

ΗΜΜΥ ΕΜΠ

6^ο Εξάμηνο

Ψηφιακές Επικοινωνίες I

Lab 3

Μέρος 1^ο: Διερεύνηση του κάδικα εξομοίωσης

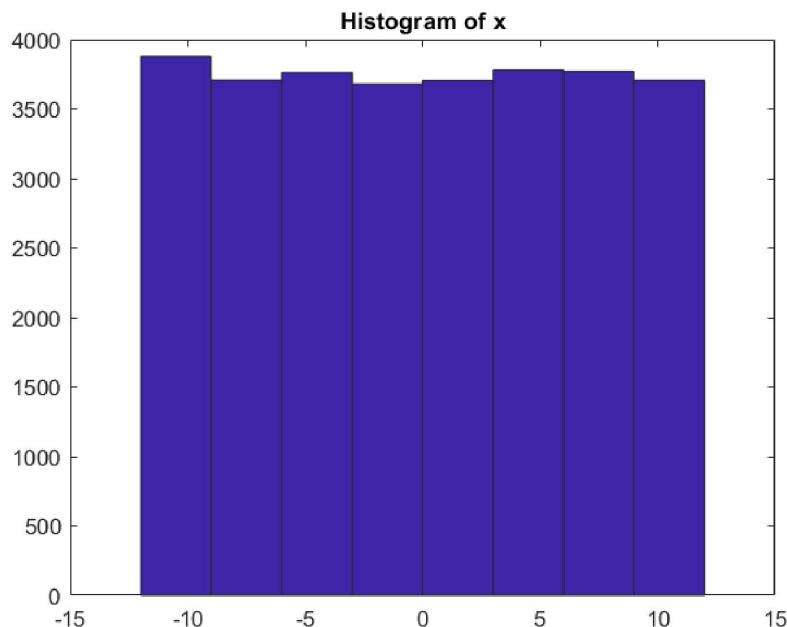
(α) Βάζουμε το d ως παράμετρο και πολλαπλασιάζουμε τα διανύσματα x και A με d/2. Τώρα στο διάνυσμα x προκύπτουν τυχαίες τιμές κατανεμημένες ομοιόμορφα στις νέες τιμές $\{\pm d/2, \pm d/2, \dots\}$ του διανύσματος A.

Η ισχύς είναι $\frac{E_1(L^2-1)}{3}$, όπου $E_1 = \left(\frac{d}{2}\right)^2$ η ενέργεια του πρώτου συμβόλου με πλάτος d/2. Γι' αυτό πολλαπλασιάζουμε το Px με $d^2/4$.

```
x=(2*floor(L*rand(1,M))-L+1)*d/2;  
Px=(L^2-1)/3*(d^2)/4;  
A=[-L+1:2:L-1]*d/2;
```

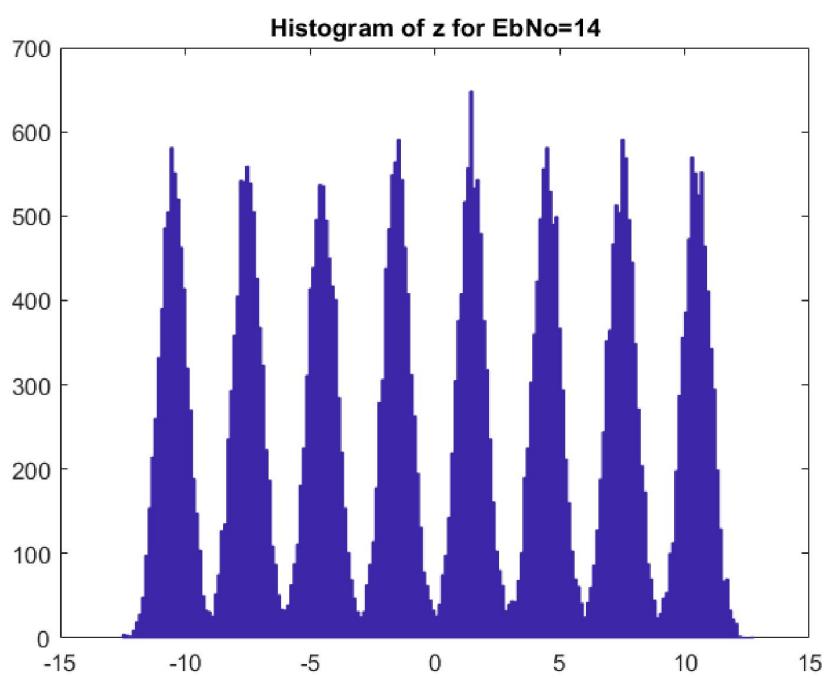
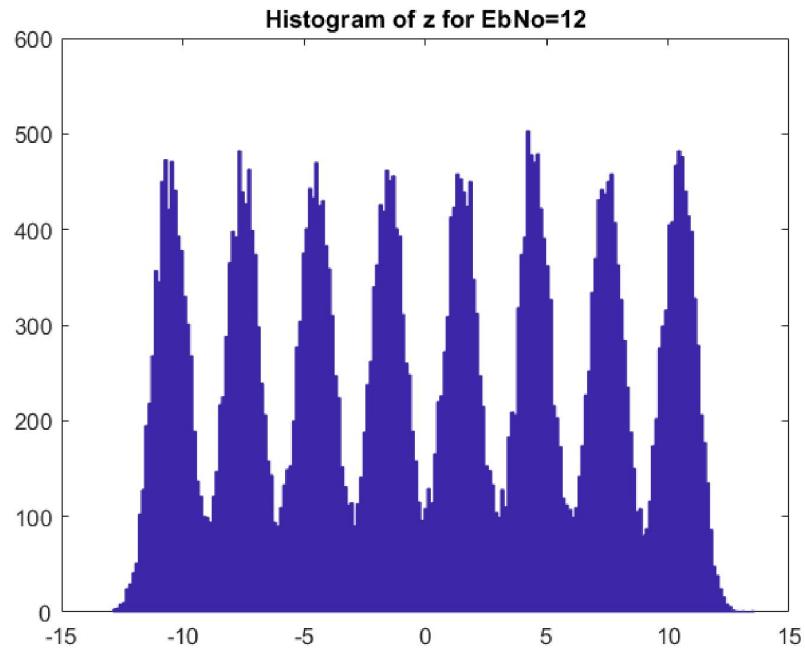
Το επαληθεύουμε παίρνοντας d=3, k=3=mod(,2)+3 και M=30000 δείγματα.

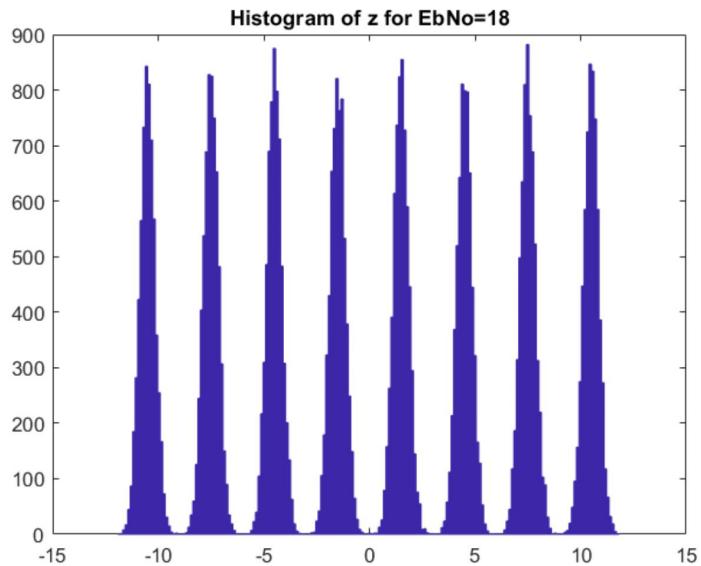
Τα δείγματα κατανέμονται ομοιόμορφα στα σημεία $\{\pm 1.5, \pm 4.5, \pm 7.5, \pm 10.5\}$:



Η θεωρητική ισχύς Px προκύπτει 47.25 και η πειραματική, από την εντολή $sum(x.^2)/length(x)$ προκύπτει 47.0826, αρκετά κοντά.

(β)





Οι τιμές του z , δηλαδή η έξοδος του προσαρμοσμένου φίλτρου, συγκεντρώνονται γύρω από τα σημεία $\{\pm 1.5, \pm 4.5, \pm 7.5, \pm 10.5\}$. Όσο αυξάνεται ο σηματοθορυβικός λόγος, τόσο μειώνεται η επίδραση του θορύβου. Αυτό φαίνεται στα παραπάνω ιστογράμματα, καθώς οι τιμές του z πλησιάζουν περισσότερο τα σημεία $\pm d/2$.

(γ) Η εντολή `y=rectpulse(x, nsamp)` δημιουργεί ορθογωνικό παλμό με πλάτη από το διάνυσμα x και $nsamp$ σημεία ανά σύμβολο. Έτσι μετά προσθέτουμε θόρυβο σε κάθε σημείο και προκύπτει θορυβώδες διάνυσμα `ynoisy` μήκους 1×800000 ($=50000$ σύμβολα επί 16 σημεία ανά σύμβολο). Δε μας νοιάζει όμως κάθε σημείο ξεχωριστά, αλλά κάθε σύμβολο, οπότε μετασχηματίζουμε το y σε 16×50000 `double` με την εντολή:

```
y=reshape(ynoisy,nsamp,length(ynoisy)/nsamp);
```

Στη συνέχεια δημιουργούμε προσαρμοσμένο φίλτρο `matched`, 1×16 `double`, με κρουστική απόκριση ορθογωνικό παλμό μήκους $nsamp$. Περνάμε το θορυβώδες σήμα από το προσαρμοσμένο φίλτρο πολλαπλασιάζοντας με την κρουστική απόκρισή του:

```
z=matched*y/nsamp;
```

Προκύπτει διάνυσμα z 1×50000 `double` ($1 \times 16 * 16 \times 50000 = 1 \times 50000$).

(δ) Το διάνυσμα z είναι η έξοδος από το προσαρμοσμένο φίλτρο, ενώ το διάνυσμα A έχει τις τιμές των συμβόλων. Στον βρόχο συγκρίνουμε τα στοιχεία του z με κάθε τιμή του A , και επιλέγουμε αυτή στην οποία είναι πιο κοντά. Με τον τρόπο αυτό προσδιορίζουμε σε ποιο σύμβολο αντιστοιχεί με μεγαλύτερη πιθανότητα η κάθε τιμή του z .

Κώδικας για το μέρος 1:

```
clear all; close all; clc;
k=mod(      ,2)+3; M=30000; d=3; nsamp=16; EbNo=[12 14 18];

L=2^k;
SNR=EbNo-10*log10(nsamp/2/k); % SNR ανά δείγμα σήματος
% Διάνυσμα τυχαίων ακεραίων {±1, ±3, ... ±(L-1)}*d/2
x=(2*floor(L*rand(1,M))-L+1)*d/2;
A=[-L+1:2:L-1]*d/2;
figure(1);
hist(x,A); title("Histogram of x");
disp('Theoretical power of x:');
Px=(L^2-1)/3*(d^2)/4 % Θεωρητική ισχύς σήματος
disp('Simulated power of x:');
sum(x.^2)/length(x) % μετρούμενη ισχύς σήματος (για επαλήθευση)
y=rectpulse(x,nsamp);
matched=ones(1,nsamp);

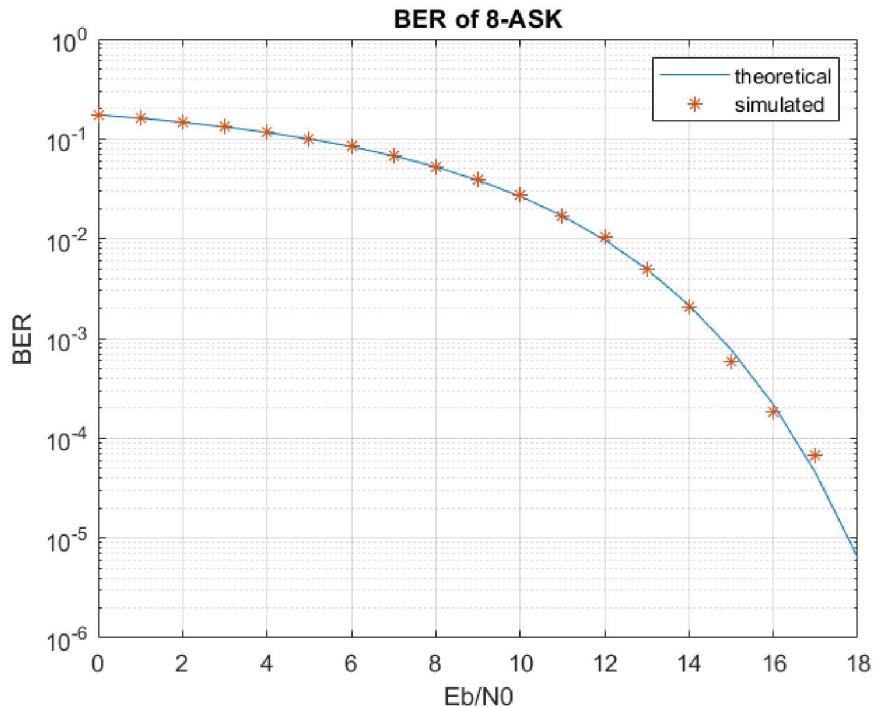
n12=wgn(1,length(y),10*log10(Px)-SNR(1));
ynoisy12=y+n12; % θορυβόδεσς σήμα με EbNo=12
y12=reshape(ynoisy12,nsamp,length(ynoisy12)/nsamp);
z12=matched*y12/nsamp;
figure(2);
hist(z12,200); title("Histogram of z for EbNo=12");

n14=wgn(1,length(y),10*log10(Px)-SNR(2));
ynoisy14=y+n14; % θορυβόδεσς σήμα με EbNo=14
y14=reshape(ynoisy14,nsamp,length(ynoisy14)/nsamp);
z14=matched*y14/nsamp;
figure(3);
hist(z14,200); title("Histogram of z for EbNo=14");

n18=wgn(1,length(y),10*log10(Px)-SNR(3));
ynoisy18=y+n18; % θορυβόδεσς σήμα με EbNo=18
y18=reshape(ynoisy18,nsamp,length(ynoisy18)/nsamp);
z18=matched*y18/nsamp;
figure(4);
hist(z18,200); title("Histogram of z for EbNo=18");
```

Μέρος 2^o: Καμπύλες επίδοσης (BER συναρτήσει του σηματοθορυβικού λόγου)

(α)



Παρατηρούμε ότι για μεγάλες τιμές του EbNo υπάρχει μια μικρή απόκλιση μεταξύ της θεωρητικής καμπύλης και των πειραματικών αποτελεσμάτων. Αυτό οφείλεται στο πεπερασμένο δείγμα. Για μεγάλο EbNo η ισχύς θορύβου είναι πολύ μικρή σχετικά με το σήμα οπότε τα λάθη είναι ελάχιστα, και χρειαζόμαστε πολύ μεγάλο δείγμα για να έχουμε πιο ακριβή αποτελέσματα.

Διαβάζοντας το διάγραμμα μιας εξομοίωσης παίρνουμε τις εξής τιμές:

| | | | | |
|-------|-----------|---------|------------|------------|
| Eb/No | 9 | 11 | 13 | 15 |
| BER | 0.0394167 | 0.01695 | 0.00501667 | 0.00058333 |

Κώδικας για το μέρος 2α:

```

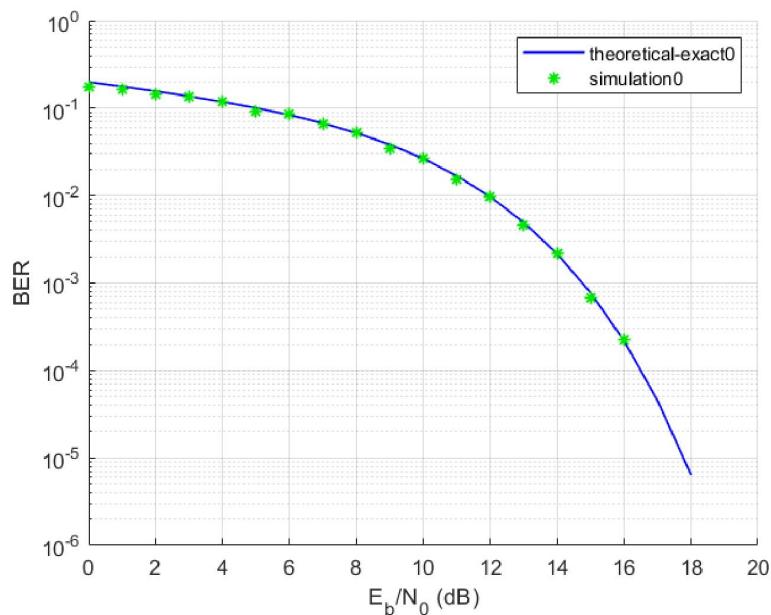
clear all; close all; clc;
k=mod(1{:},2)+3; nsamp=16; d=3; M=20000; L=2^k; %k=3

%theoretical BER
EbNo=0:18;
BERtheor=(L-1)/L*erfc(sqrt(3*k/(L^2-1).*10.^ (EbNo/10)))/k;
figure()
semilogy(EbNo,BERtheor);
grid on;
title("BER of 8-ASK");
xlabel("Eb/N0"); ylabel("BER");
hold on;

%simulation
BER=zeros(1,18);
for n=0:17
    BER(n+1)=ask_errors(k,M,nsamp,n,d)/M/k;
end
plot(0:17,BER,'*')
legend("theoretical","simulated")

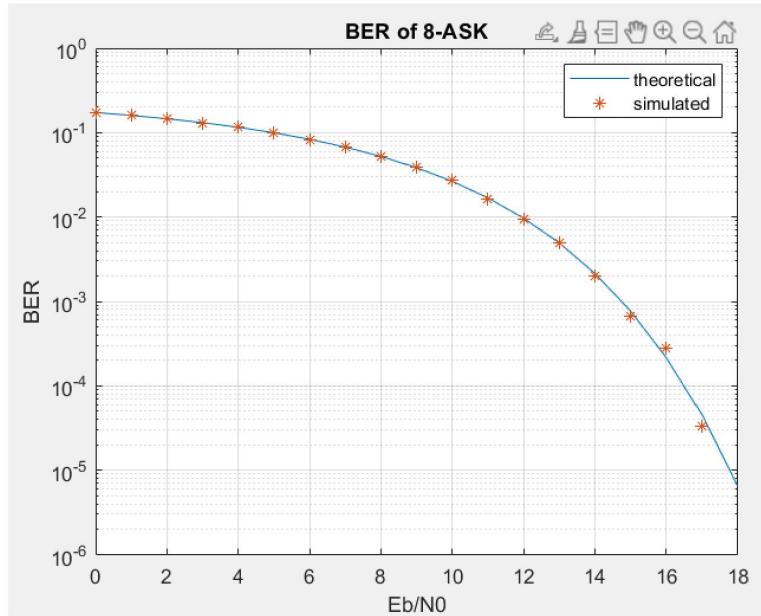
```

(β) Με χρήση του εργαλείου `bertool`, και του αρχείου `ask_ber_func_2.m` (το οποίο καλεί την τροποποιημένη από πριν συνάρτηση `ask_errors_2.m`) προκύπτει η ακόλουθη γραφική παράσταση:

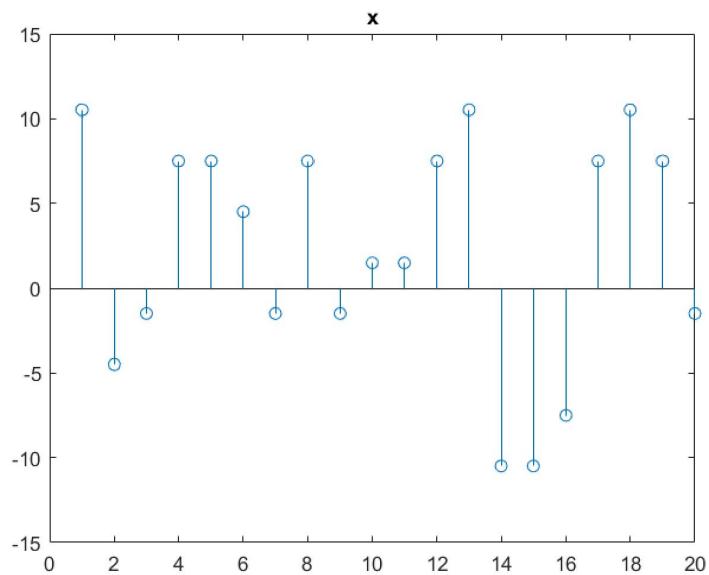


Μέρος 3: Υλοποίηση με συνέλιξη – Χρήση άλλων παλμών

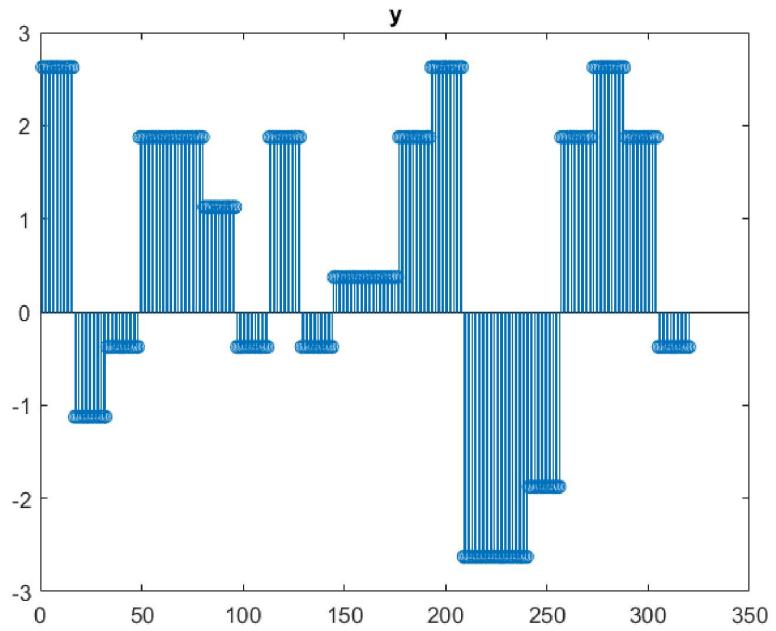
(α) Καλώ το αρχείο `lab3_3ab_` όπου περιέχεται κώδικας αντίστοιχος με το μέρος 2α, και προκύπτει παρόμοια γραφική παράσταση:



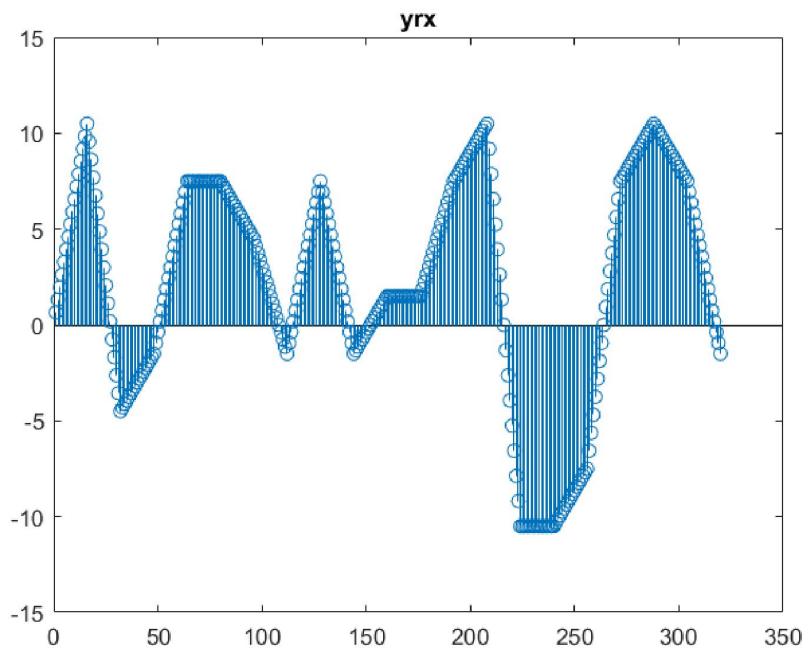
(β) Με $d=3$, $k=3$: `stem(x(1:20))`;



```
stem(y(1:20*nsamp));
```



```
stem(yrx(1:20*nsamp));
```



Με την εντολή `upsample` παραγεμίζουμε με `nsamp=1` μηδενικά ανάμεσα από τα δείγματα του x , μετά περνάμε το y από το φίλτρο πομπού (ορθογωνικός παλμός με μοναδιαία ενέργεια) με συνέλιξη και κόβουμε την ουρά. Η συνέλιξη των κρουστικών παλμών του `upsampled` για τον ορθογωνικό παλμό h δίνει το σχήμα που βλέπουμε. Παρατηρούμε ότι το πλάτος κάθε σημείου του y είναι το $\frac{1}{4}$ του πλάτους των αντίστοιχων σημείων του x . Αυτό ισχύει επειδή για $nsamp=16$, ο ορθογωνικός παλμός h έχει πλάτος 0.25.

Θεωρώντας μηδενικό θόρυβο, περνάμε το y από το προσαρμοσμένο φίλτρο (εδώ ίδιο με το h , ορθογωνικός παλμός). Η συνέλιξη ορθογωνίου με ορθογώνιο δίνει τριγωνικό παλμό, το οποίο βλέπουμε στο σχήμα. Παρατηρούμε ότι στο τέλος κάθε περιόδου η τιμή της y ισούται με το πλάτος του αντίστοιχου σημείου του x . Έτσι στη συνέχεια συγκρίνουμε το τελευταίο δείγμα κάθε περιόδου με τα πλάτη των συμβόλων για να αποφασίσουμε ποιο σύμβολο στάλθηκε.

Κώδικας για μέρος 3α, 3β:

```
clear all; close all; clc;
k=mod(      ,2)+3; M=30000; d=3; nsamp=16; EbNo=15; L=2^k;

%theoretical BER
EbNo=0:18;
BERtheor=(L-1)/L*erfc(sqrt(3*k/(L^2-1).*10.^^(EbNo/10)))/k;
figure(1)
semilogy(EbNo,BERtheor);
grid on;
title("BER of 8-ASK");
xlabel("Eb/N0"); ylabel("BER");
hold on;

%simulation
BER=zeros(1,18);
for n=0:17
    BER(n+1)=ask_errors2(k,M,nsamp,n,d)/M/k;
end
plot(0:17,BER,'*')
legend("theoretical","simulated")
hold off;

%3b
x=(2*floor(L*rand(1,M))-L+1)*d/2;
A=[-L+1:2:L-1]*d/2;
figure(2)
stem(x(1:20)); title("x");

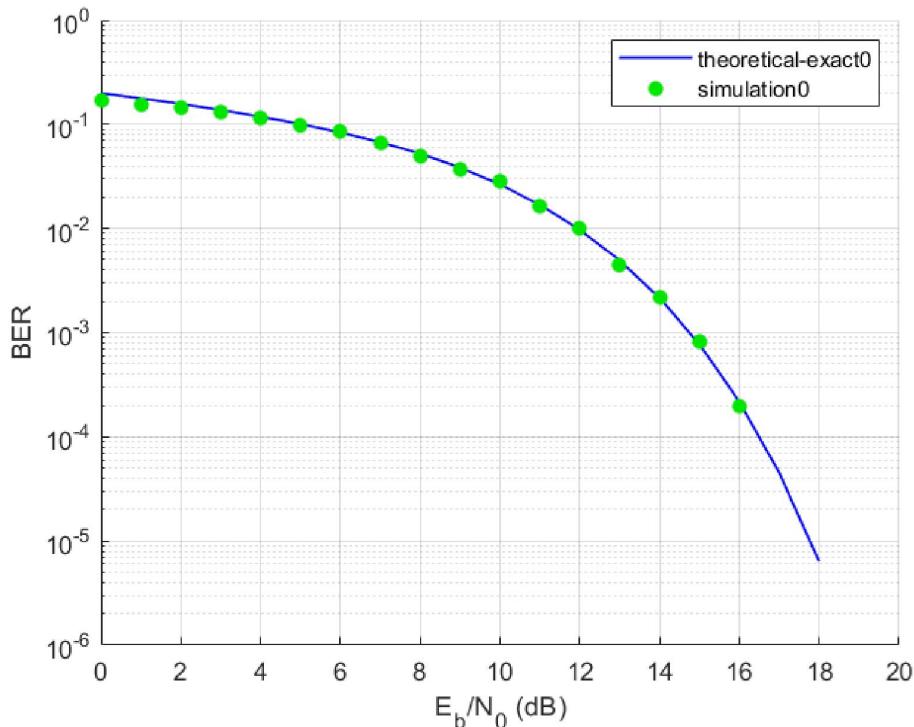
h=ones(1,nsamp);
h=h/sqrt(h*h');
y=upsample(x,nsamp);
y=conv(y,h);
```

```

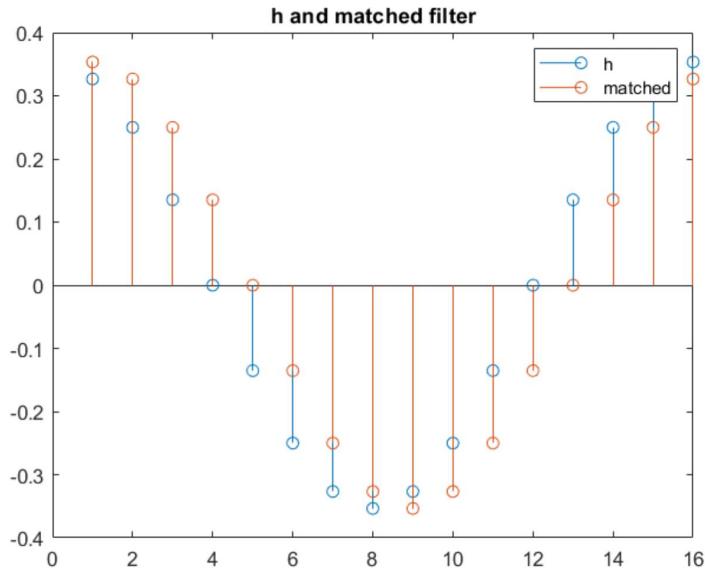
y=y(1:M*nsamp);
figure(3)
stem(y(1:20*nsamp)); title("y");
ynoisy=y; % θεωρώ μηδενικό θόρυβο
for i=1:nsamp matched(i)=h(end-i+1); end
yrx=conv(ynoisy,matched);
figure(4)
stem(yrx(1:20*nsamp)); title("yrx");

```

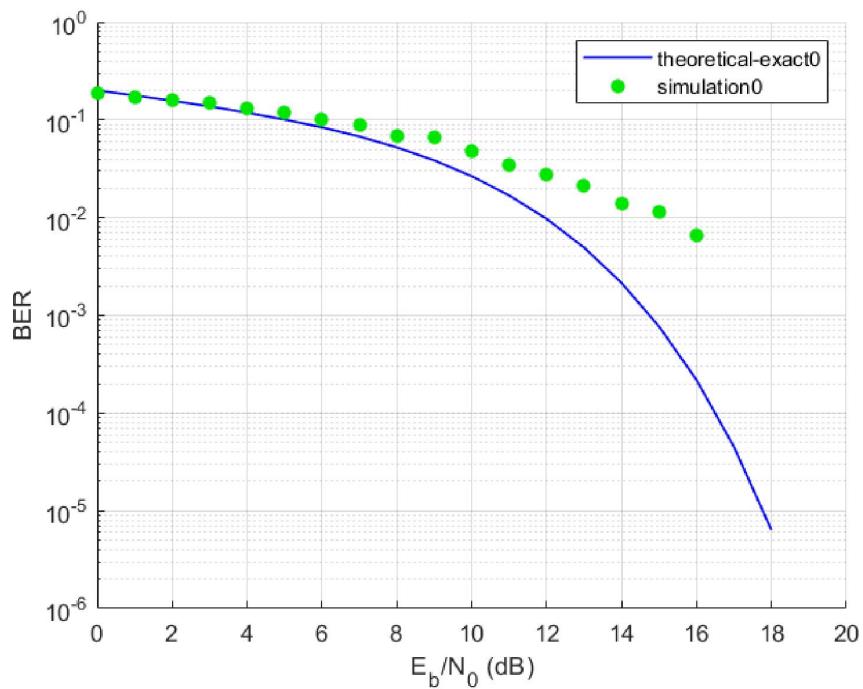
(γ) Με το εργαλείο `bertool` και το αρχείο `ask_ber_func_3_cosine_correct.m` (το οποίο καλεί την `ask_errors_3_cosine_correct.m`), προκύπτει το ακόλουθο αποτέλεσμα που φαίνεται μια χαρά:



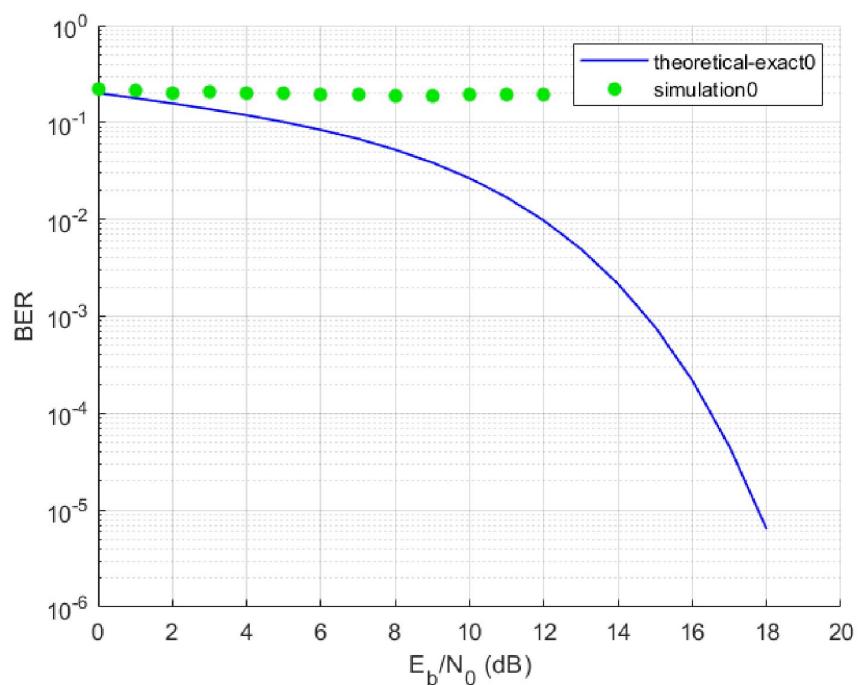
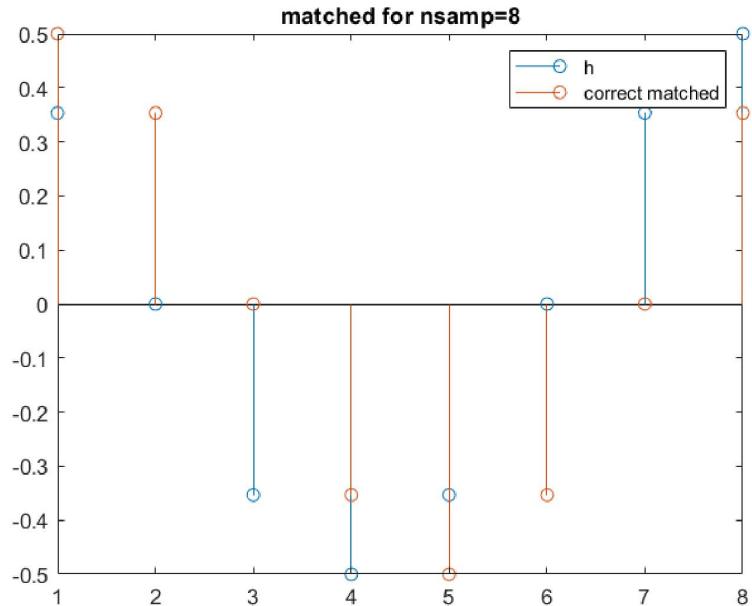
Το νέο προσαρμοσμένο φίλτρο προκύπτει από τον τύπο $matched = h[T - t]$, όχι $matched = h$. Εδώ φαίνεται η διαφορά για συνημιτονικό h :



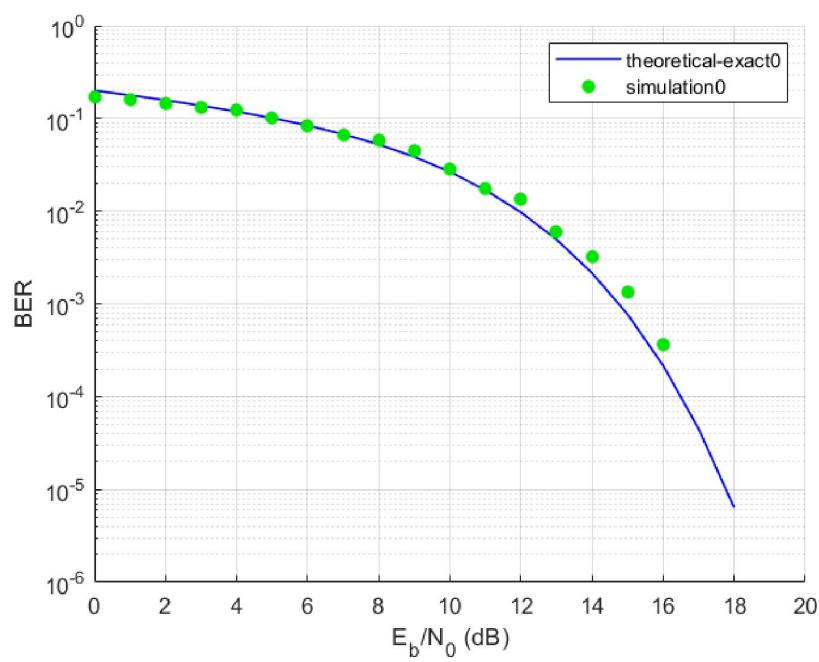
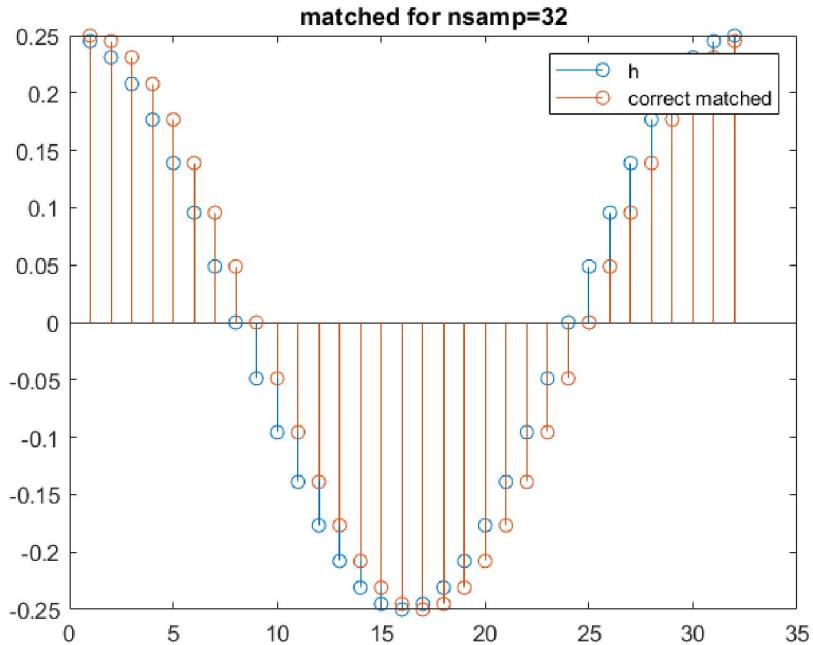
Αν το h είναι ορθογωνικός παλμός δεν υπάρχει διαφορά. Τώρα όμως που είναι συνημιτονοειδές αν θέσουμε $matched=h$; προκύπτουν πολύ περισσότερα λάθη:



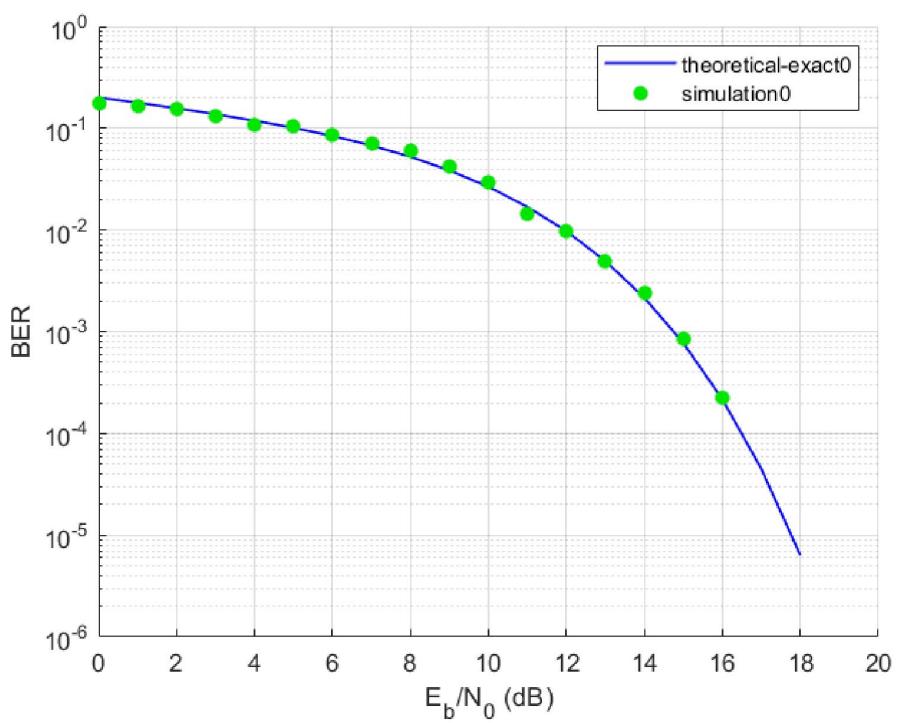
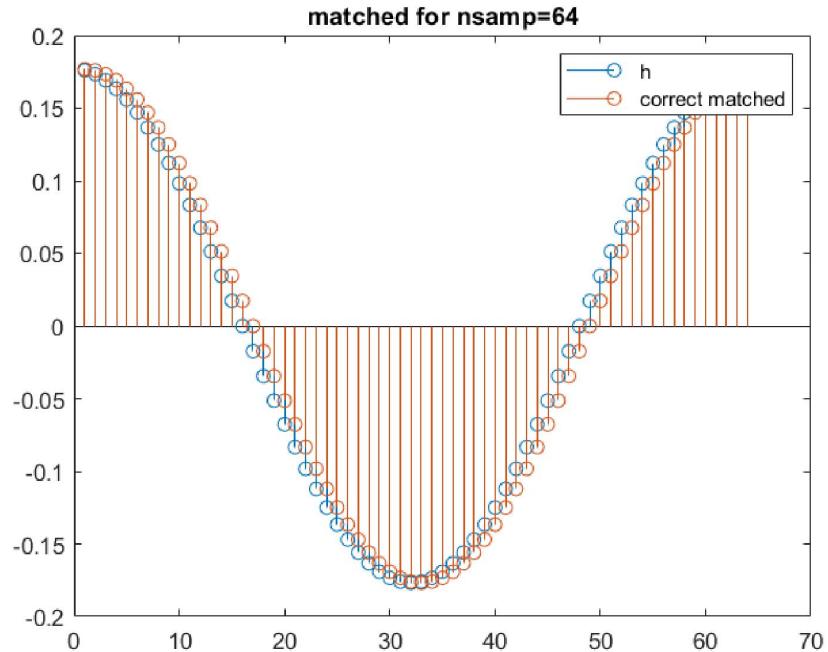
Για $nsamp=8$ η κατάσταση είναι ακόμα χειρότερη, επειδή η διαφορά ανάμεσα στο σωστό προσαρμοσμένο φίλτρο και το h είναι μεγαλύτερη:



Για $nsamp=32$ το σωστό προσαρμοσμένο φίλτρο μοιάζει αρκετά με την ή οπότε έχουμε λιγότερα λάθη από $nsamp=32$, αλλά εξακολουθούν να είναι περισσότερα από τα θεωρητικά:



Τέλος, για $nsamp=64$ η διαφορά είναι σχεδόν απαρατήρητη και το σύστημα δουλεύει σε ικανοποιητικό βαθμό:



Κώδικας για το μέρος 3γ:

Για το σωστό αποτέλεσμα απαιτείται το bertool με το *ask_ber_func_3_cosine_correct.m*, που καλεί την *ask_errors_3_cosine_correct.m*

Αν στο bertool βάλουμε το αρχείο *ask_ber_func_3_cosine_wrong.m* (που καλεί την *ask_errors_3_cosine_wrong.m*) παίρνουμε τη γραφική παράσταση με λάθη για *nsamp=16*

Οι υπόλοιπες γραφικές παραστάσεις του bertool με λάθη για διαφορετικά *nsamp* απαιτούν τροποποίηση του *ask_ber_func_3_cosine_wrong.m* (συγκεκριμένα, αλλαγή της τιμής του *nsamp*)

Οι γραφικές παραστάσεις για τη σύγκριση *h*, *matched* δεν προκύπτουν από τον κώδικα, τις έκανα για λόγους παρατήρησης εκτελώντας στο command line τις ακόλουθες εντολές, αλλάζοντας την τιμή του *nsamp*:

```
nsamp=16;  
h=cos(2*pi*(1:nsamp)/nsamp);  
for i=1:nsamp matched(i)=h(end-i+1); end  
hold on; stem(h); stem(matched); title("matched for nsamp=16");  
legend("h", "correct matched");
```