ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ (Η.Μ.ΜΥ)
ΤΗΛΕΠΙΚΟΙΝΩΝΙΑΚΑ ΣΥΣΤΗΜΑΤΑ 1 – ΤΗΛ301
ΑΝΑΦΟΡΑ ΕΡΓΑΣΤΗΡΙΑΚΗΣ ΑΣΚΗΣΗΣ 1

Φοιτητής:

Πινιάρας Δημήτριος

Σκοπός Άσκησης:

Μελέτη επικοινωνίας βασικής ζώνης υπό διαμόρφωση 2- PAM και αποκομμένους παλμούς Square Root Raised Cosine (SRRC).

Περί Υλοποίησης – Οργάνωση Εργασίας:

Η προσομοίωση των ζητηθέντων συνθηκών καθώς και η υλοποίηση των, κατά την εκφώνηση, ζητουμένων αποτελεί παραλλαγή της υλοποίησης που εστάλη και βαθμολογήθηκε το ακαδημαϊκό έτος 2019-20, ενώ πραγματοποιήθηκε στο κέλυφος Matlab (εκδ. R2018b) σε Linux.

Αναφορικά με την κατάτμιση της παρούσης, οργανώνεται στα λογικά στάδια Α, Β και Γ (έκαστο για κάθε ομάδα ερωτημάτων), ενώ προστέθηκε και η ενότητα “Εισαγωγικά” για τα προστιθέντα (σε σχέση με το 2019) ερωτήματα.

Ο κώδικας τόσο για το κυρίως μέρος, όσο και για τις συναρτήσεις αλλά και σχολιασμός, όπου αυτό έχει έννοια, βρίσκονται στα αντίστοιχα παραρτήματα.

Υλοποίηση Εργασίας:

Εισαγωγικό Μέρος:

Θ1.

$$Rxx(t_1, t_2) = E[X_1(t)\, X_2(t)] = \iiint_{-oo}^{+oo} \frac{1}{\sqrt{T}} \frac{1}{\sqrt{T}} x_1 x_2 dx_1 dx_2 = \iiint_{-T/2}^{+T/2} \frac{1}{T} x_1 x_2 dx_1 dx_2 = \int_{-T/2}^{+T/2} x_1 [\frac{(T/2)^2 - (-T/2)^2}{2}] dx_1 = 0$$

επίσης E[X(t)] = 0, άρα η Φ(τ) στάσιμη WSS.

Θ2.

$$Rxx(t_1 - 10, t_2 - 10) = E[X_1(t-10)\, X_2(t-10)] = \iiint_{-oo}^{+oo} \frac{1}{\sqrt{T}} \frac{1}{\sqrt{T}} x_1 x_2 dx_1 dx_2 = \iiint_{-T/2+10}^{+T/2+10} \frac{1}{T} x_1 x_2 dx_1 dx_2$$

$$Rxx(t_1 - 10, t_2 - 10) = \int_{-T/2+10}^{+T/2+10} x_1 [\frac{(T/2+10)^2 - (-T/2+10)^2}{2}] dx_1 = 5 \times 20\, T = 100\, T$$

Μέρος Α:

Υλοποίηση παλμών SRRC μέσω δοθείσας συνάρτησης και σχεδιασμό αυτών σε κοινό γράφημα. Οι αρχικές παράμετροι ακολουθούν:

T = 10^{-3} sec.
A = 4.
a = [0 0.5 1] Τιμές roll – off factor.

$$Ts = \frac{T}{over}$$

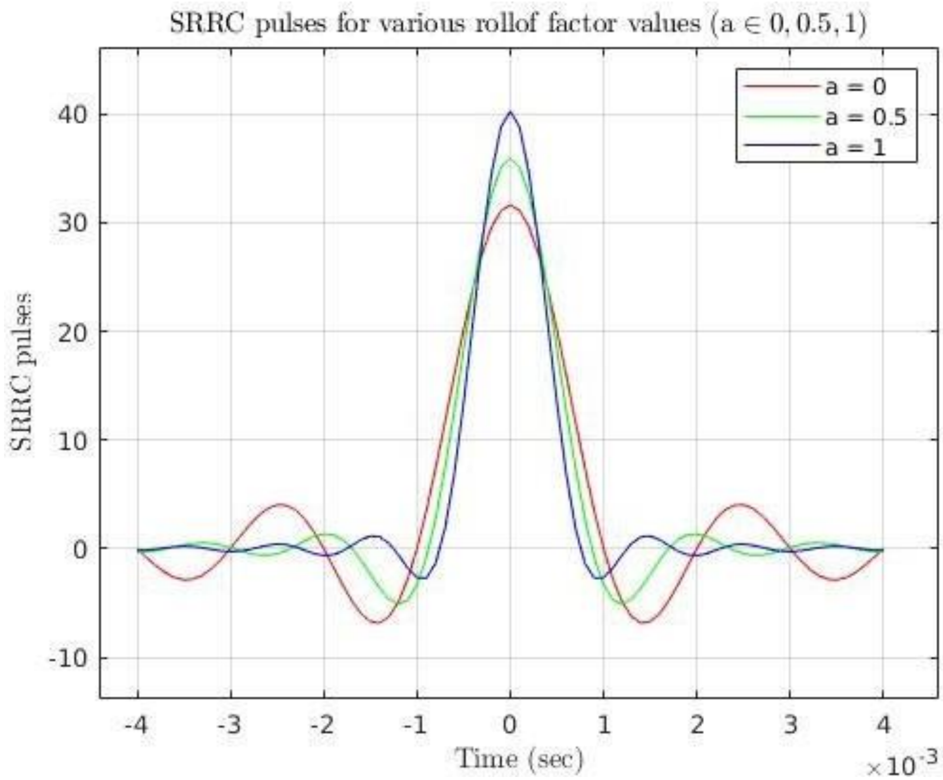
Εν γένει παλμός SRRC ορίζεται ώς:

$$s(t) = \frac{1}{\sqrt{T}} sinc(\frac{\tau}{T}), a = 0$$

$$s(t) = \frac{4a}{\sqrt{\pi}T} \frac{\cos[\ \frac{(1+a)\pi t}{T}] \ - \frac{T}{4at} sin[\ \frac{(1-a)\pi t}{T}]}{1-(\frac{4at}{T})^2} a \in (0,1)$$

Παρακάτω ακολουθούν απεικονίσεις και συγκρίσεις παλμών SRRC για διαφορετικούς συντελεστές αποκοπής (α – rolloff factor) καθώς και ολισθήσεις αυτών για διαφορετικούς συντελεστές ολίσθησης (k).

A1.



Σχολιάζοντας τα παρακάτω μπορούν να εξαχθούν τα εξής συμπεράσματα:

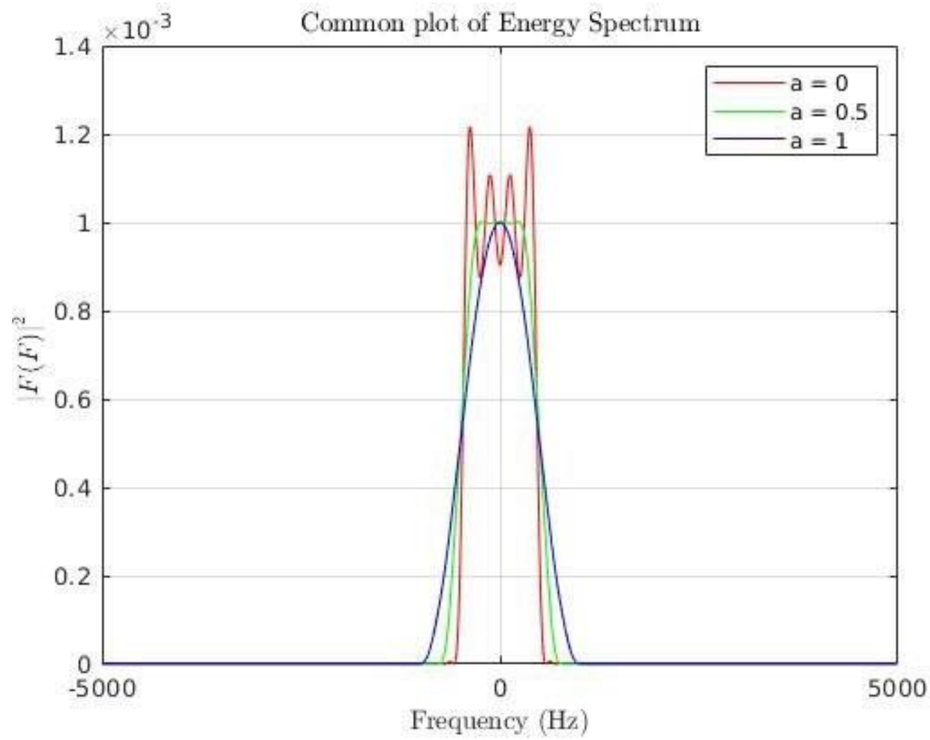
- Αύξηση του συντελεστή α (roll-of factor), θα αυξήσει το ρυθμό με τον οποίο το πλάτος του παλμού μειώνεται.Εντονότερη απόσβεση δηλαδή όσο αυξάνεται ο χρόνος.
- Οι παλμοί έχουν κοινή διάρκεια (2A).
- Μείωση περιόδου ταλάντωσης για αύξηση του α.
- Αύξηση στο μέγιστο του εκάστοτε παλμού για αύξηση του α.

Σημαντικές διαφορές μπορούν να ειδωθούν και στο πεδίο της συχνότητας κατά τη μεταβολή του roll-of factor (α).

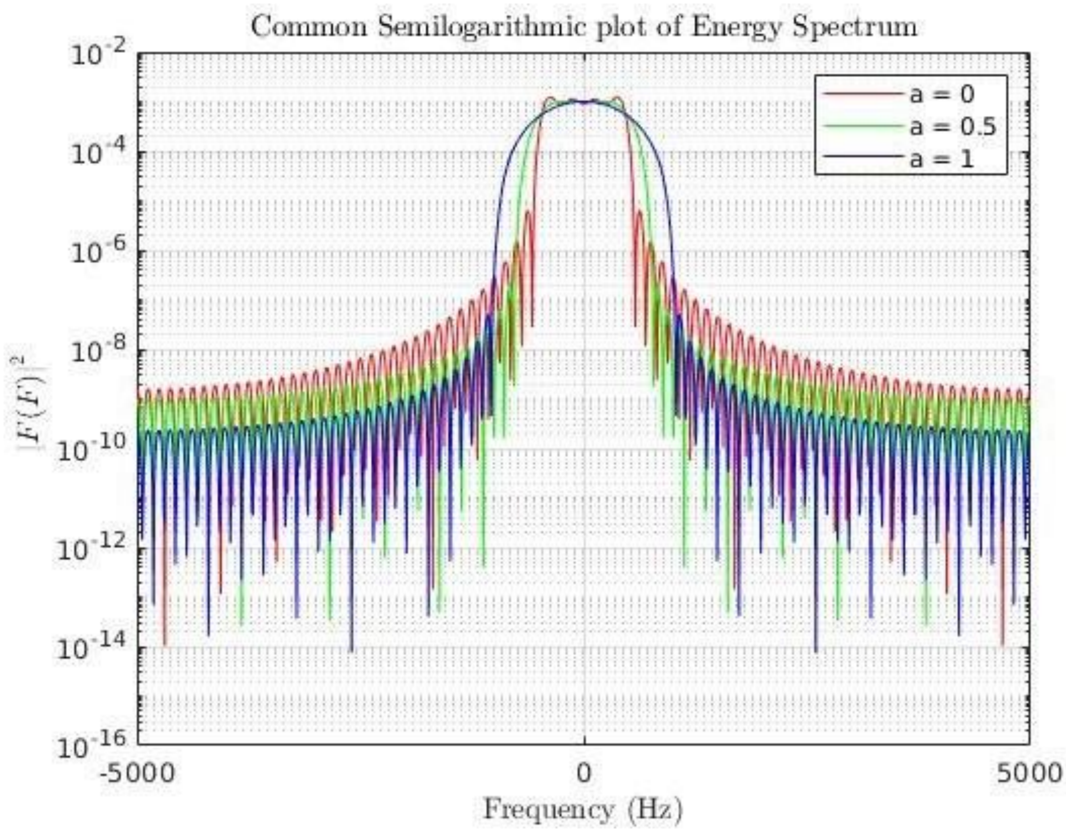
A2.

Χρησιμοποιώντας τις συναρτήσεις fft και fftshift για τον υπολογισμό των αντίστοιχων μετασχηματισμών Fourier Φ(F) σε δοθέν εύρος φάσματος ***[-Fs/2, Fs/2)***, το οποίο διαμορφώθηκε κατάλληλα ώστε να εμφανίζονται τα Nf ισαπέχοντα σημεία στον άξονα συχνοτήτων.
Στη συνέχεια, σχεδιάσθηκε η φασματική πυκνότητα ενέργειας των παλμών σε κοινό plot καιsemilogy ("συμπιεσμένη μορφή") για N=1024.

Σχηματικά το αποτέλεσμα (κοινό plot και των τριών):



"Συμπιεσμένη μορφή":



Σημείωση:

fft: Υπολογίζει τον μετασχ. Φουριέ
fftshift: Ολισθαίνει το αποτέλεσμα ώστε ο Μ.Φ που παράχθηκε να έχει κέντρο το μηδέν.

Εύκολα παρατηρούμε εντονότερες διακυμάνσεις για α=0 (***ripples)*** στο πρώτο γράφημα οι οποίες μειώνονται με την αύξηση του roll-of συντελεστή . Η semilogy που θα σχεδιάσει τον κατακόρυφο άξονα υπο λογαριθμική κλίμακα βελτιστοποιεί την ανάλυση σε περιοχές όπου η συνάρτηση λαβαίνει ιδιαιτέρως χαμηλές τιμές (εξ΄ου και "συμπίεση").
Συνεπώς η δεύτερη τεχνική μπορεί να χαρακτηρισθεί ώς αποδοτικότερη για μεταβολές σε μικρές τιμές.

Τέλος μπορεί να εξαχθεί το συμπέρασμα οτι η Φασματική Πυκνότητα Ενέργειας μειώνεται με την αύξηση του roll-of συντελεστή.

A3.

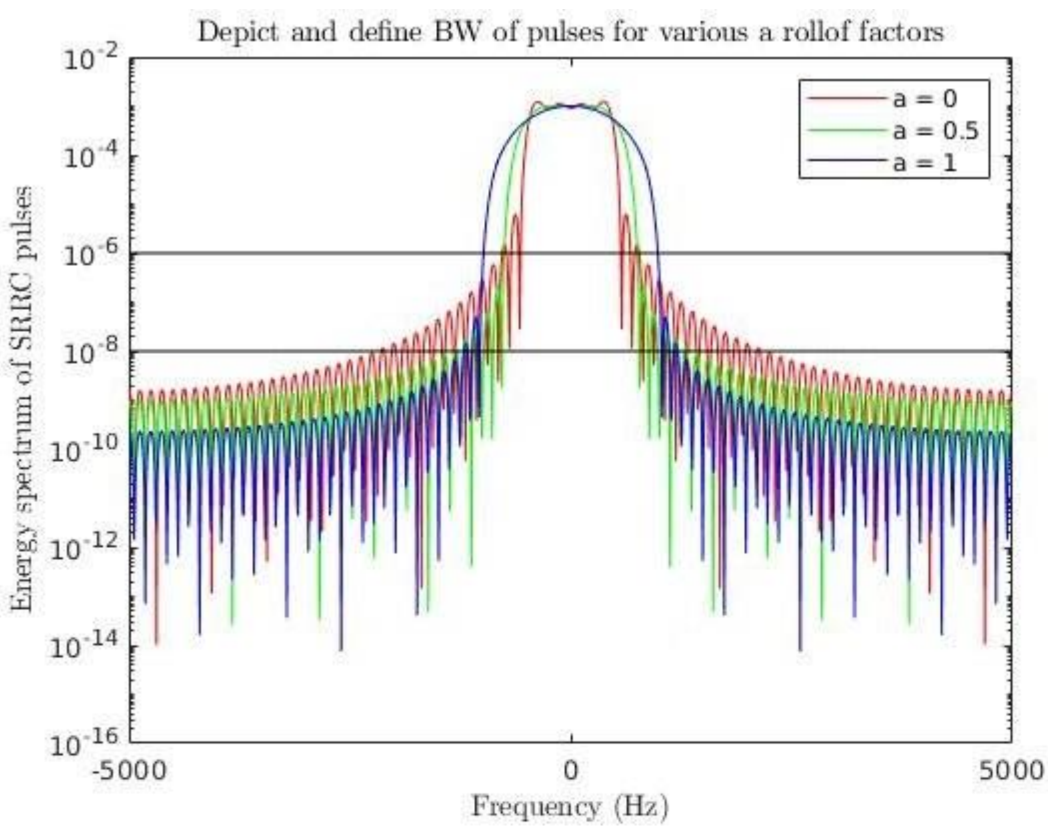
Θεωρητικά το εύρος φάσματος των άπειρης διάρκειας παλμών είναι $BW = \frac{(1+a)}{2T}$ επομένως θα προκύψουν οι εξής περιπτώσεις:

- α = 0: BW = 500.
- α = 0.5 BW = 750.
- α = 1 BW = 1000.

Εξαιτίας όμως του γεγονότος οτι οι αποκομμένοι παλμοί έχουν άπειρο εύρος φάσματος πρέπει να “κατασκευάσουμε” έναν πιο ρεαλιστικό ορισμό. Συνεπώς δρούμε ώς εξής:

Σχεδιάζουμε δύο ευθείες ***c1,c2*** κάτω απο τις οποίες θεωρούμε τυχών τιμές μηδενικές. Έτσι αφού η συχνότητα δεν μπορεί να λάβει αρνητικές τιμές ώς μέγεθος θα ισχύει ότι:

$\mathbf{f_{min}} = 0$
$\mathbf{f_{max}}$= BW (τελευταίο σ. τομής παλμού και ευθείας).



Κατόπιν οδηγίας οι ευθείες σχεδιάσθηκαν στα σημεία $(c1, c2)=(\frac{T}{10^3}, \frac{T}{10^5})=(10^{-6}, 10^{-8})$.

Συνεπώς μπορούμε εύκολα να διαπιστώσουμε οτι το ***κριτήριο*** για να χαρακτηρισθεί οποιοσδήποτε παλμός βέλτιστος ώς προς το εύρος ζώνης (BW) είναι ***ο ορισμός του μηδέν***.

Συγκεκριμένα στην πρώτη περίπτωση και με κριτήριο την ευθεία c1, ο παλμός με το μικρότερο bandwidth (άρα και ο αποδοτικότερος) είναι αυτός για α = 0, ενώ στην δεύτερη περίπτωση (με κριτήριο την ευθεία c2) ώς έτοιος προκύπτει ο παλμός για α = 0.5 .

Με χρήση του zoom-in μπορούμε ακόμη και να υπολογίσουμε προσεγγιστικά το εύρος ζώνης για κάθε παλμό υπό το κράτος του δοθέντος κριτηρίου (ευθείας).

c1

roll of factor (a)	***BW***
0	560
0.5	750
1	999

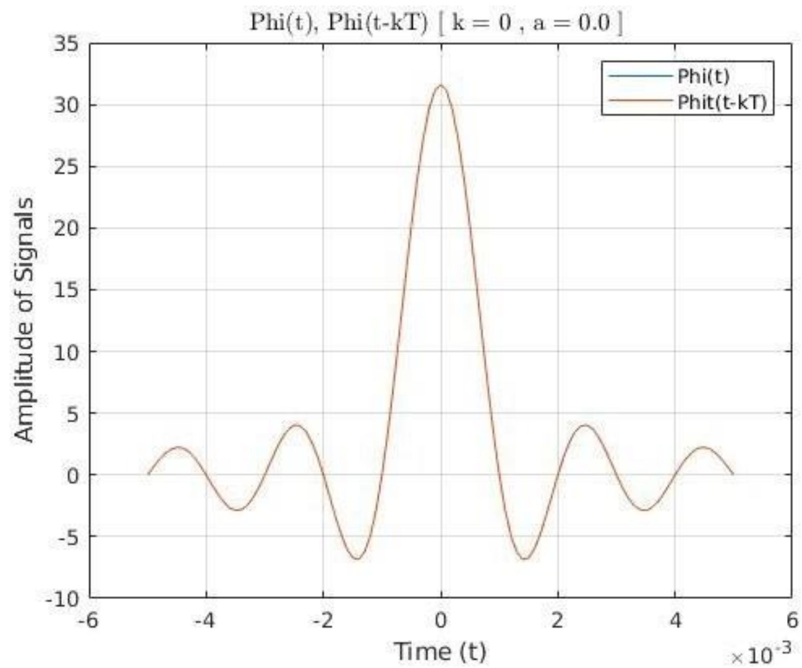
c2

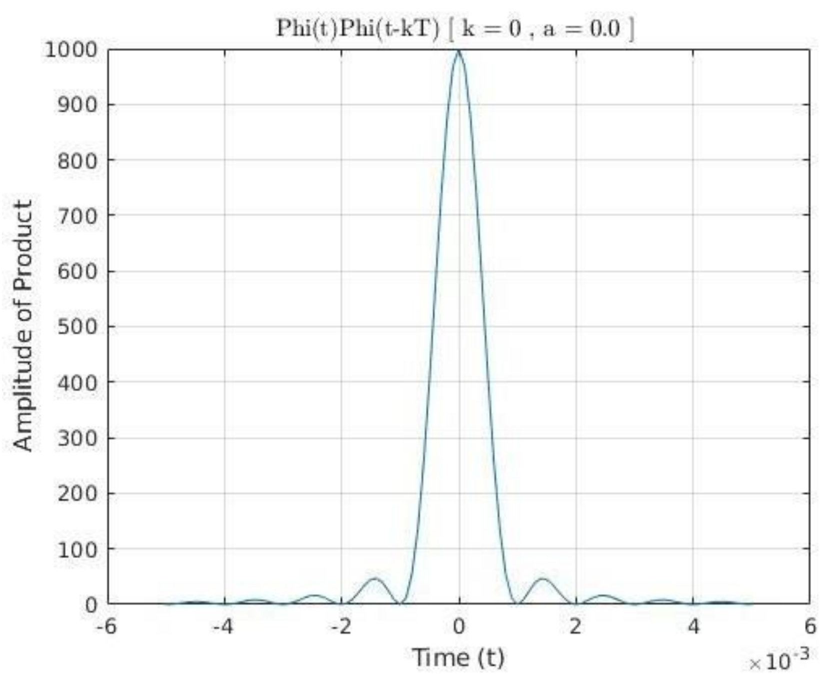
roll of factor (a)	***BW***
0	2000
0.5	1050
1	1150

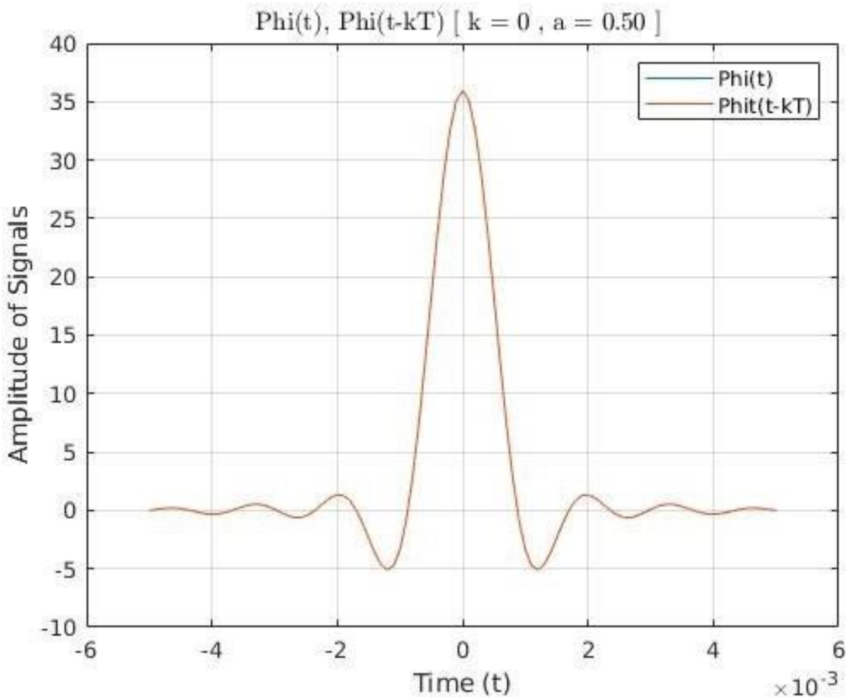
Μέρος Β:

Χρησιμοποιώντας τους παλμούς που αναπτύχθηκαν στο Μέρος Α, κατασκευάσθηκε μια επαναληπτική δομή με σκοπό την δημιουργία του μετατοπισμένου σήματος φ(t-kT) με k = {0,1,2,3} για όλες τις τιμές των διαφορετικών α (α = {0 0.5 1}).
Για την μετατόπιση χρησιμοποιήθηκε η γνωστή τεχνική του zero-padding ,ενώ εν τέλει σχεδιάσθηκαν τόσο τα αρχικά και μετατοπισμένα σήματα (σε κοινό plot) όσο και το γινόμενο αυτών.

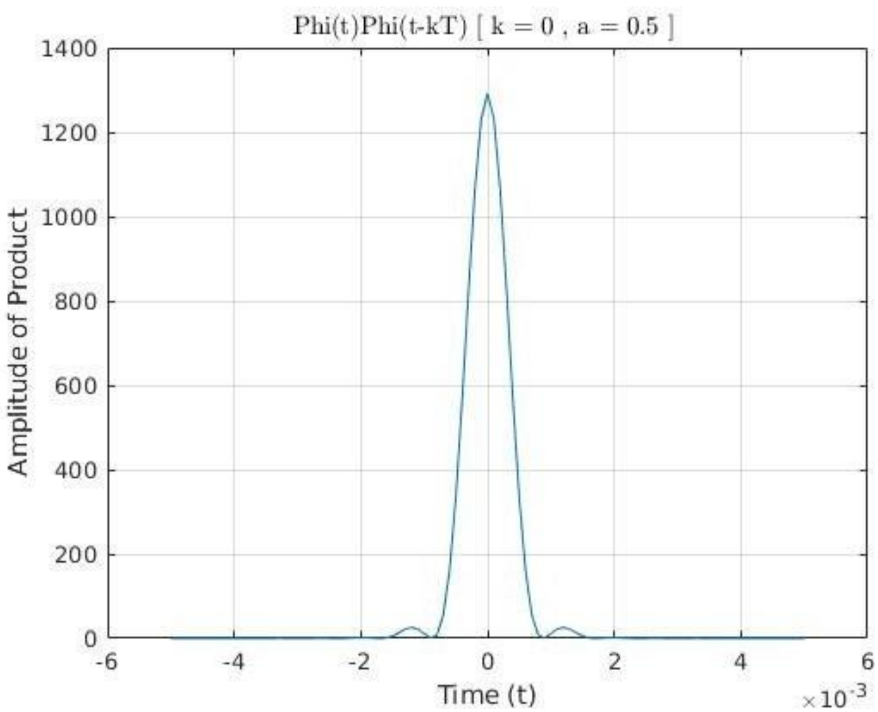
Ακολουθούν οι κυμματομορφές:



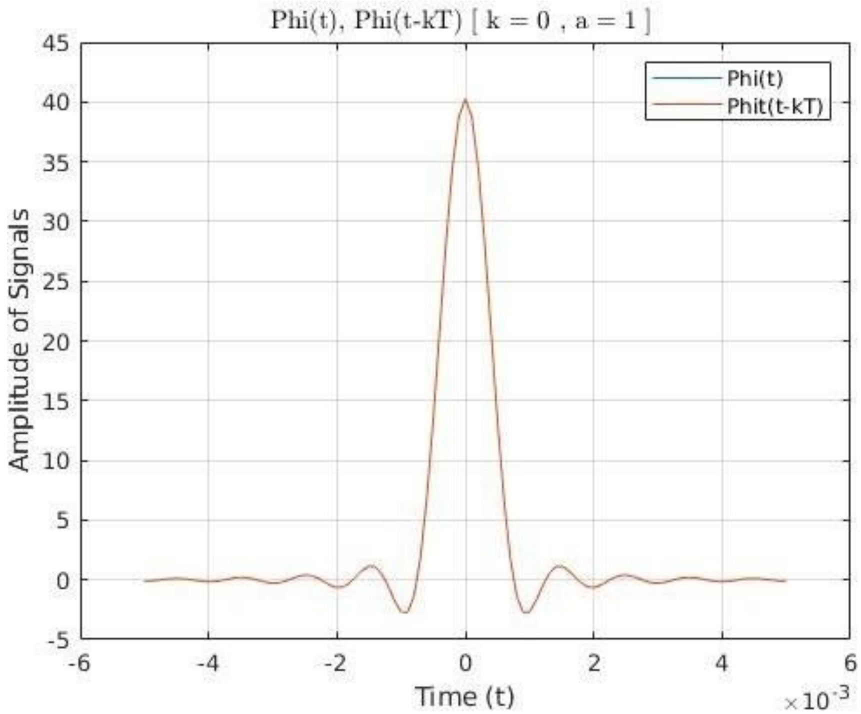




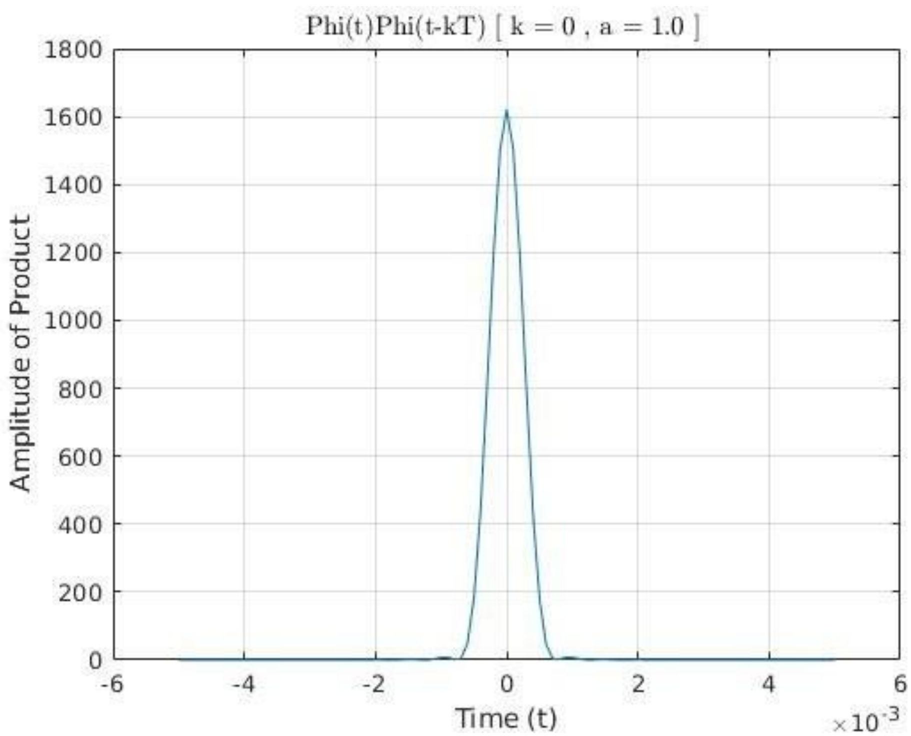
Phi(t), Phi(t-kT) [k = 0 , a = 0.50]
Phi(t)
Phit(t-kT)
Amplitude of Signals
Time (t)
×10^-3
40
35
30
25
20
15
10
5
0
-5
-10
-6
-4
-2
0
2
4
6



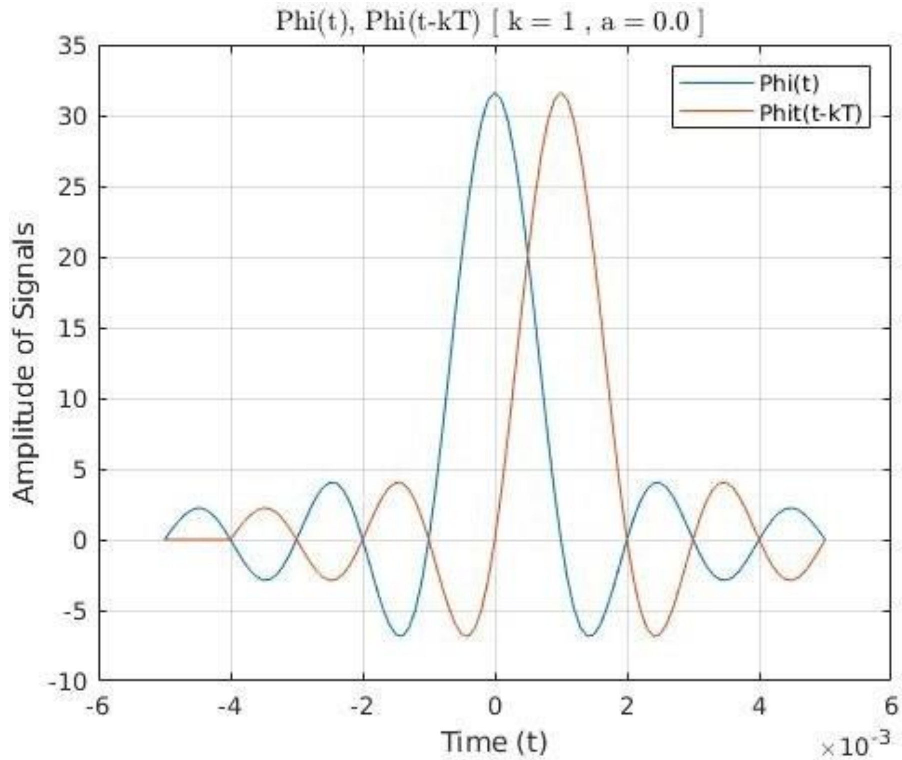
Phi(t)Phi(t-kT) [k = 0 , a = 0.5]
Amplitude of Product
Time (t)
×10^-3
1400
1200
1000
800
600
400
200
0
-6
-4
-2
0
2
4
6

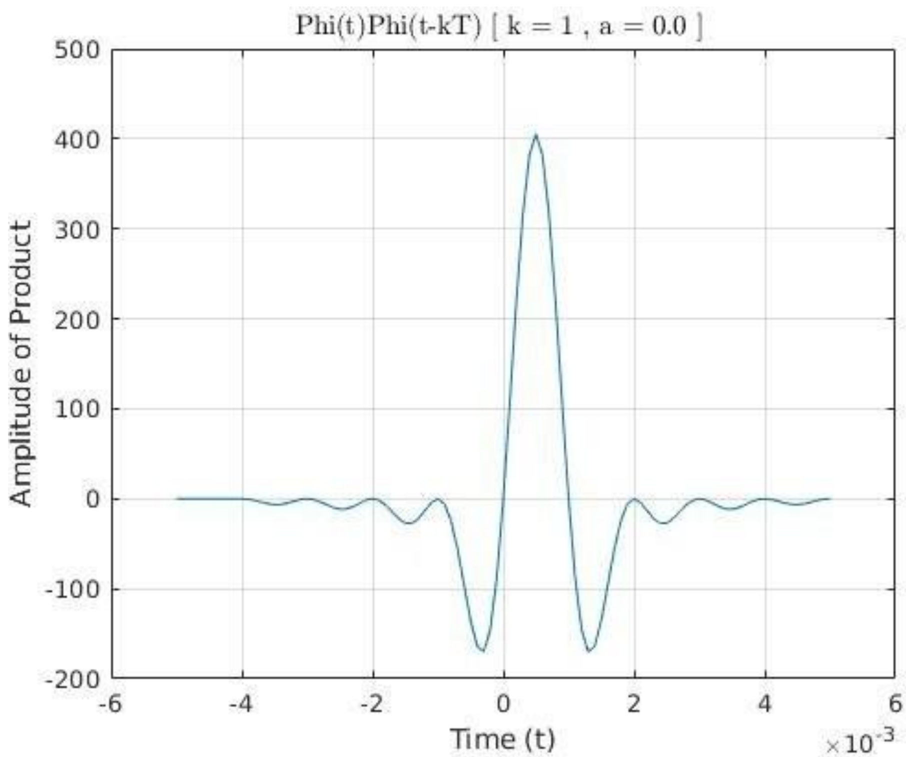


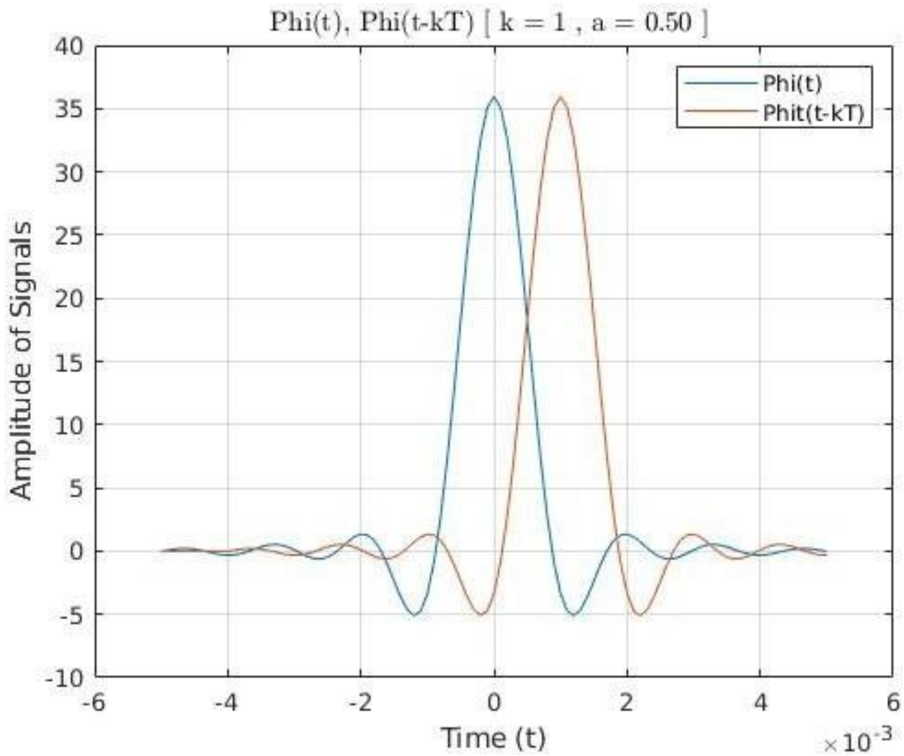
Phi(t), Phi(t-kT) [k = 0 , a = 1]
Phi(t)
Phit(t-kT)
Amplitude of Signals
Time (t)
×10^-3

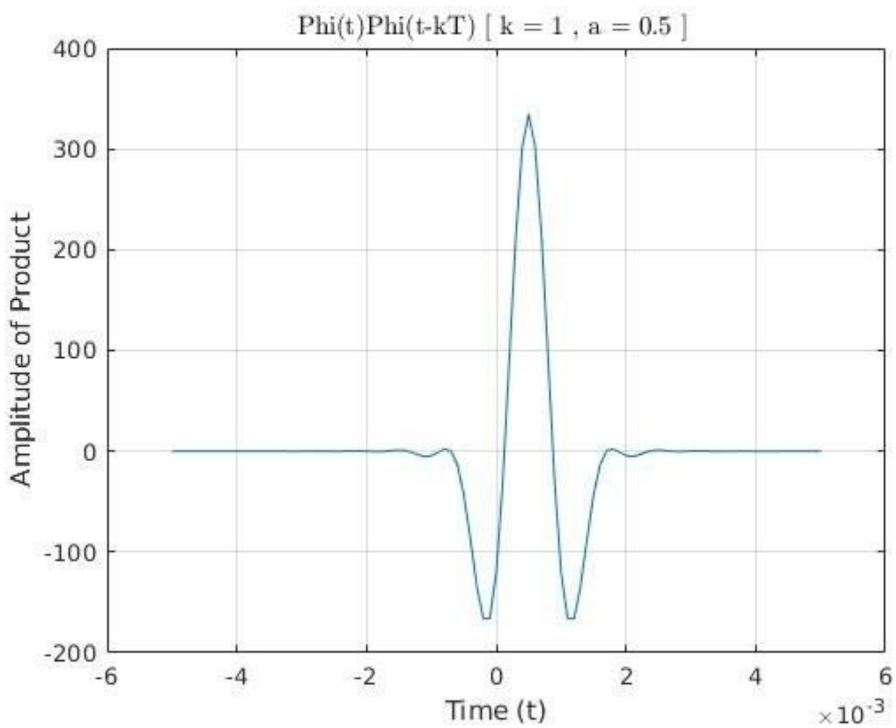


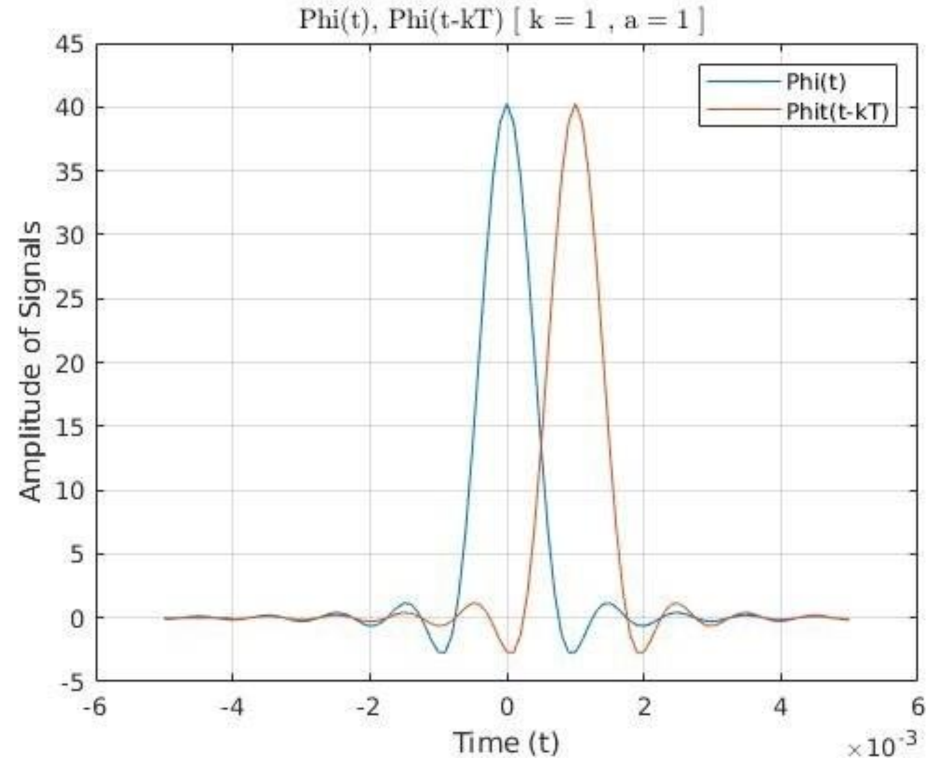
Phi(t)Phi(t-kT) [k = 0 , a = 1.0]
Amplitude of Product
Time (t)
×10^-3

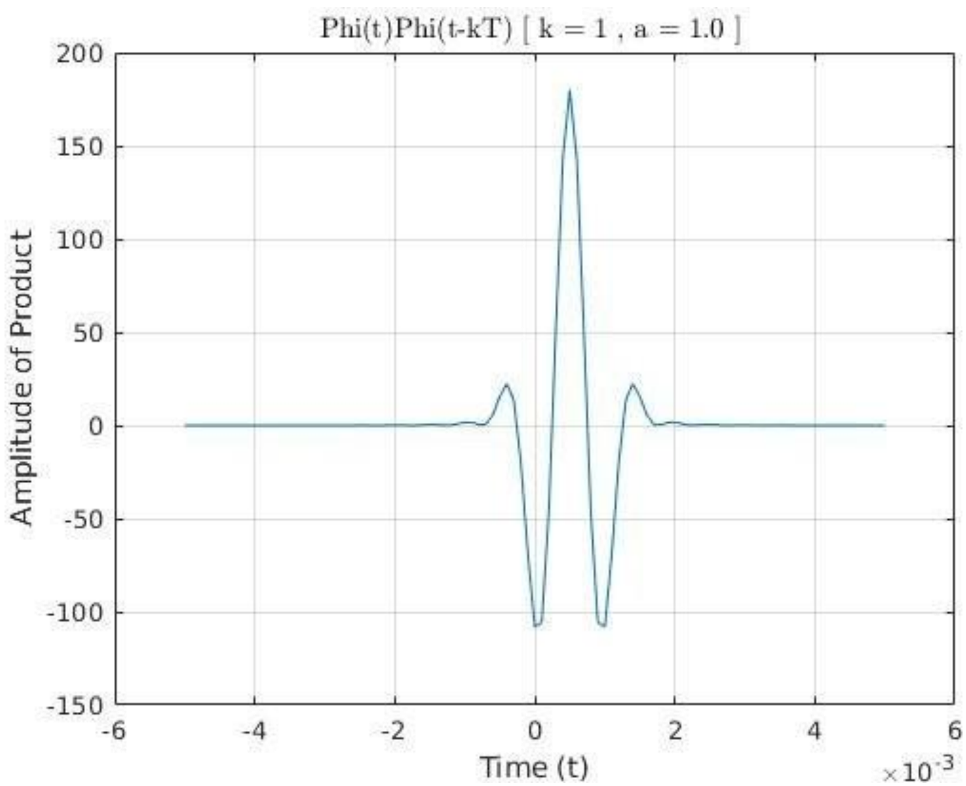

Phi(t), Phi(t-kT) [k = 1 , a = 0.0]
Phi(t)
Phit(t-kT)
Amplitude of Signals
Time (t)
×10^-3

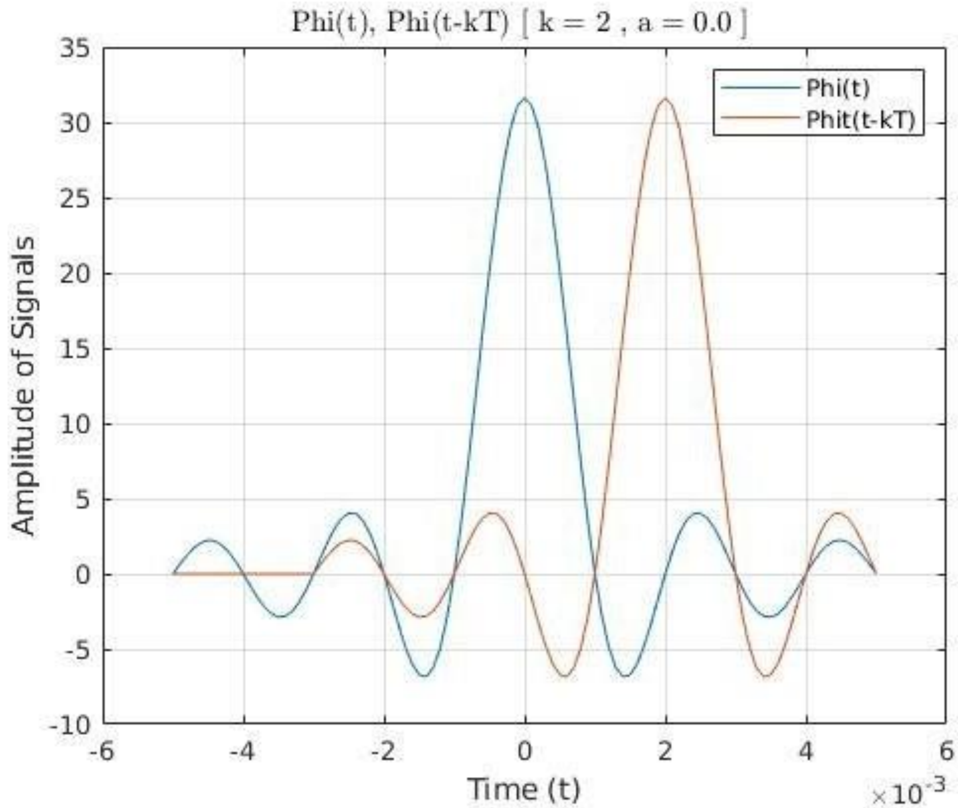

Phi(t)Phi(t-kT) [k = 1 , a = 0.0]
Amplitude of Product
Time (t)
×10^-3

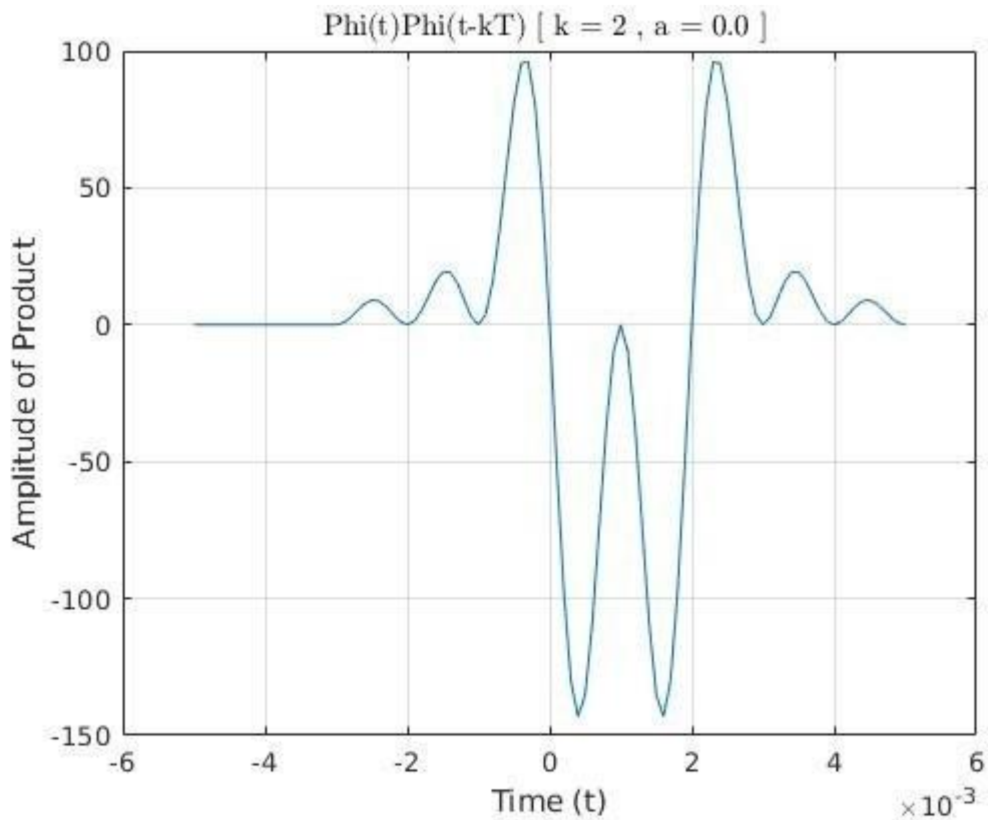

Phi(t), Phi(t-kT) [k = 1 , a = 0.50]
Phi(t)
Phit(t-kT)
Amplitude of Signals
Time (t)
×10^-3
40
35
30
25
20
15
10
5
0
-5
-10
-6
-4
-2
0
2
4
6

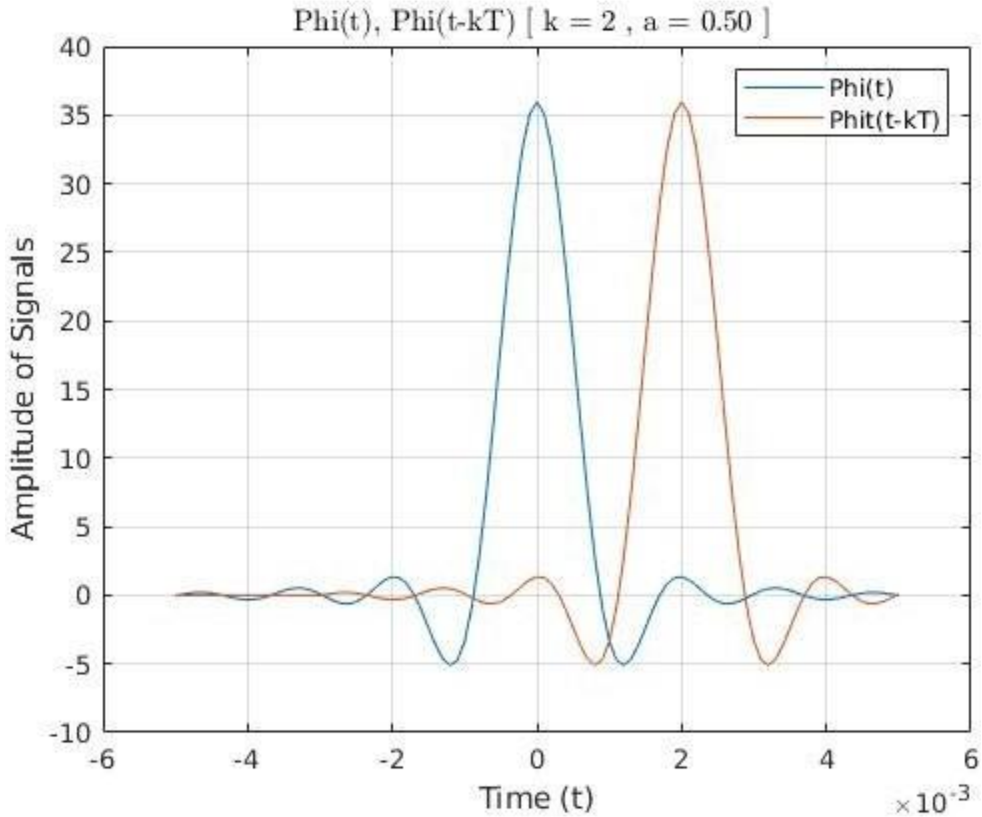

Phi(t)Phi(t-kT) [k = 1 , a = 0.5]
Amplitude of Product
Time (t)
×10^-3
400
300
200
100
0
-100
-200
-6
-4
-2
0
2
4
6

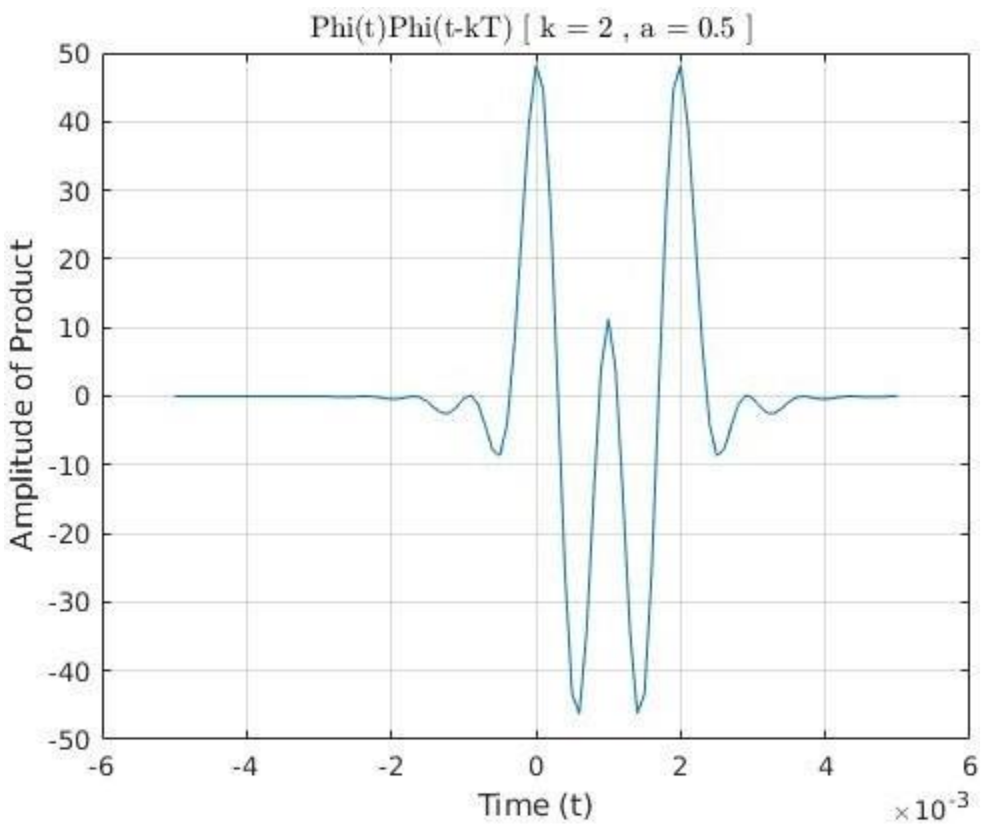

Phi(t), Phi(t-kT) [k = 1 , a = 1]
Phi(t)
Phit(t-kT)
Amplitude of Signals
Time (t)
×10^-3

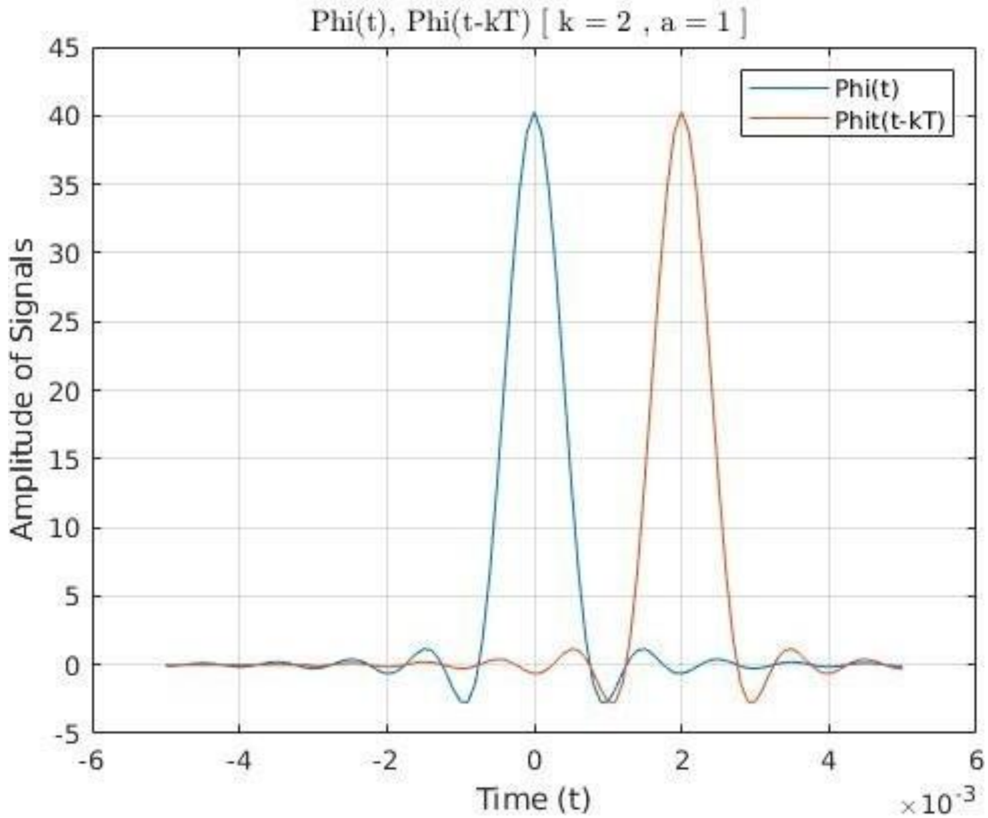

Phi(t)Phi(t-kT) [k = 1 , a = 1.0]
Amplitude of Product
Time (t)
×10^-3


Phi(t), Phi(t-kT) [k = 2 , a = 0.0]
Phi(t)
Phit(t-kT)
Amplitude of Signals
Time (t)
×10^-3
35
30
25
20
15
10
5
0
-5
-10
-6
-4
-2
0
2
4
6

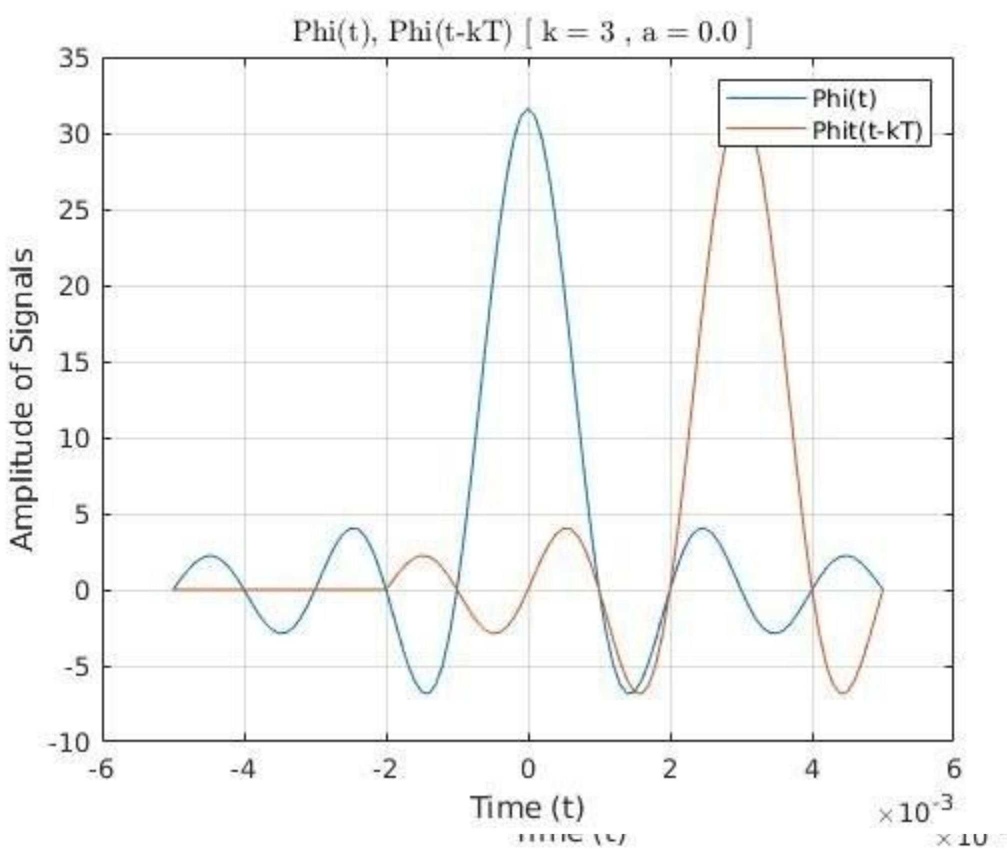

Phi(t)Phi(t-kT) [k = 2 , a = 0.0]
Amplitude of Product
Time (t)
×10^-3
100
50
0
-50
-100
-150
-6
-4
-2
0
2
4
6


Phi(t), Phi(t-kT) [k = 2 , a = 0.50]
Phi(t)
Phit(t-kT)
Amplitude of Signals
Time (t)
×10^-3
40
35
30
25
20
15
10
5
0
-5
-10
-6
-4
-2
0
2
4
6

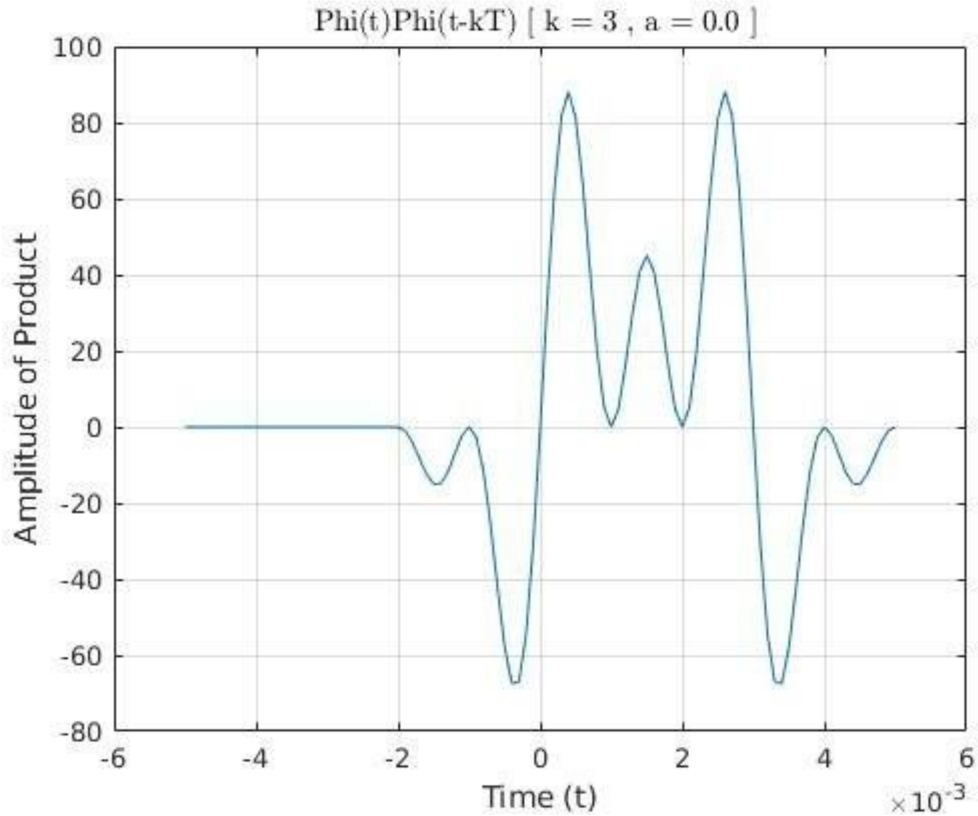

Phi(t)Phi(t-kT) [k = 2 , a = 0.5]
Amplitude of Product
Time (t)
×10^-3
50
40
30
20
10
0
-10
-20
-30
-40
-50
-6
-4
-2
0
2
4
6

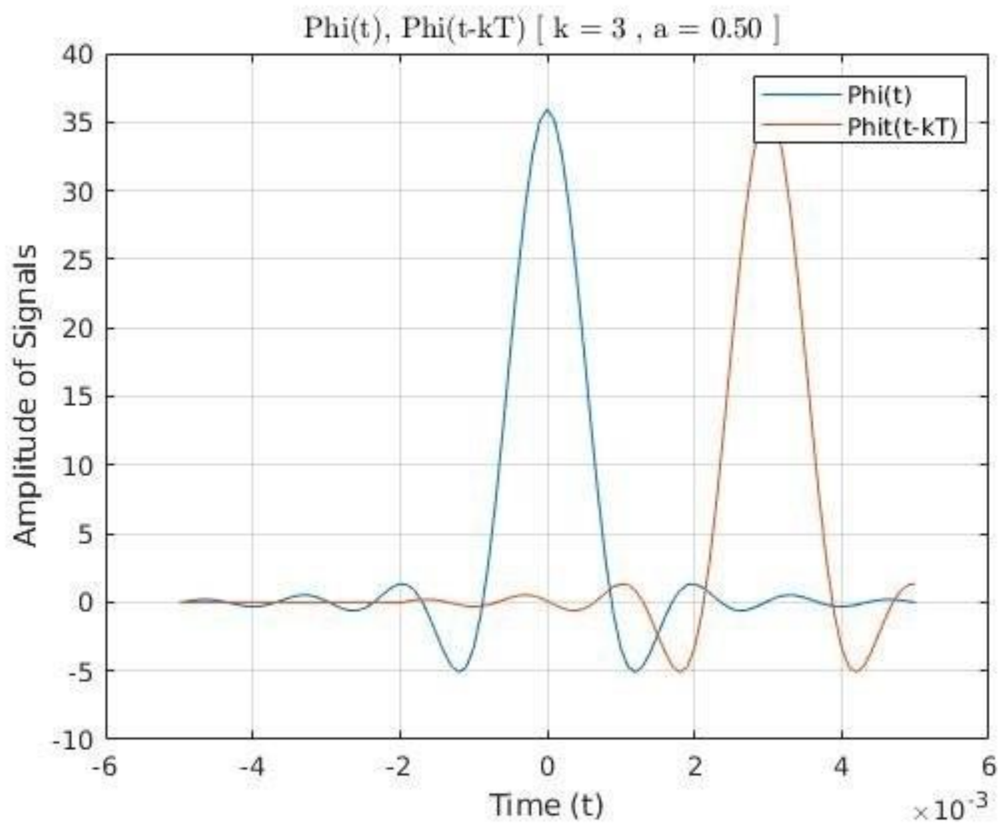


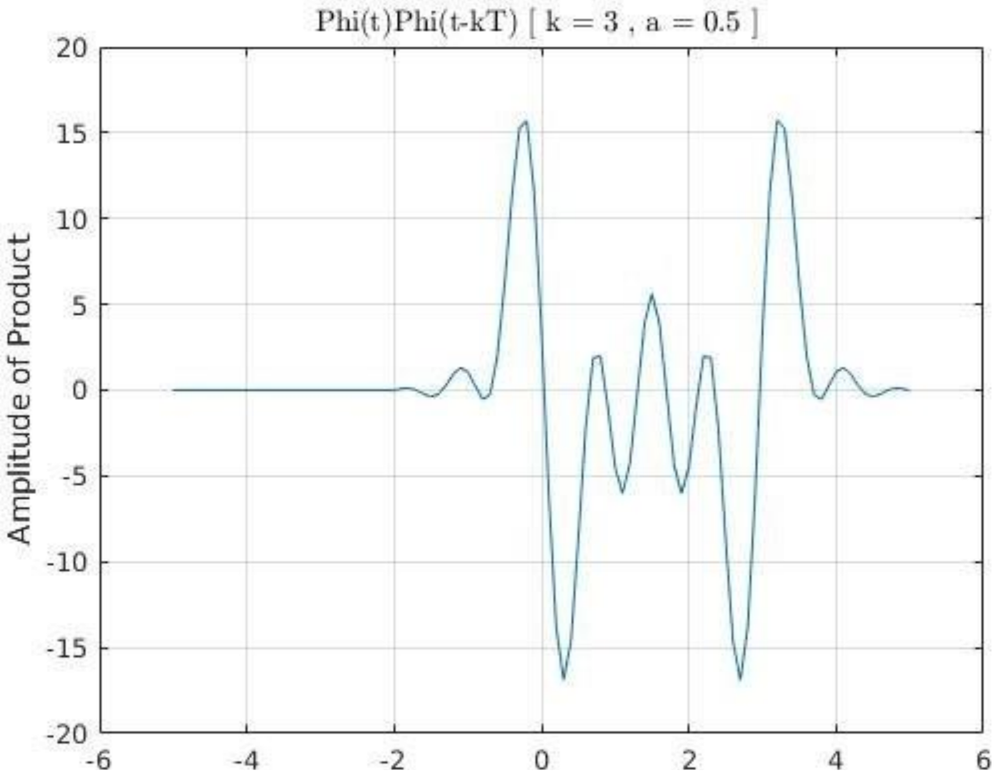
Phi(t), Phi(t-kT) [k = 2 , a = 1]
Phi(t)
Phit(t-kT)
Amplitude of Signals
Time (t)
×10^-3

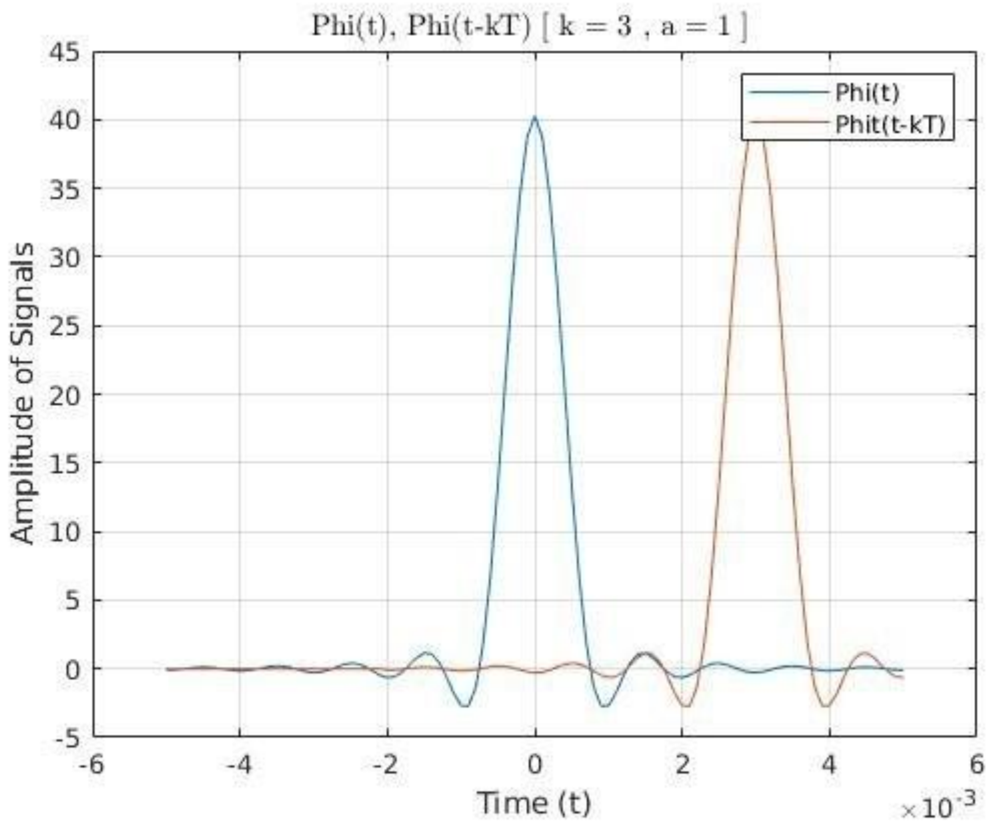


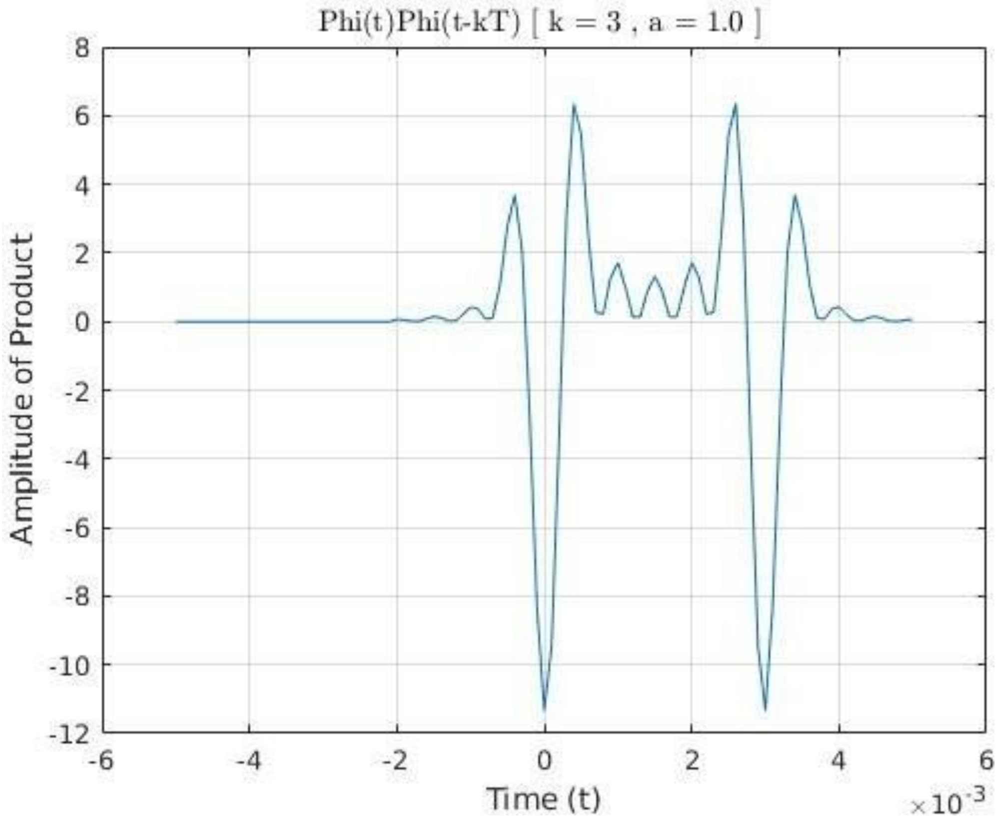
Phi(t), Phi(t-kT) [k = 3 , a = 0.0]
Phi(t)
Phit(t-kT)
Amplitude of Signals
Time (t)
×10^-3


Phi(t)Phi(t-kT) [k = 3 , a = 0.0]
Amplitude of Product
Time (t)
$\times 10^{-3}$


Phi(t), Phi(t-kT) [k = 3 , a = 0.50]
Phi(t)
Phit(t-kT)
Amplitude of Signals
Time (t)
$\times 10^{-3}$


Phi(t)Phi(t-kT) [k = 3 , a = 0.5]
Amplitude of Product
20
15
10
5
0
-5
-10
-15
-20
-6
-4
-2
0
2
4
6


Phi(t), Phi(t-kT) [k = 3 , a = 1]
Phi(t)
Phit(t-kT)
Amplitude of Signals
45
40
35
30
25
20
15
10
5
0
-5
-6
-4
-2
0
2
4
6
Time (t)
$\times 10^{-3}$



Ζητήθηκε πλέον αυτών ο υπολογισμός των ολοκληρωμάτων των γινομένων που παρουσιάσθηκαν ο οποίος θεωρητικά περιγράφεται ώς εξής:

$$\int_{-\infty}^{\infty} \varphi(\tau)\varphi(\tau - \kappa T)d\tau = \begin{Bmatrix} 1, \alpha\nu\ k = 0 \\ 0, \alpha\nu\ k \neq 0 \end{Bmatrix}$$

Ακολουθεί πίνακας των υπολογισθέντων, ιδιαίτερα μικρές τιμές οι οποίες ανάχθηκαν απο το λογισμικό σε μηδενικές ανακτήθηκαν απο τη βάση δεδομένων του λογισμικού (στη μορφή ***<num>e - <num>***).

	k=0	***k=1***	***k=2***	***k=3***
α=0	0.9798	0.0226	-0.0258	0.0308
α=0.5	0.9999	-7.2284e-06	1.5853e-04	3.4844e-05
α=1	1.0000	-2.2124e-05	-3.3230e-05	-5.7614e-05

Εύκολα αντιλαμβανόμαστε σύμφωνα με τα ανωτέρω πως οι αποκομμένοι SRRC παλμοί είναι ορθοκανονικοί (ώς προς τις μετατοπίσεις kT), έστω και προσεγγιστικά, με την προσέγγιση να βελτιώνεται για α>0.

Επιπλέον για ***k=0***, ικανοποιείται η ***ιδιότητα ορθοκανονικότητας*** που περιγράφηκε, ενώ το εμβαδόν είναι περίπου όμοιο με το θεωρητικό.

Μέρος Γ:

Στο τελευταίο κομμάτι της εργασίας προσομοιώθηκε ένα 2-PAM σύστημα βασικής ζώνης, το οποίο μεταφέρει N bits (με διαμόρφωση 2- PAM).

Γ1.
Δημιουργία N τυχαίων bits μέσω της εντολής ***(sign(randn(N,1))+1)/2,*** η rand θα δημιουργήσει τυχαίους πραγματικούς αριθμού τους οποίους η sign() θα μετατρέψει δεχόμενους ώς όρισμα είτε σε +1 ή σε -1.

Η προσθήκη μονάδας (+1) στην αρχική κλήση εξασφαλίζει την προσομείωση στην περίπτωση του μηδενός (-1 → 0).

Γ2.

Υλοποίηση συνάρτησης bits_2_PAM, η οποία με είσοδο την ακολουθία του Γ1, κάνει mapping τα bit ώς εξής:

$$0 \longrightarrow +1,$$

$$1 \longrightarrow -1.$$

Source code:

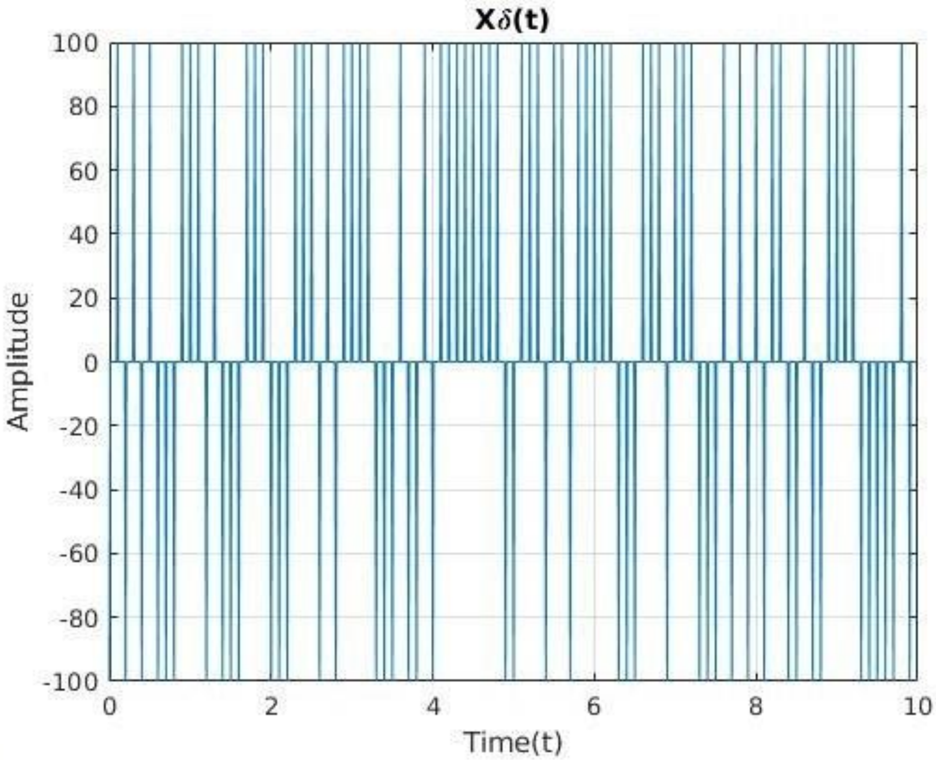
```
X=zeros(size(b)); %Initialise as zeros with respective length

for k=1:length(b)
    if(b(k)==0)
        X(k)=1;
    elseif(b(k)==1)
        X(k)=-1;
    else
        disp('Error')
        return
    end
end
end
```

$$X_\delta(t) = \sum_{k=0}^{N-1} X_k\, \delta(t - kT).$$

Συνεχίζοντας προσομοιώθηκε το σήμα

Το παραπάνω καθέστη δυνατό μέσω της εντολής ***X_delta= 1/T_s *upsample(X,over)*** η οποία δέχεται ώς όρισμα τα σύμβολα που προκύπτουν απο την 2-PAM και ένα διάστημα over, επιστρέφοντας over-1 μηδενικά τοποθετημένα σε ένα stream μεταξύ των συμβόλων (του Χ).

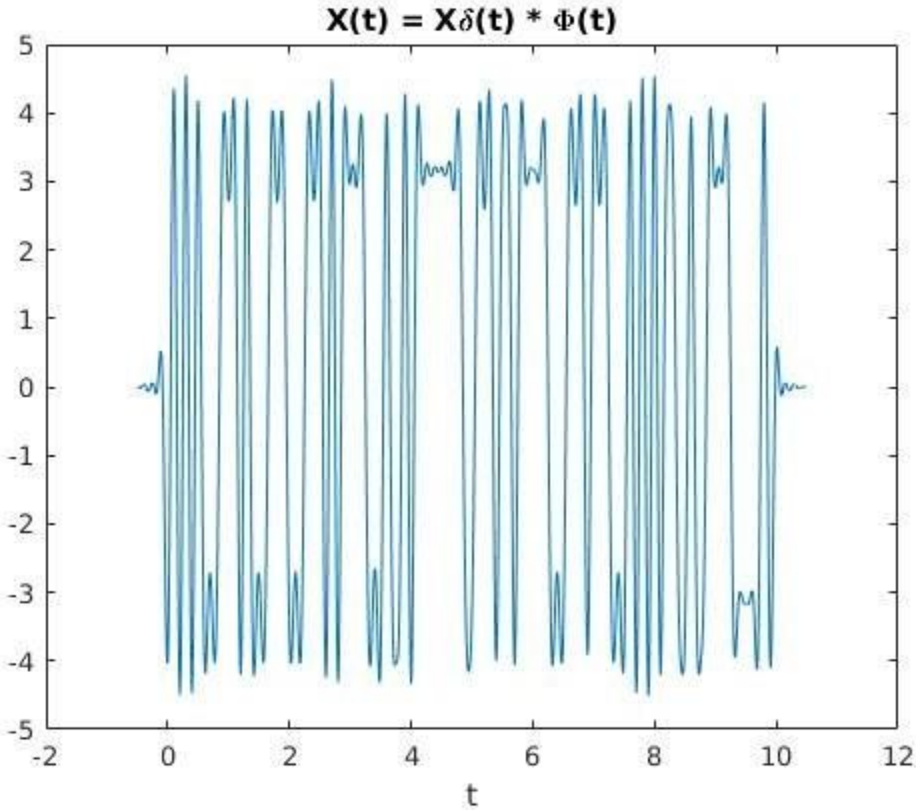
Ορίζωντας τον άξονα του χρόνου στο ***Δ= [0,(N+N(over-1))-1)Ts]***, πετυχαίνουμε **N(over-1)** μηδενικά μεταξύ συμβόλων με το αποτέλεσμα να εικονίζεται παρακάτω.



Πλέον αυτών δημιουργείται παλμός ***SRRC***, ο οποίος ***μέσω της conv() συνελίσεται*** με το άνωθεν σήμα. Ο ορισμός του διανύσματος χρόνου για αυτή τη διαδικασία είναι γνωστός και απο προηγούμενα μαθήματα ώς το άθροισμα ελαχίστων και το άθροισμα μεγίστων τιμών των χρονικών διαστημάτων των συμμετεχόντων σημάτων, δηλαδή:

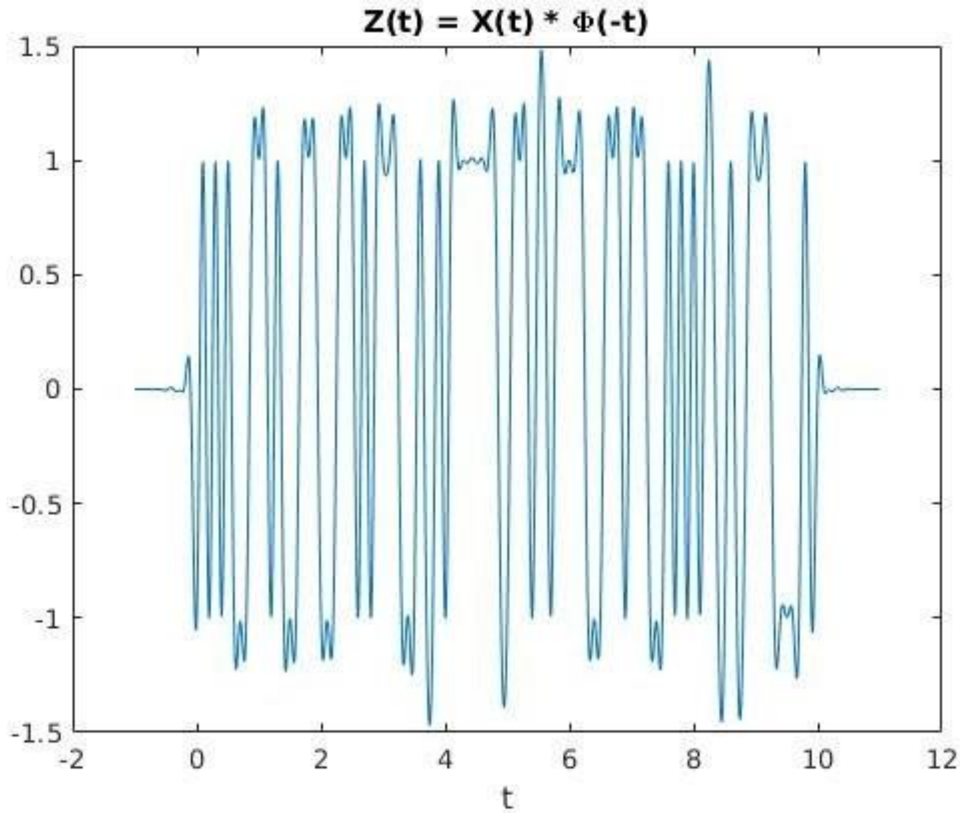
t_min(signal1) + t_min(signal2) = tmin_convolution
t_max(signal1) + t_max(signal2) = tmax_convolution

Το αποτέλεσμα ακολουθεί στη γραφική του μορφή:



Ακολούθως "κατασκευάζεται" και η συνέλιξη του σήματος που προέκυψε (X(t)) με τον ανακλασμένο Φ(-t). Εδώ τόσο ο χρόνος όσο και το ίδιο το διάνυσμα κατασκευάσθηκαν με ανακλάσεις του αρχικού.

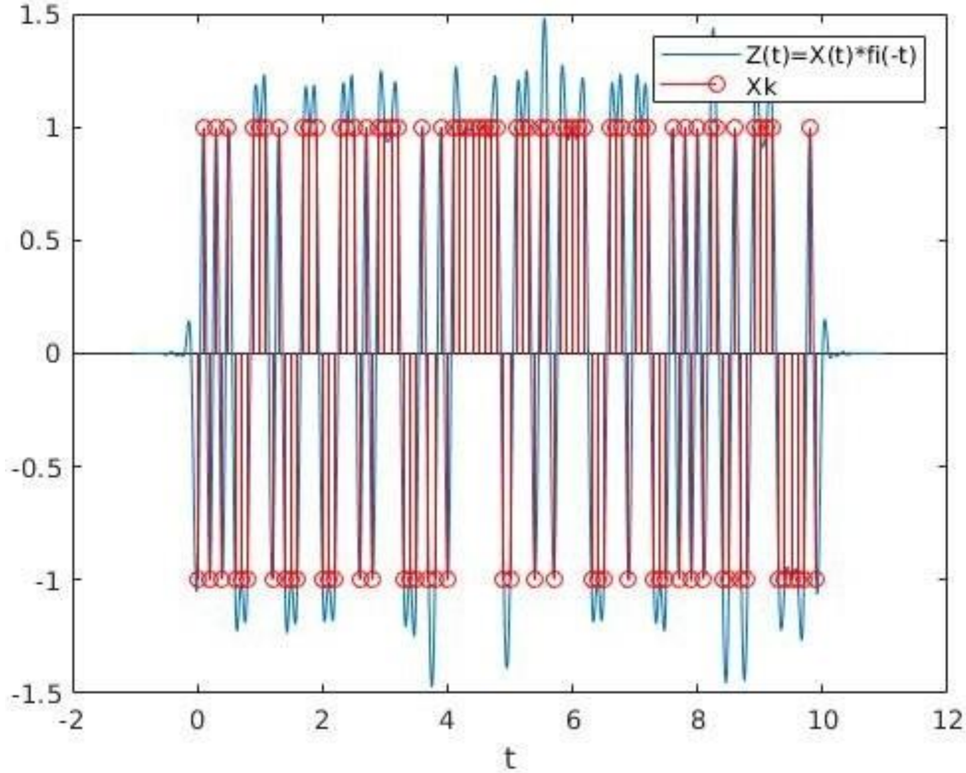
Τέλος συγκρίνουμε γραφικά την Z(kT) με τα Xk εκτελώντας και την ***stem([0:N-1]*T,X)***. Το αποτέλεσμα είναι τιμές του Xk διανύσματος να ταυτίζονται με τις Z(kT).



Η διαδικασία αυτή μπορεί να χαρακτηρισθεί και ώς ***"φιλτράρισμα με προσαρμωσμένο φίλτρο"*** (ιδανικό με βάση θεωρία).

Δηλαδή μπορούμε να ανακτήσουμε το αρχικό stream πληροφορίας δειγματοληπτώντας το Z.

Συνεπώς η συνολική διαδικασία του Μέρους Γ μπορεί να χαρακτηρισθεί και ***"διαμόρφωση και αποδιαμόρφωση σήματος σε ιδανικό κανάλι"***.



Παράρτημα – Κώδικας:

```
clear all
close all
clc

%A.1

%define values under given assumptions


T = 10^(-3);
over = 10;
Ts = T/over;
A = 4;

%various roll-off factor values, will be simulated.

[phi1, t1] = srrc_pulse(T,Ts,A,0);
[phi2, t2] = srrc_pulse(T,Ts,A,0.5);
[phi3, t3] = srrc_pulse(T,Ts,A,1);

%open a figure, plot some stuff then hold until done plotting.

figure;
plot(t1,phi1,'r')
hold on;
plot(t2,phi2,'g')
plot(t3,phi3,'b')
legend('a = 0', 'a = 0.5', 'a = 1');
xlabel('Time (sec)', 'Interpreter', 'latex');
ylabel('SRRC pulses', 'Interpreter', 'latex');


title('SRRC pulses for various rollof factor values (a $$ \in {0, 0.5 , 1} $$)', 'Interpreter', 'latex')
grid on;
hold off;



%A.2

%given data
Nf=1024;
Fs=1/Ts;

%shifted signals

fft_phi1 = fftshift(fft(phi1,Nf)*Ts);
fft_phi2 = fftshift(fft(phi2,Nf)*Ts);
fft_phi3 = fftshift(fft(phi3,Nf)*Ts);

%defining the frequency axis - vector.

F=-Fs/2:Fs/Nf:Fs/2-Fs/Nf;

%spectrum of shifted signals

spec_phi1 = abs(fft_phi1).^2;
spec_phi2 = abs(fft_phi2).^2;
spec_phi3 = abs(fft_phi3).^2;
```

```
%plotting the energy spectrum
figure;
plot(F,spec_phi1,'r')
title('Common plot of Energy Spectrum', 'Interpreter', 'latex');
hold on;

plot(F,spec_phi2,'g')
plot(F,spec_phi3,'b')
legend('a = 0', 'a = 0.5', 'a = 1');
xlabel('Frequency (Hz)', 'Interpreter', 'latex');
ylabel('${\left|F(F)\right|}^{2}$ ', 'Interpreter', 'latex');
grid on;


%semilogarithmic plot on a new figure

figure;
semilogy(F,spec_phi1,'r')
hold on;
semilogy(F,spec_phi2,'g')
semilogy(F,spec_phi3,'b')
legend('a = 0', 'a = 0.5', 'a = 1');
xlabel('Frequency (Hz)', 'Interpreter', 'latex');
ylabel('${\left|F(F)\right|}^{2}$ ', 'Interpreter', 'latex');
title('Common Semilogarithmic plot of Energy Spectrum', 'Interpreter', 'latex')
grid on;
hold off;


%A.3


%defining the theoritical bandwidth [BW = (1+a)/2T]

B1=1/(2*T);
B2=1.5/(2*T);
B3=1/T;


%cut-off lines

c1=T/(10.^3);
c2=T/(10.^5);


%semilogarithmic plot

figure;
semilogy(F,spec_phi1,'r')
hold on;
semilogy(F,spec_phi2,'g')
semilogy(F,spec_phi3,'b')
semilogy([F(1) F(end)],[c1 c1],'k')
semilogy([F(1) F(end)],[c2 c2],'k')
legend('a = 0', 'a = 0.5', 'a = 1');
xlabel('Frequency (Hz)', 'Interpreter', 'latex');
ylabel('Energy spectrum of SRRC pulses', 'Interpreter', 'latex');
title('Depict and define BW of pulses for various a rollof factors', 'Interpreter', 'latex');
hold off;



%B - remove clearance if u have enough horsepower in ur GPU,RAM [I don't].
```

```
close all
clear all
clc


%redefining stuff.

T = 10.^(-3);
over = 10;
A = 5;
Ts = T/over;
k=0;
a=[0 0.5 1]; %roll-off factors gathered in a vector



%original signals construction

[phi1, t1] = srrc_pulse(T, Ts, A, a(1));
[phi2, t2] = srrc_pulse(T, Ts, A, a(2));
[phi3, t3] = srrc_pulse(T, Ts, A, a(3));

figure;
plot(t1,phi1,'r')
hold on;
plot(t2,phi2,'g');
hold on;
plot(t3,phi3,'b');
legend('a = 0', 'a=0.5', 'a = 1');
grid on;
title ('Original Signal for various a (roll - of factor) values', 'Interpreter', 'latex');
hold off;


%useful vectors


kVector=0:2*A;


% initialization

integr1=zeros(1,length(kVector));
integr2=zeros(1,length(kVector));
integr3=zeros(1,length(kVector));

for j=1:length(a)

    for k=0:2*A

        %zero-padding and concatenate
        phi1_kT=[zeros(1,(1/Ts)*k*T) phi1(1:end-(1/Ts)*k*T)];
        phi2_kT=[zeros(1,(1/Ts)*k*T) phi2(1:end-(1/Ts)*k*T)];
        phi3_kT=[zeros(1,(1/Ts)*k*T) phi3(1:end-(1/Ts)*k*T)];

        %products
        prod1=phi1.*phi1_kT;
        prod2=phi2.*phi2_kT;
        prod3=phi3.*phi3_kT;

        %intergrals

        integr1(k+1)=sum(prod1)*Ts;
        integr2(k+1)=sum(prod2)*Ts;
```

```
integr3(k+1)=sum(prod3)*Ts;

%print job for respective rollof factor (a) values and specific
%k-Values


if(( k ==0) || (k ==1) || ( k == 2) || (k == 3))



 if( j == 1 )  %case a = 0

   %draw orig. and delayed signals
   capt=sprintf('Phi(t), Phi(t-kT) [ k = %d  , a = %.1f ] ',k,a(j));
   figure;
   plot(t1,phi1,t1,phi1_kT);
   legend('Phi(t)','Phit(t-kT)');
   grid on;
   title(capt, 'Interpreter', 'latex');
   xlabel('Time (t)');
   ylabel('Amplitude of Signals');

   %draw product of vectors
   capt=sprintf('Phi(t)Phi(t-kT) [ k = %d  , a = %.1f ] ',k,a(j));
   figure;
   plot(t1,prod1);
   grid on;
   title(capt, 'Interpreter', 'latex');
   xlabel('Time (t)');
   ylabel('Amplitude of Product');

 elseif( j == 2 ) %case a = 0.5

   %draw orig. and delayed signals
   capt=sprintf('Phi(t), Phi(t-kT) [ k = %d  , a = %.2f ] ',k,a(j));
   figure;
   plot(t2,phi2,t2,phi2_kT);
   legend('Phi(t)','Phit(t-kT)');
   grid on;
   title(capt, 'Interpreter', 'latex');
   xlabel('Time (t)');
   ylabel('Amplitude of Signals');

    %draw product of vectors
   capt=sprintf('Phi(t)Phi(t-kT) [ k = %d  , a = %.1f ] ',k,a(j));
   figure;
   plot(t2,prod2);
   grid on;
   title(capt, 'Interpreter', 'latex');
   xlabel('Time (t)');
   ylabel('Amplitude of Product');

 else   %case a = 1



   %draw orig. and delayed signals
   capt=sprintf('Phi(t), Phi(t-kT) [ k = %d  , a = %d ] ',k,a(j));
   figure;
   plot(t3,phi3,t3,phi3_kT);
   legend('Phi(t)','Phit(t-kT)');
   grid on;
   title(capt, 'Interpreter', 'latex');
   xlabel('Time (t)');
   ylabel('Amplitude of Signals');
```

```
        %draw product of vectors
        capt=sprintf('Phi(t)Phi(t-kT) [ k = %d  , a = %.1f ] ',k,a(j));

        figure;
        plot(t3,prod3);
        grid on;
        title(capt, 'Interpreter', 'latex');
        xlabel('Time (t)');
        ylabel('Amplitude of Product');

       end


      end
    end
end


%display the integrals, low values will be floored to zero by default

disp('Integral of product (a = 0) , K in [0,10] : ');
disp(integr1)
disp('Integral of product  (a = 0.5), K in [0,10] : ');
disp(integr2)
disp('Integral of product  (a = 1), K in [0,10] : ');
disp(integr3)


%C - comment clearance if available horsepower is present.
clear all
close all
clc

a = 0.5;
A = 5;
T = 0.1;
over = 10;
Ts = T/over;
N = 100;

%[c1]
%construct N random bits

b = (sign(randn(N, 1)) + 1)/2;


%[c2]
%transform the N-bits Vector created in 2PAM
X = bits_to_2PAM(b);

%simulate X(delta(t))=sum(Xk*delta(t-kT)) [by default]
xDelta=1/Ts*upsample(X,over);

%time Vector
%adds over-1 zeros between symbols respectivelly
tVector=0:Ts:(N+N*(over-1)-1)*Ts;

%drawing xDelta signal
figure;
plot(tVector,xDelta);
grid on;
xlabel('Time(t)');
ylabel('Amplitude');
```

```
title('X\delta(t)');


%Phi(t) signal pulse generation
[phi,t_phi] = srrc_pulse(T,Ts,A,a);

%time Vector for convolution

convTime =tVector(1)+t_phi(1):Ts:tVector(end)+t_phi(end);


%implement the convolution between the two signals
x=conv(xDelta,phi)*Ts;

%draw the result

figure;
plot(convTime,x);
xlabel('t');
title('X(t) = X\delta(t) * \Phi(t)');


%constructing second signal,Phi(-t), by inverting time Axis and Values

phi_Inv=phi(end:-1:1);
t_Inv= -t_phi(end:-1:1);

%convoluting inverted signal with Xdelta

z=conv(x,phi_Inv)*Ts;

%new time Vector for the second convolution
convTime_Inv=convTime(1)+t_Inv(1):Ts:convTime(end)+t_Inv(end);

%draw the result
figure;
plot(convTime_Inv,z);
xlabel('t');
title('Z(t) = X(t) * \Phi(-t)');


figure;
plot(convTime_Inv,z);
hold on;
stem([0:N-1]*T,X,'r');
xlabel('t');
legend('Z(t)=X(t)*fi(-t)','X{k}')
```

Συναρτήσεις:

```
function X = bits_to_2PAM ( b )
% X = bits_to_2PAM(b)
% OUTPUT
%
Xk: Xk, k=0,...,N-1
% INPUT
%
b: sequence of bits% USAGE:
% Map the input bits as shown below:
%
% Inp.
Output.
% 0
-> +1
% 1
-> -1
%
% Konstantinos T. Pantelis (the whole ECE Dept. as well has a prototype of this function)
X=zeros(size(b)); %Initialise as zeros with respective length
for k=1:length(b)
if(b(k)==0)
X(k)=1;
elseif(b(k)==1)
X(k)=-1;
else
disp('Error')
return
end
end
end
function [phi, t] = srrc_pulse(T, Ts, A, a)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%
% phi = srrc_pulse(T, Ts, A, a)
%
% OUTPUT
%
%
phi: truncated SRRC pulse, with parameter T,
%
%
roll-off factor a, and duration 2*A*T
%
%
t: time axis of the truncated pulse
%
% INPUT
%
% T: Nyquist parameter or symbol period (positive real number)
%
% Ts: sampling period (Ts=T/over)
%
%
where over is a positive INTEGER called oversampling factor %
% A: half duration of the pulse in symbol periods (positive INTEGER) %
%
a: roll-off factor (real number between 0 and 1)
%
%
%
% A. P. Liavas, Nov. 2011
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%
```

```
t = [-A*T:Ts:A*T] + 10^(-8); % in order to avoid division by zero problems at t=0.
if (a>0 && a<=1)
num = cos((1+a)*pi*t/T) + sin((1-a)*pi*t/T) ./ (4*a*t/T);
denom = 1-(4*a*t./T).^2;
phi = 4*a/(pi*sqrt(T)) * num ./ denom;
elseif (a==0)
phi = 1/(sqrt(T)) * sin(pi*t/T)./(pi*t/T);
else
phi = zeros(length(t),1);
disp('Illegal value of roll-off factor')
return
end
```