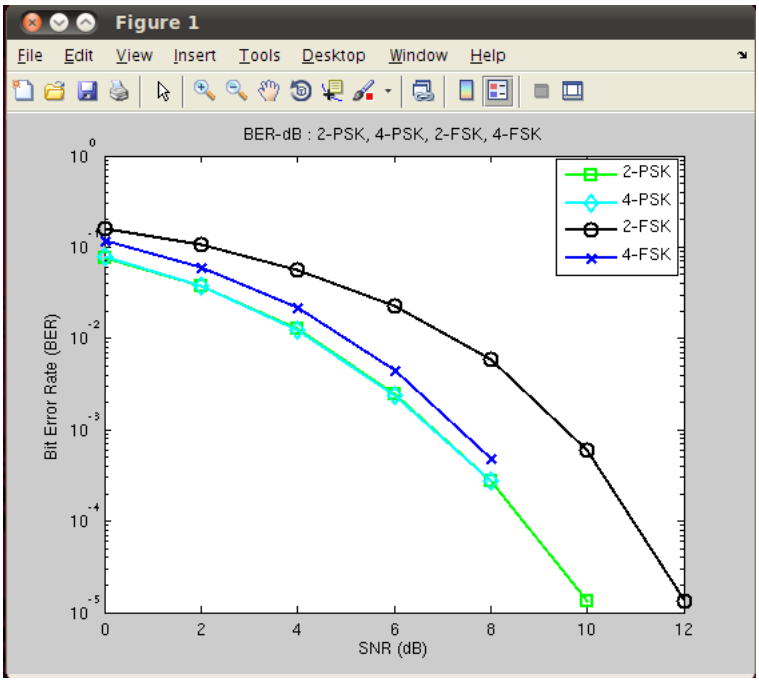

Figure 1: Διάγραμμα για καμπύλες BER-dB

Συμπεράσματα:

- Σχολιάστε τα αποτελέσματα

  Παρατηρούμε οτι οι καμπύλες των 2-PSK και 4-PSK σχεδον ταυτιζονται.Παραλληλα βλέπουμε οτι το χειρότερο συστήμα ως προς την πιθανότητα σφάλματος ειναι το 2-FSK ενω αντίθετα το 4-FSK ειναι αρκετα κοντα στα συστήματα PSK αφου γνωρίζουμε και απο θεωρία οτι οταν αυξάνεται το M στα M-FSK συστήματα μειώνετε και το BER.

- Ποιο σύστημα είναι καλύτερο ως προς την πιθανότητα σφάλματος για το ίδιο SNR;

  Παρατηρώντας το σχημα 1 βλεπουμε οτι για το ιδιο SNR το καλύτερο BER το εχουν τα PSK συστήματα.

- Πόσο παραπάνω SNR απαιτείται για να έχει το χειρότερο την ίδια πιθανότητα σφάλματος με το καλύτερο;

  Παρατηρώντας προσεκτικα το σχημα 1 και κυρίως τις καμπύλες του 2-PSK και 2-FSK βλεπουμε οτι για περίπου 3,5-4 dB το BER του 2-FSK ειναι ιδιο με το BER του 2-PSK.

4 Ερώτημα 4

Πώς επηρεάζει η αύξηση του M τη συμπεριφορά των δύο συστημάτων;

Παρατηρούμε οτι για τα M-PSK συστήματα η αυξηση M οδηγει σε παρόμοια μειωση του BER. Κάτι που ειναι αναμενόμενο και από θεωρία, αφου θα έπρεπε να έχουν ίδιο BER για ίδιο M.Από την άλλη για τα M-FSK συστήματα η αυξηση M οδηγεί σε πολύ φανερή μείωση του BER.

5 Ερώτημα 5

Το πρόγραμμα Matlab που μας δινει τα διαγράμματα για τις καμπύλη BER για το 4-PSK με κωδικοποιήση Gray, 4-PSK χωρις Gray και τον θεωρητικό υπολογισμό της πιθανότητας σφάλματος για το 4-PSK φαινετα παρακατω.

```
%Erwthma 5o
function Results_5()

SNR= [0:2:16];


%4-PSK Gray.
figure(2);
[BER1] = PSK_System(1,4)
semilogy(SNR, BER1,'rx-','LineWidth',2,'MarkerSize',8)
%diagramm's title
title(' BER-dB 4-PSK with Gray, without Gray, Theoritical');

%4-PSK without Gray.
[BER2] = PSK_System(2,4)
hold on;
semilogy(SNR, BER2,'gs-','LineWidth',2,'MarkerSize',8)

%4-PSK theoritical
BER3 = BER_theoritical
hold on;
semilogy(SNR, BER3,'bd-','LineWidth',2,'MarkerSize',8)

xlabel('SNR (dB)');
ylabel('Bit Error Rate (BER)');
legend('4-PSK with Gray', '4-PSK without Gray', '4-PSK theoritical');


end
```

Για τον θεωρητικό υπολογισμό της πιθανότητας σφάλματος για το 4-PSK υλοποιήθηκε η παρακάτω συνάρτηση.

```
%theoritical BER
function ber_theor = BER_theoritical
for k=0:8
    snr_db = 2*k;
    snr = 10^(snr_db*0.1);
    ber_theor(k+1) = 0.5*erfc(sqrt(snr));
end

end
```

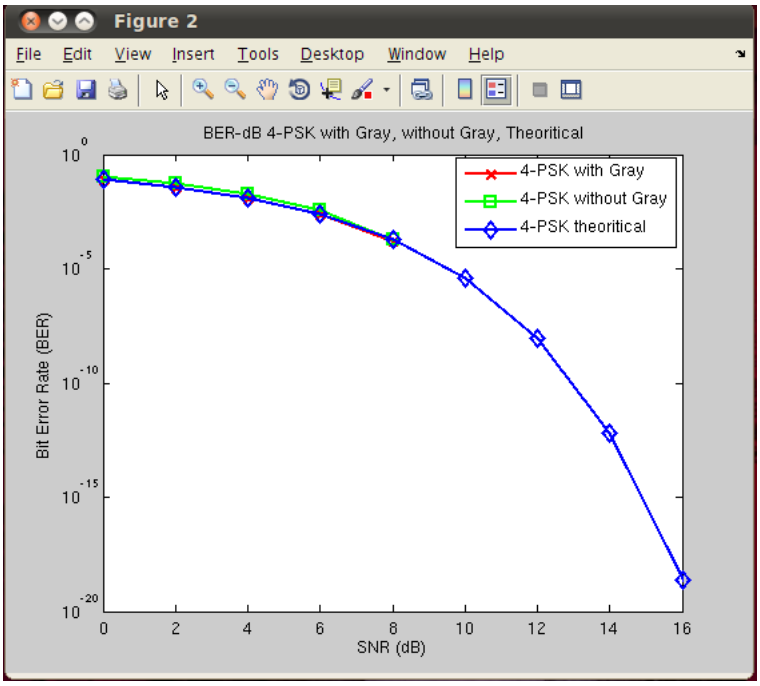

Figure 2: Διάγραμμα για καμπύλες BER-dB με Gray και χωρις

Δυο ειναι τα κύριες παρατηρησεις που εξάγονται απο το παραπάνω σχημα.

1. Οτι οι θεωρητικοι υπολογισμοι ειναι πολυ κοντα στις υλοποιήσεις που πραγματοποιήσαμε.

2. Οτι χρησιμοποιώντας κωδικοποίηση Gray εχουμε ελαφρώς μικροτερο BER (πιθανότητα εμφάνισης σφάλματος bit) αφου οι κωδικοποίησεις γειτονικών συμβόλων διαφερουν κατα 1 bit μονο.Επίσης οσο μικροτερο SNR εχουμε τοσο καλύτερα αποδιδει το συστήμα με κωδικοποιήση Gray.