ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΨΗΦΙΑΚΕΣ ΕΠΙΚΟΙΝΩΝΙΕΣ Ι

Εργαστηριακή Άσκηση 3

Ονοματεπώνυμο: Μπέτζελος Χρήστος

Α.Μ. : 031 16 067

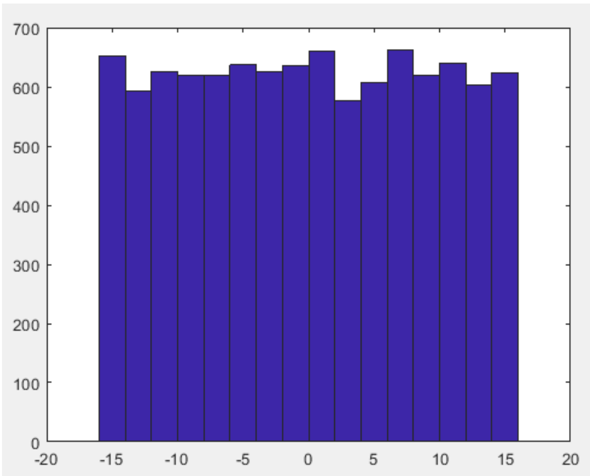
Εξάμηνο: 6^{o}

Μέρος 1^{o} . Διερεύνηση του κώδικα εξομοίωσης

a) Επαληθεύουμε, με υπολογισμό και προβολή σχετικού ιστογράμματος, ότι τα στοιχεία του διανύσματος x της εντολής 14 ακολουθούν πράγματι την ομοιόμορφη κατανομή στο σύνολο των ακεραίων {±1, ±3, ... ±(L-1)}. Χρησιμοποιήσαμε k=mod(nnnnn,2)+3, όπου nnnnn το τελευταίο 5-ψήφιο τμήμα του αριθμού μητρώου μας. (k=4). Παράγουμε (με την εντολή 14) τουλάχιστον 10000 τυχαίους ακέραιους, και χρησιμοποιούμε την εντολή hist(x,A) για τον υπολογισμό και την προβολή του ιστογράμματος, όπου A το διάνυσμα των L διαφορετικών τιμών αυτών των ακεραίων.

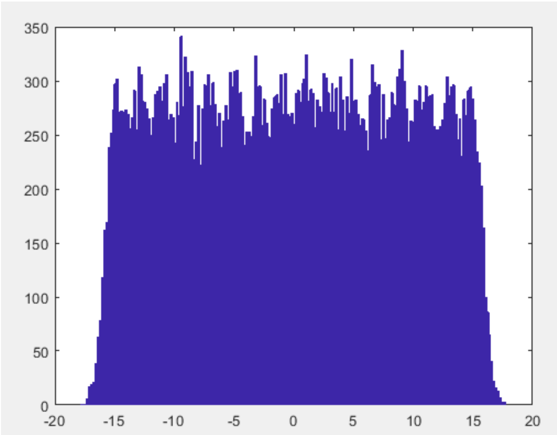
```
>> k=mod(16067,2)+3;
>> L=2^k;
>> M=10000;
>> x=2*floor (L*rand (1, M))-L+1;
>> A=-(L-1):2:(L-1);
>> hist(x,A)
```

Παρατηρούμε ότι οι ακέραιοι {±1, ±3, ... ±(L-1)} ακολουθούν όντως ομοιόμορφη κατανομή καθώς περίπου εμφανίζονται γύρω στις 625 φορές. (M/L=10000/16=625)

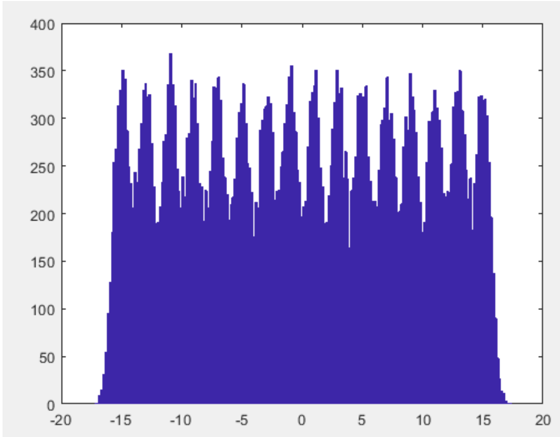


b) Με τιμές των παραμέτρων της συνάρτησης k=4, M=50000, nsamp=16 και EbNo=12 εκτελούμε τις εντολές 11 έως 22 και, στη συνέχεια, σχεδιάζουμε το ιστόγραμμα του z με την εντολή hist(z,200). Επαναλαμβάνουμε το παραπάνω για EbNo=14 και EbNo=18.

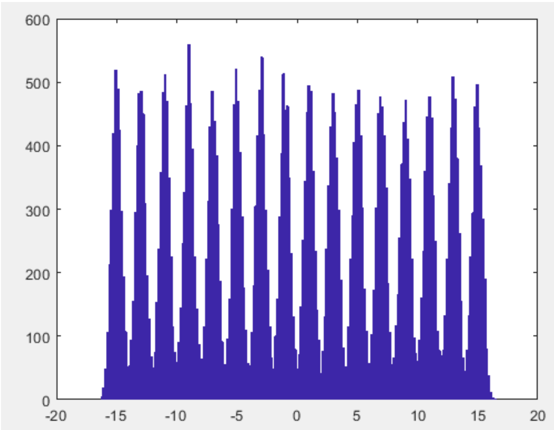
Eb/No=12



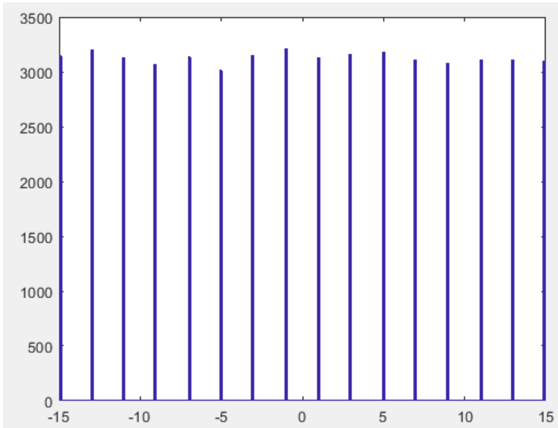
Eb/No=14



Eb/No=18



Eb/No→πολύ μεγάλο



Παρατηρούμε ότι όσο μεγαλώνει το EbNo τόσο περισσότερο εξασθενεί ο θόρυβος και οι τιμές τείνουν να πέσουν στις θεωρητικές (χωρίς θόρυβο). Τα δείγματα όταν έχουμε χαμηλές τιμές του EbNo κατανέμονται σε μεγάλο διάστημα στο (-16,16) και εμφανίζονται άρα λιγότερες φορές. Όσο μεγαλώνει το EbNo τα δείγματα τείνουν να 'πέσουν' πάνω στους ακεραίους {±1, ±3, ... ±(L-1)} (διακριτές τιμές) και να εμφανιστούν σχεδόν ομοιόμορφα (50000/16=3125).

c) Η εντολή 20 (reshape) κάνει τον πίνακα ynoisy που είναι (1x800000) σε πίνακα διαστάσεων (nsamp, length(ynoisy)/nsamp) δηλαδή σε (16x50000). Παρατηρούμε όπως είναι φυσικό ο αριθμός των στοιχείων των πινάκων να μην αλλάζει (16x50000=800000).

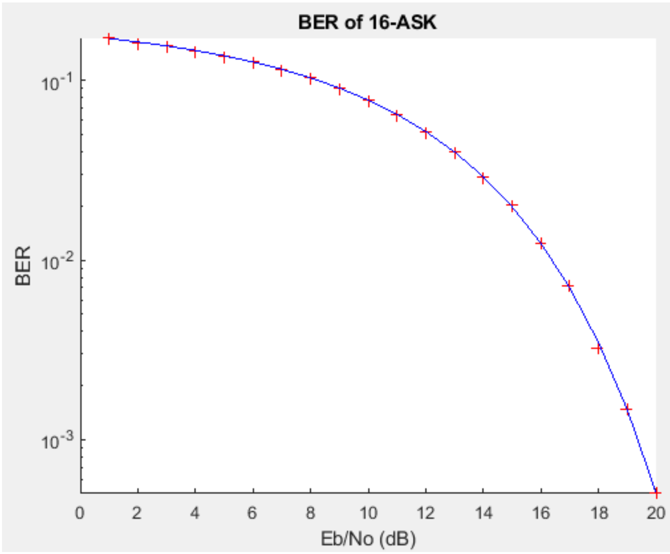
Η εντολή 22 κάνει εσωτερικό γινόμενο τον πίνακα matched με τον πίνακα y ο οποίος έχει πρώτα διαιρέσει κάθε στοιχείο του με nsamp (1x16) * (16x50000)=(1x50000). Δηλαδή, η μεταβλητή matched είναι τύπου double διαστάσεων 1x16, ενώ η μεταβλητή y είναι τύπου double διαστάσεων 16x50000.

d) Ο βρόχος 24-27 του Κώδικα 3.3: Για κάθε στοιχείο του z το ζεύγος [m,j] μας δίνει τη θέση j του κοντινότερου στοιχείου του l από το στοιχείο z, και το m την απόσταση από αυτό το σημείο. Έτσι αντικαθίσταται το στοιχείο του z με το αντίστοιχο “κοντινότερό” του από τον πίνακα l.

Μέρος 2ο. Καμπύλες επίδοσης (BER συναρτήσει του σηματοθορυβικού λόγου)

Θα επαληθεύσουμε την καμπύλη του σχήματος 3.10 των σημειώσεων για την L-ASK, με $L=2^k$, k=mod(nnnnn, 2)+3, όπου nnnnn το τελευταίο 5-ψήφιο τμήμα του αριθμού μητρώου μας. Θα σχεδιάσουμε τη θεωρητική καμπύλη και θα υπερθέσουμε τα αποτελέσματα της εξομοίωσης (διακριτά σημεία) a) με δικό μας κώδικα b) αλλά και με τη χρήση του BERTOOL του matlab. Έχουμε k=mod(16067,2)+3=4, άρα L=16-ASK.

a) *Με το δικό μας κύριο πρόγραμμα:* Σχεδιάζουμε την καμπύλη με χρήση της σχέσης (3.33) των σημειώσεων και την προσέγγιση BER≈Pe /log2L, και καλούμε κατάλληλα τη συνάρτηση ask_errors() για τον υπολογισμό των διακριτών σημείων. Ο κώδικας που χρησιμοποιήσαμε και το διάγραμμα φαίνονται παρακάτω.

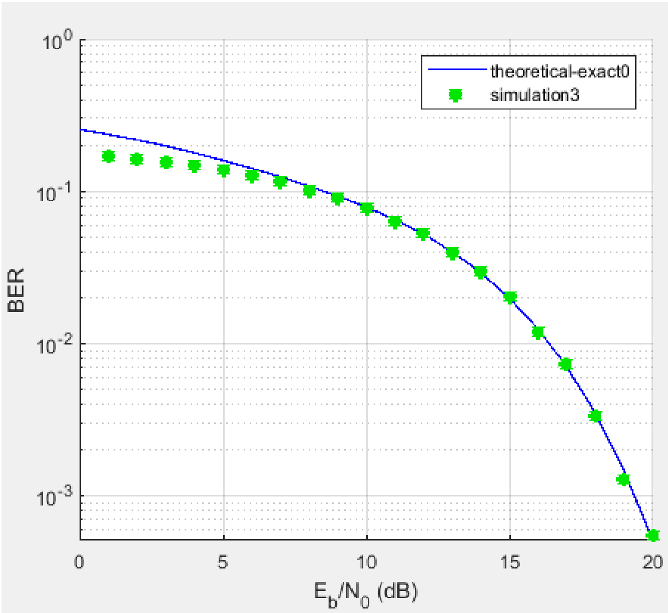


Η μπλε καμπύλη είναι για τη θεωρητική και τα κόκκινα σταυρουδάκια για τα διακριτά σημεία της εξομοίωσης.

<u>Κώδικας Matlab</u>

```
clear all;
close all;
clc;

k=mod(16067,2)+3;
L=2^k;
M=50000;
nsamp=16;
numBits=k*M;

EbNo=1:20;
Pe=((L-1)/L)*erfc(sqrt(3*k/(L^2-1)*(10.^(EbNo/10))));
theBER=Pe/k;
for i=1:20
    errors(i)=ask_errors(k,M,nsamp,i);
    simBER(i)=errors(i)/numBits;
end

figure(1);
hold on;
set(gca,'yscale','log');
semilogy(EbNo,simBER,'r+');
semilogy(EbNo,theBER,'b-');
title('BER of 16-ASK');
xlabel('Eb/No (dB)');
ylabel('BER');
hold off;
```

b) *Με τη χρήση του εργαλείου BERTOOL του MATLAB:* Το εργαλείο καλείται με πληκτρολόγηση της εντολής bertool στο παράθυρο εντολών του MATLAB. Στο παράθυρο που εμφανίζεται κάνουμε τις κατάλληλες, για κάθε περίπτωση, ρυθμίσεις. Το bertool καλεί τη συνάρτησή μας, *ask_errors()*, μέσω της συνάρτησης *ask_ber_func.m* του κώδικα 3.4, στην οποία έχουν τεθεί οι κατάλληλες τιμές για τις παραμέτρους k, Nsymb, και nsamp. Η θεωρητική καμπύλη και τα αποτελέσματα της εξομοίωσης (διακριτά σημεία) φαίνονται στο παρακάτω κοινό διάγραμμα. (Οι αποκλίσεις στην αρχή της προσομοίωσης υπάρχουν πιθανότατα λόγω της μικρής ισχύς του σήματος μας και του θορύβου).



Μέρος 3ο . Υλοποίηση με συνέλιξη - Χρήση άλλων παλμών

Αντικαταστούμε την εντολή 17 του Κώδικα 3.3 με τις εξής εντολές:

```
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:M*nsamp); % περικόπτεται η ουρά που αφήνει η συνέλιξη
```

Επίσης, το θορυβώδες σήμα, ynoisy, παράγεται με την εντολή:

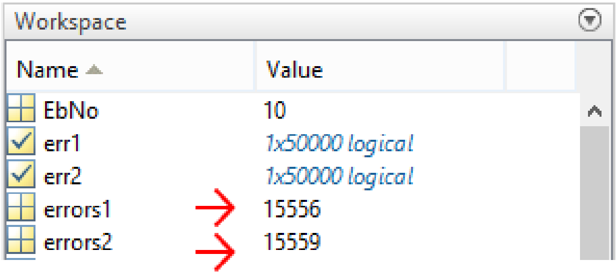
```
ynoisy=awgn(y,SNR,'measured');    % θορυβώδες σήμα
```

και το προσαρμοσμένο φίλτρο (εντολές 21,22) υλοποιείται με συνέλιξη ως:

```
for i=1:nsamp matched(i)=h(end-i+1); end
yrx=conv(ynoisy,matched);
z = yrx(nsamp:nsamp:M*nsamp); %Υποδειγμάτιση στο τέλος κάθε περιόδου T
```

Επίσης η εντολή που κάνει reshape το σήμα ynoisy εδώ καταργείται.

a) Προκειμένου να επιβεβαιώσουμε ότι ο τροποποιημένος κώδικας παράγει τα ίδια αποτελέσματα όπως ο Κώδικας 3.3 δημιουργούμε ένα script που έχουμε στην ουσία και τους δύο κώδικες μαζί. Δηλαδή, για ίδιο *k=4; M=50000; nsamp=16; EbNo=10;* αλλά και για ίδιο διάνυσμα *x* τυχαίων ακεραίων, με τους δύο τρόπους που δείξαμε βγάζουμε σαν αποτέλεσμα δύο μεταβλητές errors. Την errors1 και την errors2. Όπως φαίνεται και από το διπλανό απόκομμα, τα σφάλματα αυτά είναι σχεδόν ίσα, γεγονός που σημαίνει ότι ο τροποποιημένος κώδικας παράγει τα ίδια αποτελέσματα.



<u>Κώδικας Matlab</u>

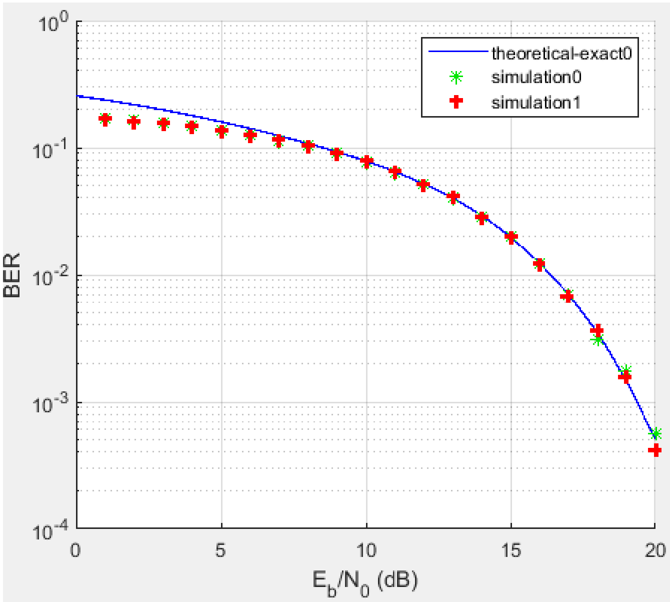
```
clc;
clear all;
close all;

k=4;
M=50000;
nsamp=16;
EbNo=10;
L=2^k;
SNR=EbNo-10*log10(nsamp/2/k); % SNR ανά δείγμα σήματος
% Διάνυσμα τυχαίων ακεραίων {±1, ±3, ... ±(L-1)}
x=2*floor(L*rand(1,M))-L+1;
Px=(L^2-1)/3; % θεωρητική ισχύς σήματος
y1=rectpulse(x,nsamp);
n=wgn(1,length(y1),10*log10(Px)-SNR);
y1noisy=y1+n; % θορυβώδες σήμα
y1=reshape(y1noisy,nsamp,length(y1noisy)/nsamp);
matched=ones(1,nsamp);
z1=matched*y1/nsamp;
l=[-L+1:2:L-1];
for i=1:length(z1)
  [m,j]=min(abs(l-z1(i)));
  z1(i)=l(j);
end
err1=not(x==z1);
errors1=sum(err1);

h=ones(1,nsamp); h=h/sqrt(h*h'); % κρουστική απόκριση φίλτρου
 % πομπού (ορθογωνικός παλμός μοναδιαίας ενέργειας)
y2=upsample(x,nsamp); % μετατροπή στο πυκνό πλέγμα
y2=conv(y2,h); % το προς εκπομπή σήμα
y2=y2(1:M*nsamp); % περικόπτεται η ουρά που αφήνει η συνέλιξη
y2noisy=awgn(y2,SNR,'measured'); % θορυβώδες σήμα
for i=1:nsamp matched(i)=h(end-i+1); end
yrx=conv(y2noisy,matched);
z2 = yrx(nsamp:nsamp:M*nsamp); % Υποδειγμάτιση στο τέλος κάθε περιόδου T
l=[-L+1:2:L-1];
for i=1:length(z2)
  [m,j]=min(abs(l-z2(i)));
  z2(i)=l(j);
end
err2=not(x==z2);
errors2=sum(err2);
```

Επίσης, αν τον νέο κώδικα τον αποθηκεύσουμε σε νέα συνάρτηση με όνομα *ask_errors_new* και τρέξουμε και τις δύο συναρτήσεις στο Command Window με τα ίδια ορίσματα, όπως φαίνεται και στο παρακάτω απόκομμα, θα μας δώσουν πολύ κοντινά αποτελέσματα.

Command Window

```
>> ask_errors(4,50000,16,10)

ans =

       15521

>> ask_errors_new(4,50000,16,10)

ans =

       15572
```

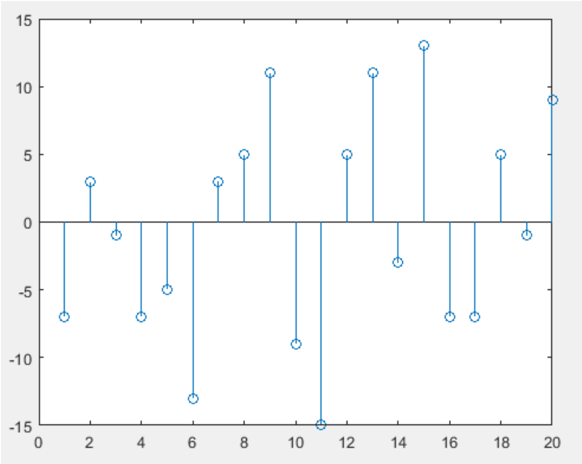
Τέλος, εκτός των δύο προηγούμενων τρόπων, για να επαληθεύσουμε ότι ο τροποποιημένος κώδικας παράγει τα ίδια αποτελέσματα μπορούμε να κάνουμε χρήση του bertool. Τα πράσινα σημάδια είναι με βάση την αρχική συνάρτηση *ask_errors()*, ενώ οι κόκκινοι σταυροί με την νέα τροποποιημένη *ask_errors_new()*. Όπως παρατηρούμε και από τη γραφική παράσταση, ο τροποποιημένος κώδικας παράγει σχεδόν τα ίδια αποτελέσματα με τον προηγούμενο.



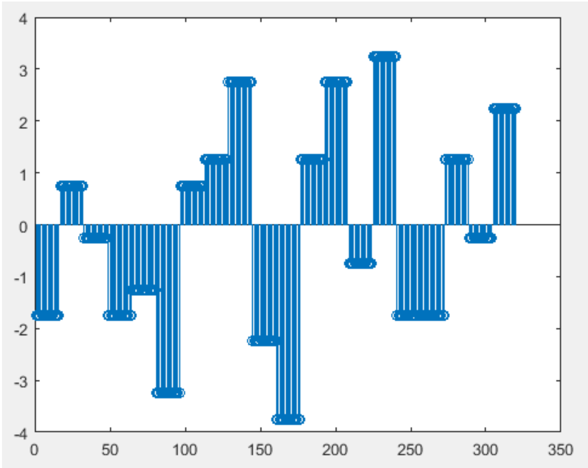
b) Εκτελούμε το σώμα της τροποποιημένης συνάρτησης *ask_errors()* (δηλαδή την *ask_errors_new()* χωρίς την προσθήκη θορύβου (δηλ. με ynoisy=y) και σχεδιάζουμε τμήμα των σημάτων x, y και yrx με τις παρακάτω τρεις εντολές.

```
figure; stem(x(1:20));
figure; stem(y(1:20*nsamp));
figure; stem(yrx(1:20*nsamp));
```

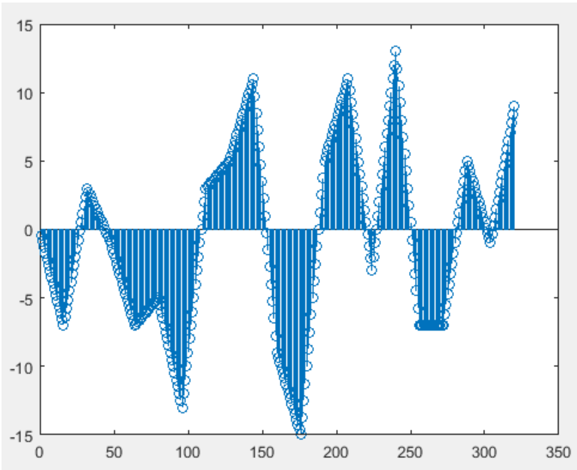
figure; stem(x(1:20));



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



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



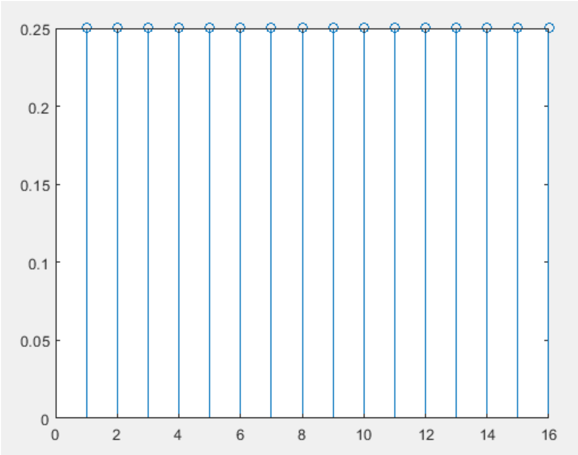
Το δεύτερο σχήμα είναι στην ουσία παλμική αναπαράσταση του πρώτου, δηλαδή προσπαθούμε να το αναπαραστήσουμε σαν παλμό. Αυτό γίνεται βάζοντας «δίπλα δίπλα» τα αρχικά μας σήματα nsamp φορές το καθένα. Στην συγκεκριμένη περίπτωση είναι 16. Γι' αυτό και παρατηρούμε στον οριζόντιο άξονα το 20 να έχει γίνει όπως αναμέναμε 320 (20x16=320).

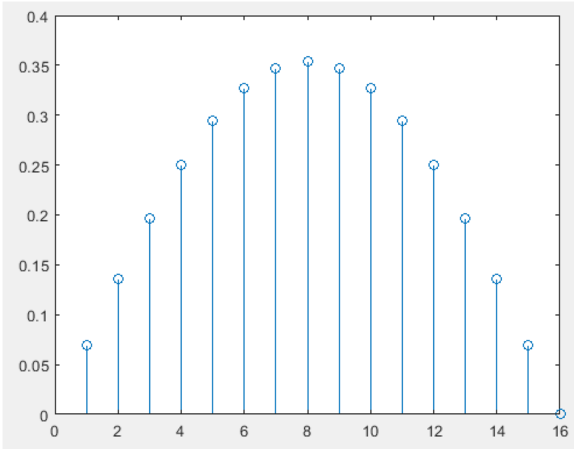
Το τρίτο σχήμα είναι η συνέλιξη του δεύτερου με τον ορθογωνικό παλμό h.

Από το τρίτο σχήμα στη συνέχεια παίρνουμε το τελευταίο από τα 16 δείγματα κάθε περιόδου T, με σκοπό να «πετύχουμε» τα δείγματα του πρώτου σχήματος. Στην περίπτωση μας, που δεν έχουμε θόρυβο (ynoisy=y), αυτό γίνεται όπως ήταν φυσικό με 100% επιτυχία δηλαδή δεν έχουμε κανένα σφάλμα.

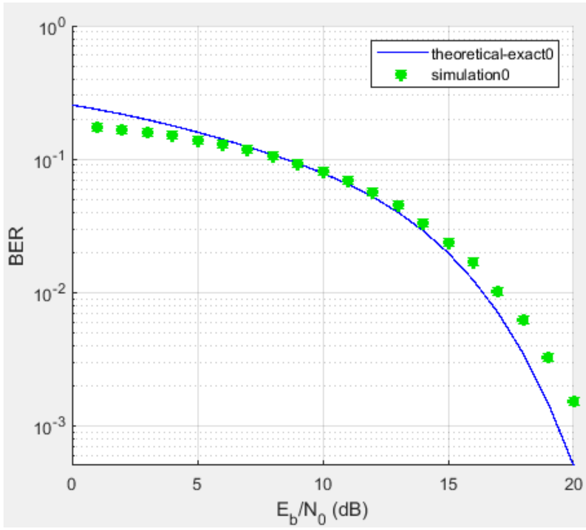
c) Αντικαταστούμε τον ορθογωνικό παλμό h με άλλον, ημιτονικό μισής περιόδου:
`h=sin(pi*(1:nsamp)/nsamp); h=h/sqrt(h*h');`

Παρατηρούμε τη μορφή του παλμού h στις δύο περιπτώσεις (ορθογωνικό παλμό και ημιτονικό παλμό), σχεδιάζοντάς τον με μίσχους (stem(h)). Τα σχήματα φαίνονται παρακάτω.





Εκτελούμε και πάλι την εξομοίωση με το bertool και παρατηρούμε ότι λόγω του ημιτονικού παλμού η νέα επίδοση του 16-ASK έχει μεγαλύτερες αποκλίσεις από τη θεωρητική καμπύλη, γεγονός που οφείλεται στο είδος του παλμού που γίνεται η συνέλιξη.



Επαναφέρουμε την εντολή matched=h για το προσαρμοσμένο φίλτρο και μικραίνουμε την τιμή του nsamp (π.χ. nsamp=8). Έτσι, παρατηρούμε ότι δεν παίρνουμε τα σωστά αποτελέσματα, γεγονός που οφείλεται στον μικρό αριθμό δειγμάτων των παλμών μας (nsamp), καθώς και στο ότι παραλείψαμε την ολίσθηση στον ημιτονικό παλμό h.