# Ψηφιακές Επικοινωνίες

## 3η Εργαστηριακή Άσκηση

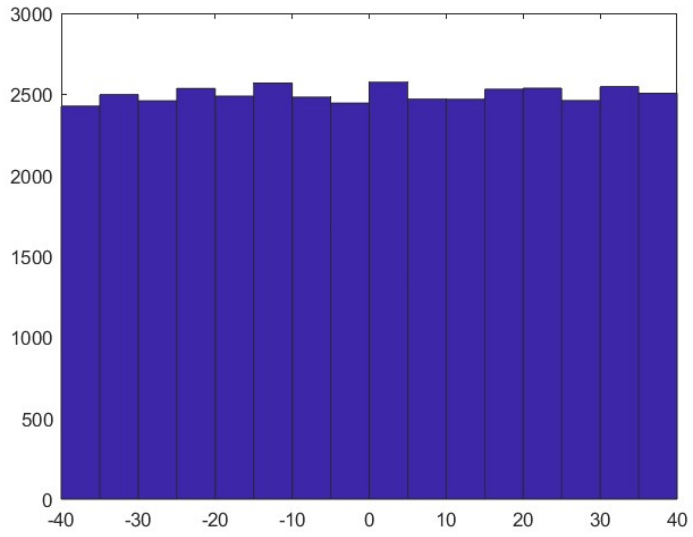
*Άννα Κουτσώνη-03120019*

### Μέρος 1ο

a) Ο τροποποιημένος κώδικας που χρησιμοποιούμε είναι ο εξής:

```
k=mod(20019,2)+3;
L=2^k;
Nsymb=40000;
d=5;
EbNo=20;
nsamp=20;
x=(2*floor(L*rand(1,Nsymb))-L+1)*(d/2);
Px=(L^2-1)*(d^2)/4/3; % θεωρητική ισχύς σήματος
sum(x.^2)/length(x); % μετρούμενη ισχύς σήματος (για επαλήθευση)
A=(d/2)*[(-L+1):2:(L-1)];
figure(1);
hist(x,A);
```

Προκύπτει το παρακάτω ιστόγραμμα, στο οποίο και φαίνεται ότι τα στοιχεία του διανύσματος x ακολουθούν ομοιόμορφη κατανομή.




b) Χρησιμοποιούμε το εξής τροποποιημένο κατάλληλα κομμάτι της ask_errors:

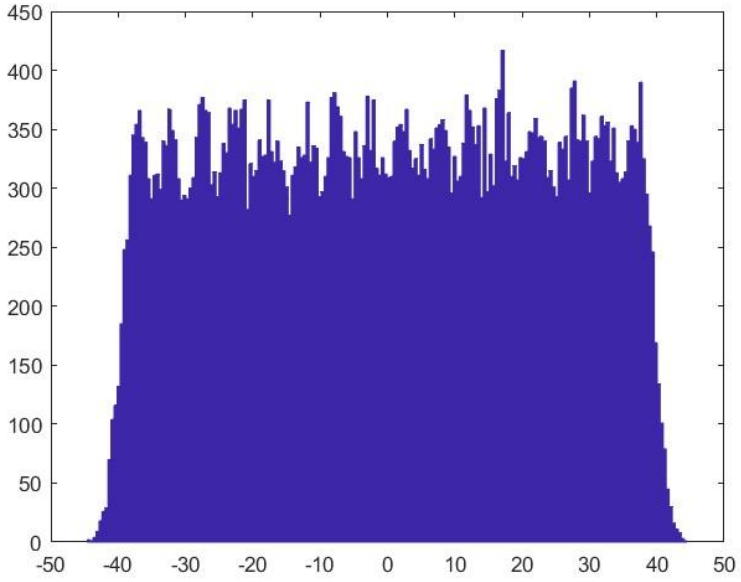
```
function errors=ask_errors(k,Nsymb,nsamp,EbNo)
L=2^k;
d=5;
SNR=EbNo-10*log10(nsamp/2/k); % SNR ανά δείγμα σήματος
x=(2*floor(L*rand(1,Nsymb))-L+1)*(d/2);
Px=(L^2-1)*(d^2)/4/3; % θεωρητική ισχύς σήματος
sum(x.^2)/length(x); % μετρούμενη ισχύς σήματος (για επαλήθευση)
y=rectpulse(x,nsamp);
n=wgn(1,length(y),10*log10(Px)-SNR);
ynoisy=y+n; % θορυβώδες σήμα
y=reshape(ynoisy,nsamp,length(ynoisy)/nsamp);
matched=ones(1,nsamp);
z=matched*y/nsamp;
figure(1);
hist(z,200);
end
```

Και καλούμε τη συνάρτηση 3 φορές με τις τιμές που δίνονται:

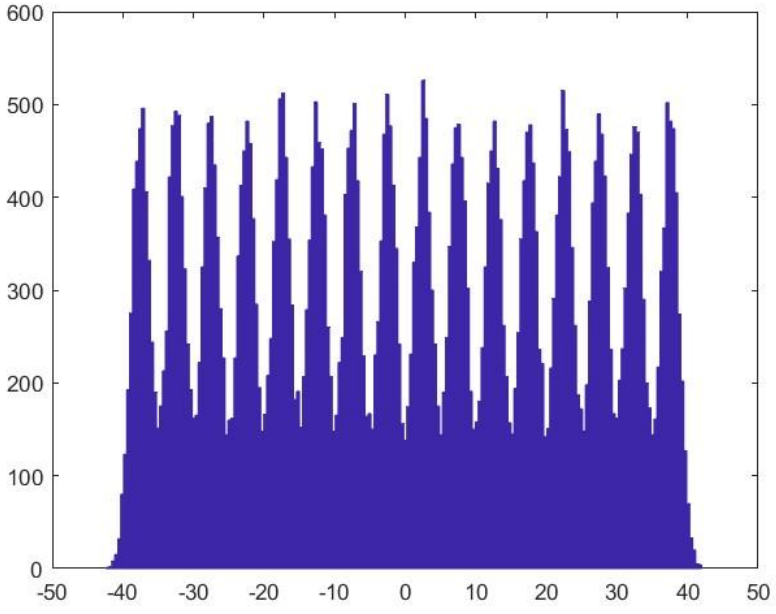
```
>> ask_errors(4,60000,20,12);
>> ask_errors(4,60000,20,16);
>> ask_errors(4,60000,20,20);
>>
```

Προκύπτουν τα εξής διαγράμματα:

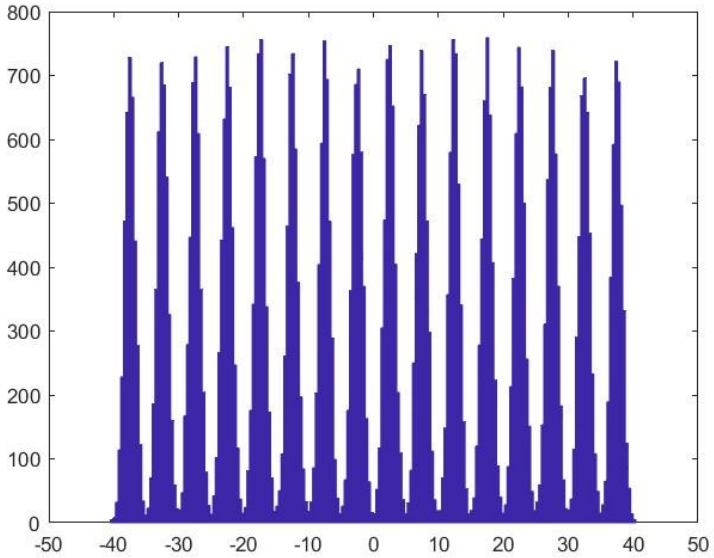
Για EbNo=12:




Για EbNo=16:



Για EbNo=20:




Παρατηρούμε ότι όσο αυξάνεται ο λόγος Eb/No σε dB, το ιστόγραμμα αραιώνει και δεν υπάρχουν επικαλύψεις. Αυτό συμβαίνει διότι υπάρχουν κενά ανάμεσα στα σύμβολα και οι τιμές του z συγκεντρώνονται πιο ευδιάκριτα γύρω από τις τιμές των πλατών. Αυτό διευκολύνει τη λήψη απόφασης για το ποια τιμή στάλθηκε και άρα μικρότερο σφάλμα. Αυξάνοντας το λόγο Eb/No αυξάνεται και η ισχύς του εκπεμπόμενου συμβόλου, δηλαδή ο SNR. Έτσι ο θόρυβος επηρεάζει λιγότερο την αρχική εκπομπή του κάθε συμβόλου και είναι ευκολότερο να ληφθεί απόφαση στον δέκτη για το σύμβολο που έχει σταλεί.

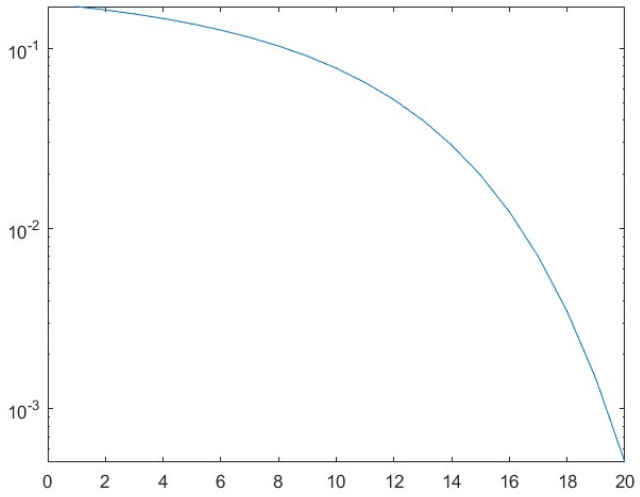
c) Η εντολή 20 είναι η εντολή reshape, η οποία τοποθετεί σε πίνακα με διαστάσεις [(nsamp) × (length(ynoisy)/nsamp)] το διάνυσμα του ynoisy. Δημιουργεί δηλαδή ένα πίνακα διαστάσεων (20,60000). Η εντολή 22 δίνει τη συσχέτιση του y με το προσαρμοσμένο φίλτρο (matched) φίλτρο. Η μεταβλητή matched είναι τύπου double και διαστάσεων (1,20). Η μεταβλητή x είναι τύπου double και διαστάσεων(1 40000). Η μεταβλητή y στην εντολή 17 είναι τύπου double και διαστάσεων(1 1200000). Η μεταβλητή y στην εντολή 20 είναι τύπου double και διαστάσεων(20 60000). Η μεταβλητή z είναι τύπου double και διαστάσεων(1 60000).
*έχουμε καλέσει την ask_errors(4,60000,20,12)

d) Στον βρόχο 24-27 ελέγχονται όλα τα σύμβολα στον πίνακα z που έχει λάβει ο δέκτης μέσω του βρόχου. Οι τιμές του z που παράγονται συγκρίνονται με μία από τις στάθμες. Για κάθε στοιχείο αφαιρούμε από τον πίνακα με τις τιμές των πλατών των συμβόλων και παίρνουμε την απόλυτη τιμή αυτών και αναζητούμε τη μικρότερη από αυτές προκειμένου να βρούμε τη θέση του συμβόλου που στάλθηκε. Το z παίρνει την πιο κοντινή τιμή και λαμβάνεται απόφαση για το ποια τιμή του x πραγματικά στάλθηκε.

## Μέρος 2ο

a) Χρησιμοποιώντας τον εξής κώδικα επαληθεύουμε την καμπύλη που δίνεται, καθώς προκύπτει η παρακάτω που είναι η ίδια:

```
k=mod(20019,2)+3;
L=2^k;
EbNo=linspace(1,20,20);
EbNo2=10.^(0.1*EbNo);
Pe=((L-1)/L)*erfc(sqrt((3*EbNo2*log2(L)/(L^2-1))));
BER=Pe/log2(L);
figure(1);
semilogy(EbNo,BER);
```



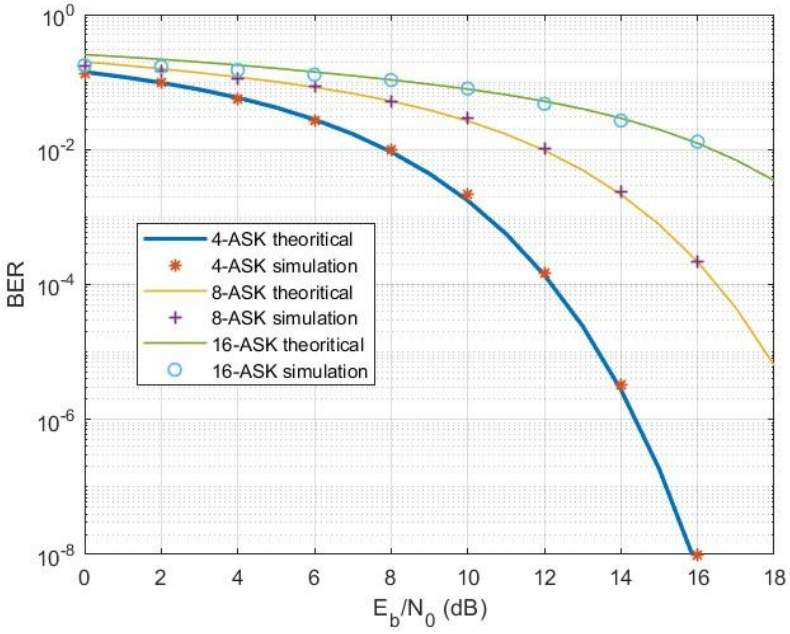

b) Για την χρήση του BERTOOL χρησιμοποιούμε την ask_errors:

```
function errors=ask_errors(k,Nsymb,nsamp,EbNo)
L=2^k;
SNR=EbNo-10*log10(nsamp/2/k); % SNR ανά δείγμα σήματος
x=(2*floor(L*rand(1,Nsymb))-L+1);
Px=(L^2-1)/3; % θεωρητική ισχύς σήματος
sum(x.^2)/length(x); % μετρούμενη ισχύς σήματος (για επαλήθευση)
y=rectpulse(x,nsamp);
n=wgn(1,length(y),10*log10(Px)-SNR);
ynoisy=y+n; % θορυβώδες σήμα
y=reshape(ynoisy,nsamp,length(ynoisy)/nsamp);
matched=ones(1,nsamp);
z=matched*y/nsamp;
A=[(-L+1):2:(L-1)];
for i=1:length(z)
[m,j]=min(abs(A-z(i)));
z(i)=A(j);
end
err=not(x==z);
errors=sum(err);
end
```

Και την ask_ber_func τροποποιημένη, στην οποία θέτουμε στο k τις τιμές {2,3,4} για να προκύψουν αντίστοιχα L={4,8,16} για τις αντίστοιχες 3 L-ASK:

```
function [ber,numBits] = ask_ber_func(EbNo, maxNumErrs, maxNumBits)
import com.mathworks.toolbox.comm.BERTool.*;
totErr = 0; % Number of errors observed
numBits = 0; % Number of bits processed
k=4;% number of bits per symbol
Nsymb=2000; % number of symbols in each run
nsamp=16; % oversampling,i.e. number of samples per T
while((totErr < maxNumErrs) && (numBits < maxNumBits))
 errors=ask_errors(k,Nsymb,nsamp,EbNo);
 totErr=totErr+errors;
 numBits=numBits + k*Nsymb;
end
ber = totErr/numBits;
end
```

Με τις κατάλληλες ρυθμίσεις του bertool προκύπτει το εξής γράφημα:

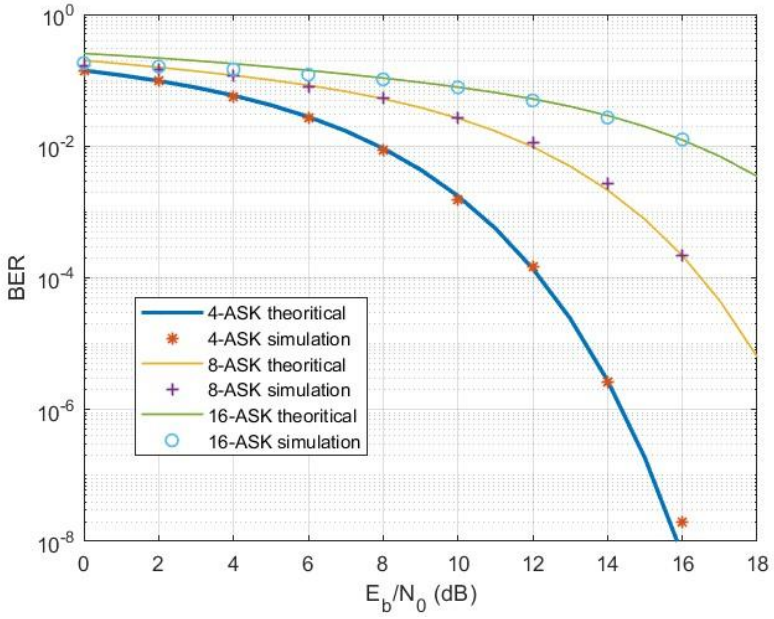



## Μέρος 3ο

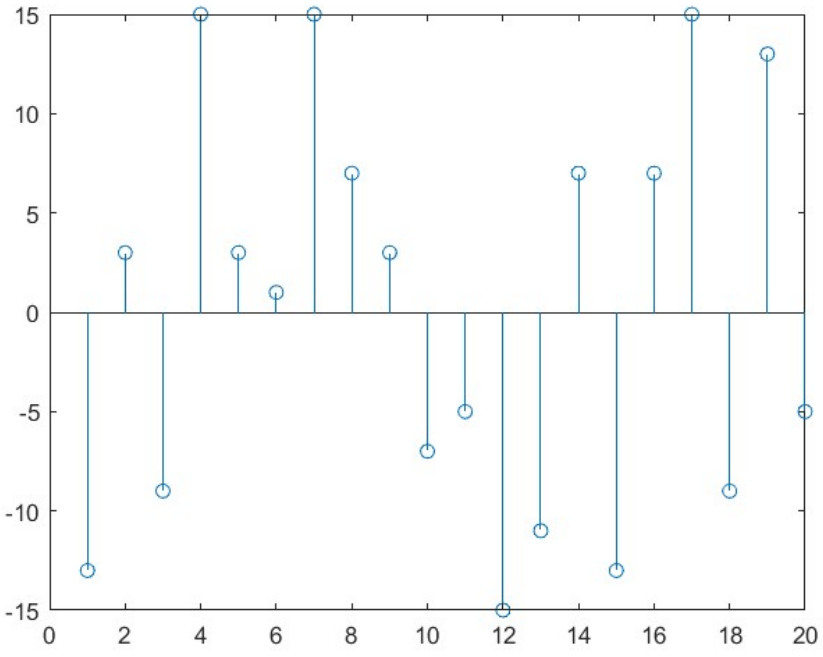
a) Ο τροποποιημένος κώδικας που χρησιμοποιούμε για την ask_errors είναι ο εξής:

```
function errors=ask_errors(k,Nsymb,nsamp,EbNo)
L=2^k;
SNR=EbNo-10*log10(nsamp/2/k); % SNR ανά δείγμα σήματος
x=2*floor(L*rand(1,Nsymb))-L+1;
Px=(L^2-1)/3; % θεωρητική ισχύς σήματος
sum(x.^2)/length(x); % μετρούμενη ισχύς σήματος (για επαλήθευση)
h=ones(1,nsamp);
h=h/sqrt(h*h'); % κρουστική απόκριση φίλτρου πομπού (ορθογωνικός παλμός μοναδιαίας ενέργειας)
y=upsample(x,nsamp); % μετατροπή στο πυκνό πλέγμα
y=conv(y,h); % το προς εκπομπή σήμα
y=y(1:Nsymb*nsamp); % περικόπτεται η ουρά που αφήνει η συνέλιξη
ynoisy=awgn(y,SNR,'measured');  % θορυβώδες σήμα
for i=1:nsamp matched(i)=h(end-i+1); end
yrx=conv(ynoisy,matched);
z = yrx(nsamp:nsamp:Nsymb*nsamp);
A=[-L+1:2:L-1];
for i=1:length(z)
[m,j]=min(abs(A-z(i)));
z(i)=A(j);
end
err=not(x==z);
errors=sum(err);
end
```

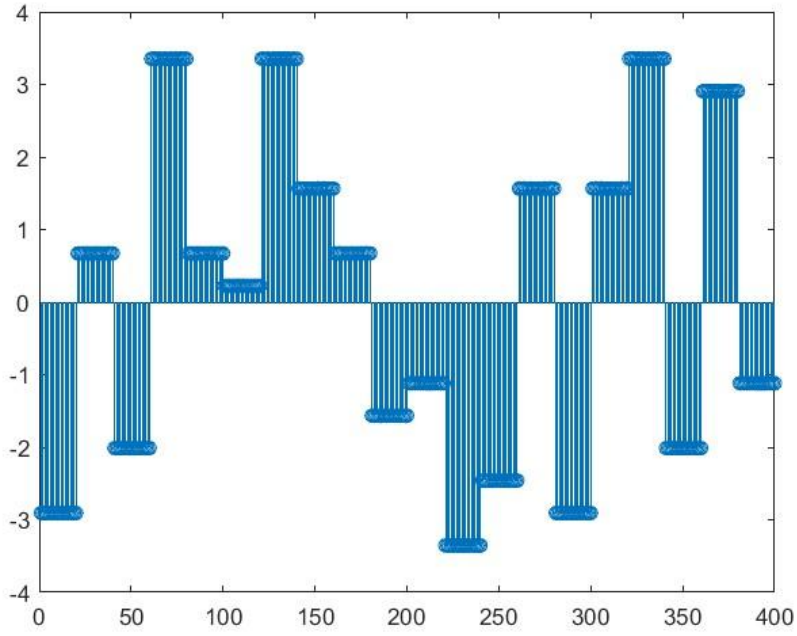
Η ask_ber_func παραμένει ίδια και όπως και πριν θέτουμε τις κατάλληλες τιμές στο k, και προκύπτει το εξής γράφημα, το οποίο είναι ίδιο με το προηγούμενο πριν την τροποποίηση της ask_errors.



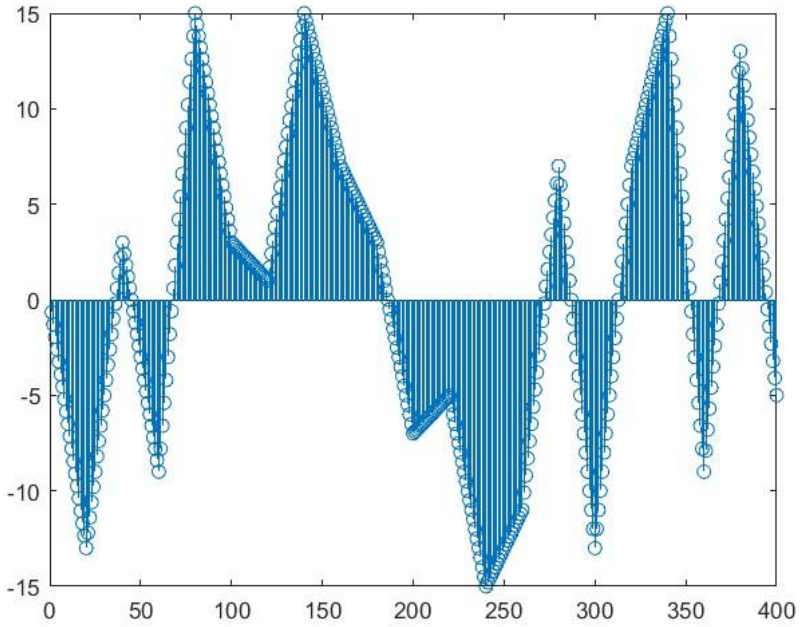

b) Η 1η εντολή figure; stem(x(1:20)); απεικονίζει με μίσχους το πλάτος των 20 πρώτων δειγμάτων του διανύσματος x. Οι τιμές αυτές είναι τυχαίοι ακέραιοι αριθμοί, καθώς έχουν στρογγυλοποιηθεί στον πλησιέστερο ακέραιο με τη συνάρτηση floor(). Προκύπτει το εξής διάγραμμα:



Η 2η εντολή figure; stem(y(1:20*nsamp)); απεικονίζει την εξέλιξη του y στον χρόνο. Το y είναι το προϊόν της συνέλιξης του x με τον παλμό μορφοποίησης h, ο οποίος είναι ένας τετραγωνικός παλμός nsamp δειγμάτων με διαμορφωμένο πλάτος ανάλογα με το μήκος του παλμού. Για το λόγο αυτό το διάγραμμα, λόγω του τετραγωνικού παλμού, παρουσιάζει σταθερό πλάτος για κάθε τιμή του x, όπως φαίνεται στο διάγραμμα:

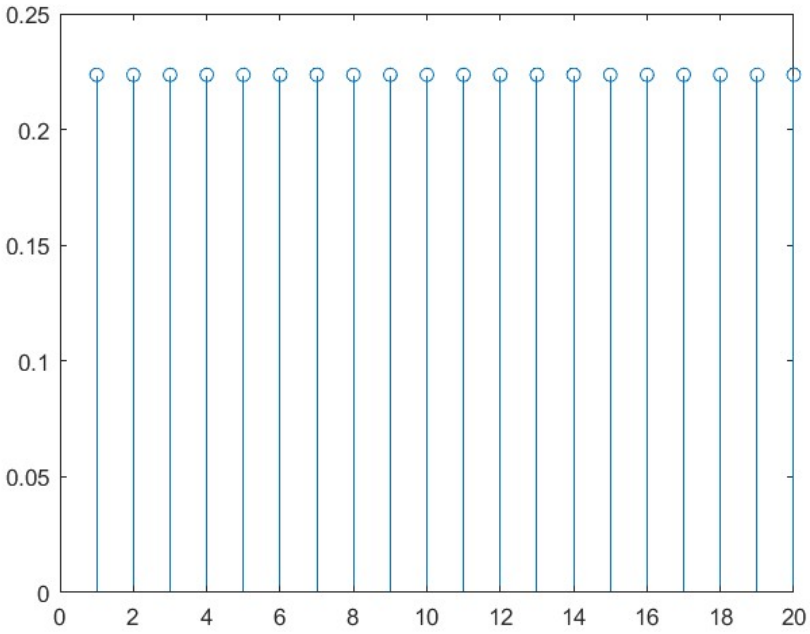



Η 3η εντολή figure; stem(yrx(1:20*nsamp)); απεικονίζει τα πρώτα 20*nsamp δείγματα του σήματος yrx στο διακριτό χρόνο, αφού έχει πραγματοποιηθεί συνέλιξη με το προσαρμοσμένο φίλτρο h(T-t), όπου T το μήκος του διανύσματος h. Το διάγραμμα που προκύπτει δείχνει την τιμή που θα έχει ο προς αποστολή παλμός κάθε χρονική στιγμή:

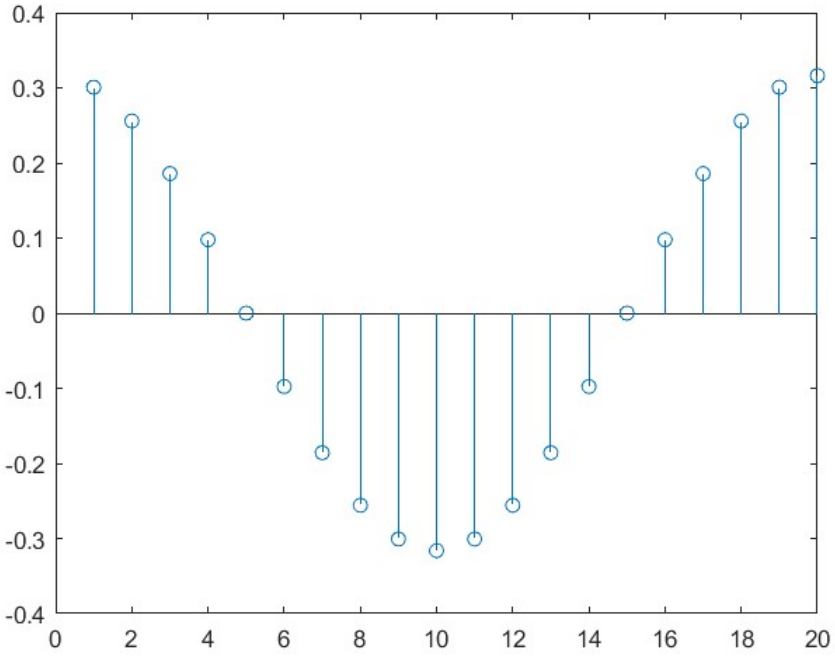


c) Εκτελούμε την εξομοίωση με bertool για nsamp=20 όπως και πριν με τον ορθογώνιο παλμό ώστε να γίνει η σύγκριση και σχεδιάζουμε με μίσχους τον παλμό h στις δύο περιπτώσεις (ορθογωνικό παλμό και ημιτονικό παλμό).

Ορθογωνικός παλμός h:



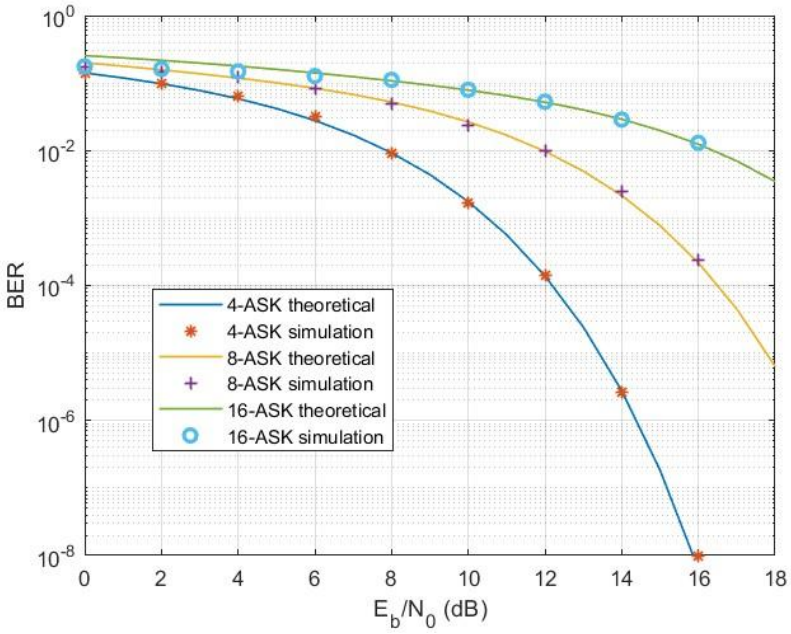

Ημιτονικός παλμός h:



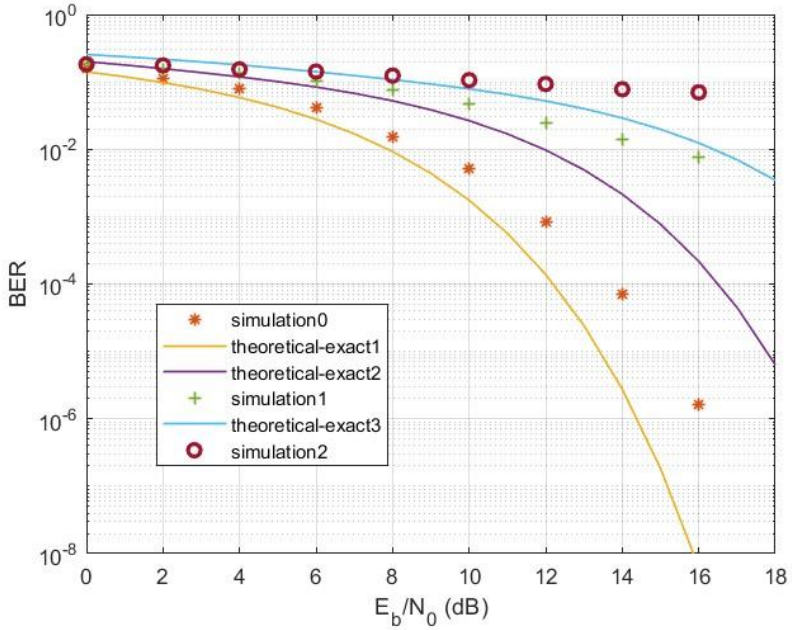
Για την εξομοίωση με το bertool χρησιμοποιείται ο εξής κώδικας για την ask_errors:

```
function errors=ask_errors(k,Nsymb,nsamp,EbNo)
L=2^k;
SNR=EbNo-10*log10(nsamp/2/k); % SNR ανά δείγμα σήματος
x=2*floor(L*rand(1,Nsymb))-L+1;
Px=(L^2-1)/3; % θεωρητική ισχύς σήματος
sum(x.^2)/length(x); % μετρούμενη ισχύς σήματος (για επαλήθευση)
h=cos(2*pi*(1:nsamp)/nsamp);
h=h/sqrt(h*h');
y=upsample(x,nsamp); % μετατροπή στο πυκνό πλέγμα
y=conv(y,h); % το προς εκπομπή σήμα
y=y(1:Nsymb*nsamp); % περικόπτεται η ουρά που αφήνει η συνέλιξη
ynoisy=awgn(y,SNR,'measured');
for i=1:nsamp matched(i)=h(end-i+1); end
yrx=conv(ynoisy,matched);
z = yrx(nsamp:nsamp:Nsymb*nsamp);
A=[-L+1:2:L-1];
for i=1:length(z)
[m,j]=min(abs(A-z(i)));
z(i)=A(j);
end
err=not(x==z);
errors=sum(err);
end
```

Η ask_ber_func παραμένει ίδια και όπως και πριν θέτουμε τις κατάλληλες τιμές στο k, και προκύπτει το εξής γράφημα, το οποίο είναι ίδιο με αυτό που προκύπτει με ορθογωνικό παλμό h:

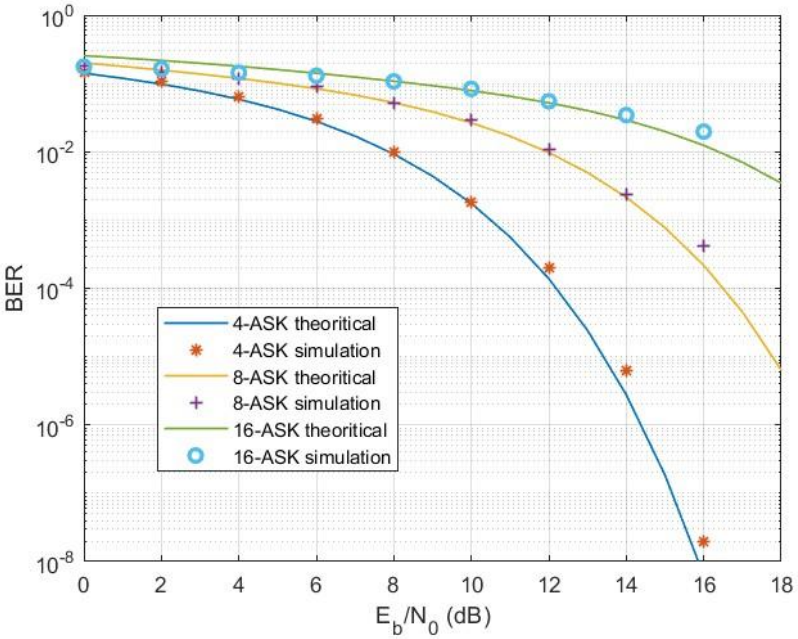


Επαναφέροντας την εντολή matched=h και για nsamp=16 προκύπτει το εξής λανθασμένο διάγραμμα:

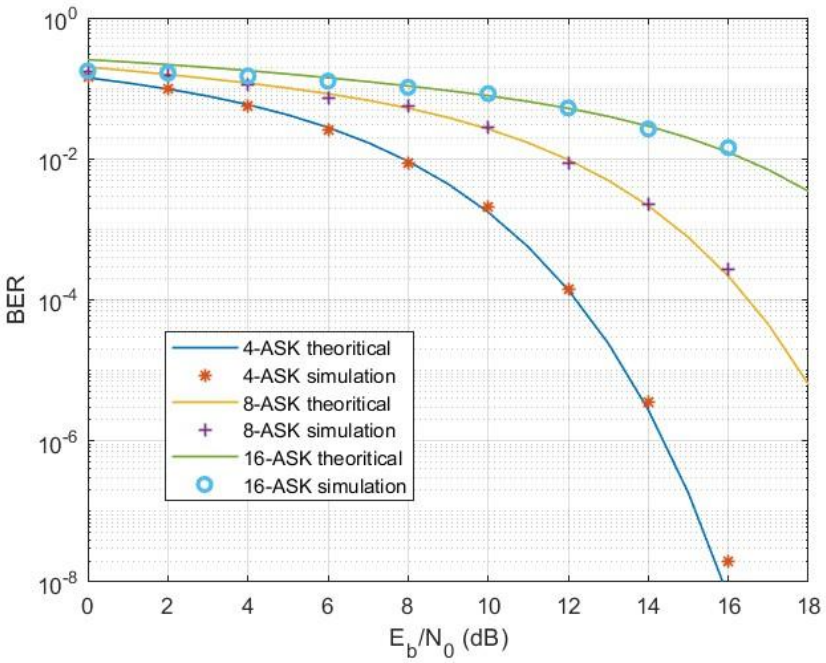



Τα αποτελέσματα δεν είναι σωστά διότι το προσαρμοσμένο φίλτρο στην γενική περίπτωση έχει ως κρουστική απόκριση h=kg(T-t), όπου g ο παλμός μορφοποίησης.

Για nsamp=32:



Για nsamp=64:




Παρατηρούμε ότι αυξάνοντας τον αριθμό των δειγμάτων για κάθε παλμό (nsamp=32,64) το πρόγραμμα προσομοίωσης ταυτίζεται με το θεωρητικό. Αυτό συμβαίνει, διότι όσα περισσότερα δείγματα παίρνουμε από τον συνημιτονικό παλμό μορφοποίησης τόσο πιο εύκολα ο δέκτης διακρίνει το μέγιστο πλάτος του και έτσι αποφασίζει για το πλάτος και το σύμβολο που έχει σταλεί. Αντίθετα στον τετραγωνικό παλμό παρατηρούμε ότι για όλες τις τιμές nsamp παίρνουμε το ίδιο αποτέλεσμα με το θεωρητικό γιατί έχει ίδιο πλάτος για κάθε χρονική στιγμή και για κάθε δείγμα. Αυτό φαίνεται από τα διαγράμματα του ορθογωνικού και του συνημιτονικού παλμού μέσω της εντολής stem (nsamp=20) που παρατέθηκαν παραπάνω.