ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧ/ΚΩΝ ΚΑΙ ΜΗΧ/ΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

'ΨΗΦΙΑΚΗ ΕΠΕΞΕΡΓΑΣΙΑ ΣΗΜΑΤΟΣ'

ΑΚΑΔ. ΕΤΟΣ: 2019-20

ΕΡΓΑΣΙΑ MATLAB

Ονοματεπώνυμο: ΚΟΥΚΟΥΓΙΑΝΝΗΣ ΔΗΜΗΤΡΙΟΣ

ΜΕΡΟΣ Α: ΣΧΕΔΙΑΣΗ ΦΙΛΤΡΩΝ

Στο μέρος Α ζητείται η σχεδίαση 2 ψηφιακών φίλτρων (IIR και FIR), τα οποία έχουν τα παρακάτω χαρακτηριστικά:

- Ζώνη διάβασης ορισμένη από $\omega_p = 0.10\pi$.
- Ζώνη αποκοπής ορισμένη από $\omega_s = 0.30\pi$.
- Μέγιστο κυματισμό στη ζώνη διάβασης ορισμένη από την Rp = 1.00 dB.
- Μέγιστο κυματισμό στη ζώνη διάβασης ορισμένη από την As = 40.00 dB.

Σχεδίαση ως FIR με παράθυρο Hamming:

-

Δίνεται ο κώδικας της σχεδίασης:

```
function FIR_hamm = FIR()

Rf_p = 0.1; %% Syxnothta gia th zwnh diavashs

Rf_s = 0.3; %% Syxnothta gia th zwnh apokophs


%% Den pollaplasiazetai me pi epeidh auto to kanei to fir1


Rf_diff = Rf_s - Rf_p; %% Domega tou zhtoumenou filtrou

cutoff = (Rf_p + Rf_s) / 2; %% Omega idanikou katwperatou filtrou


filter_length = 8*pi / (Rf_diff*pi);

FIR_hamm = fir1(filter_length, cutoff); %% Xrhsh hamming me mhkos = filter_length + 1.


figure(1); impz(FIR_hamm, 1); title('Impulse response of FIR'); %% arithmhths FIR_hamm PARONOMASTIS 1

figure(2); stepz(FIR_hamm, 1); title('Step response of FIR');

figure(3); freqz(FIR_hamm, 1, 512); title('Frequency response of FIR');

figure(4); grpdelay(FIR_hamm, 1, 512); title('Group delay of FIR');
```

Αρχικά, ορίζουμε τις παραμέτρους του συστήματος. Υπογραμμίζεται ότι στις γωνιακές συχνότητες (Rf_p, Rf_s) δεν ορίζονται ως 0.1π και 0.3π, αλλά ως 0.1 και 0.3. Αυτό συμβαίνει εξαιτίας της εντολής fir1(n, Wn). Έπειτα βρίσκουμε τη διαφορά των δύο γωνιακών συχνοτήτων (Rf_diff) και την συχνότητα αποκοπής (cutooff) του ιδανικού φίλτρου που προκύπτει από τα δεδομένα του προβλήματος. Έτσι αφού το ζητούμενο

παράθυρο είναι Hamming τότε χρησιμοποιώντας τον τύπο Δω (Rf_diff) = $\frac{8\pi}{M}$ βρίσκουμε το M. Για το παράθυρο θα ισχύει ότι θα ορίζεται στο διάστημα $0 \leq n \leq M$ με τον τύπο του κι εκτός αυτού θα είναι 0 (Θα δούμε στην περιγραφή της εντολής fir1 που μας χρησιμεύει αυτό).

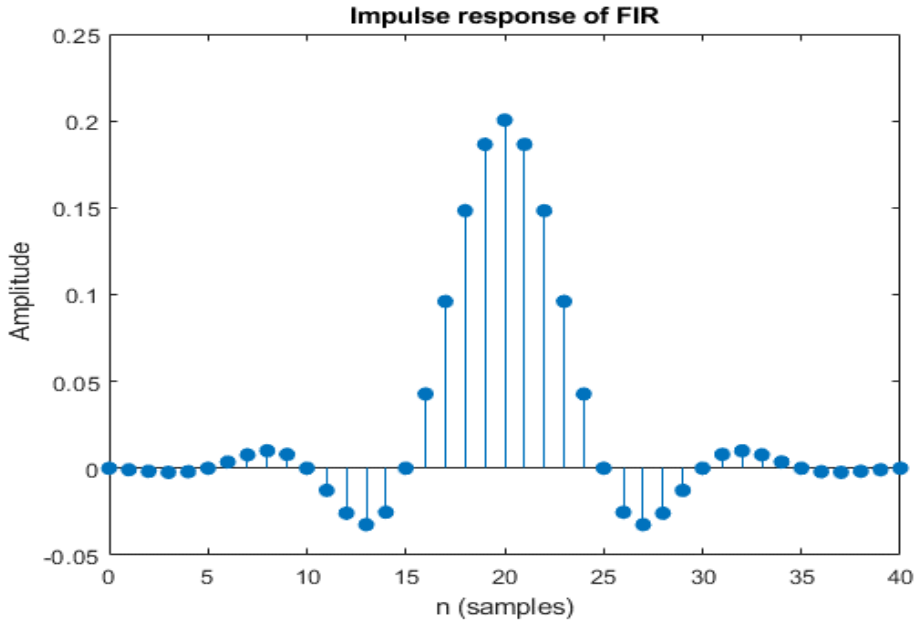
Συναρτήσεις (για περισσότερες πληροφορίες help function στο command window του MATLAB):

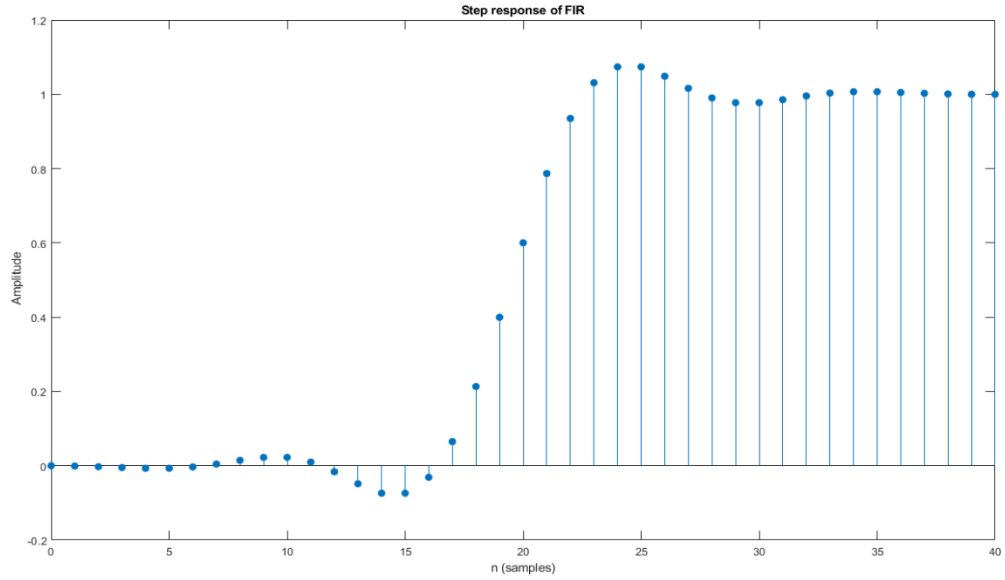
- Filter = fir1(n, Wn): Χρησιμοποιεί ένα παράθυρο Hamming ένα ψηφιακό FIR φίλτρο σε μορφή διανύσματος. Το φίλτρο είναι Filter(z) = Filter(1) + Filter(2)z^{-1} + … + Filter(n + 1)z^{-n} (Σχέση 1). Με αυτή τη σύνταξη η έξοδος είναι ένα lowpass FIR με cutoff frequency ω = Wn π (για αυτό δεν πολλαπλασιάζονται με π οι δύο παραπάνω συχνότητες) με χρήση ενός Hamming μήκους n + 1 (για αυτό το λόγο χρειαζόμαστε το Rf_diff).
- impz(b, a): Σχεδιασμός της κρουστικής απόκρισης του ψηφιακού φίλτρου με ορίσματα αριθμητή του διανύσματος b και αριθμητή του a (με τον τρόπο που περιγράφηκε στη σχέση 1)
- stepz(b, a): Σχεδιασμός της βηματικής απόκρισης του του ψηφιακού φίλτρου με ορίσματα αριθμητή του διανύσματος b και αριθμητή a.
- freqz(b, a, n): Σχεδιάζει την απόκριση συχνότητας για το ψηφιακό φίλτρο με συνάρτηση μεταφοράς με ορίσματα αριθμητή του διανύσματος b και αριθμητή a.
- grpdelay(b, a, n): Σχεδιασμός της καθυστέρησης ομάδας του φίλτρου με ορίσματα αριθμητή του διανύσματος b και αριθμητή του a.

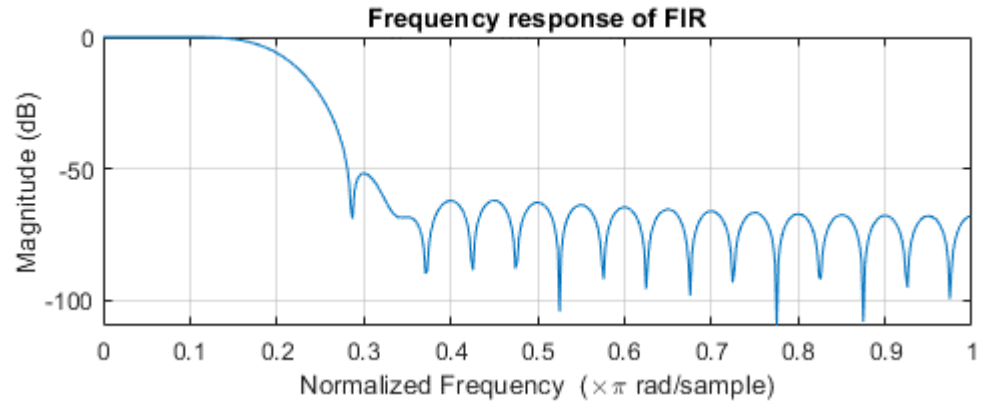
Για τις ανάγκες της εργασίας χρησιμοποιήθηκαν n = 512 δείγματα.

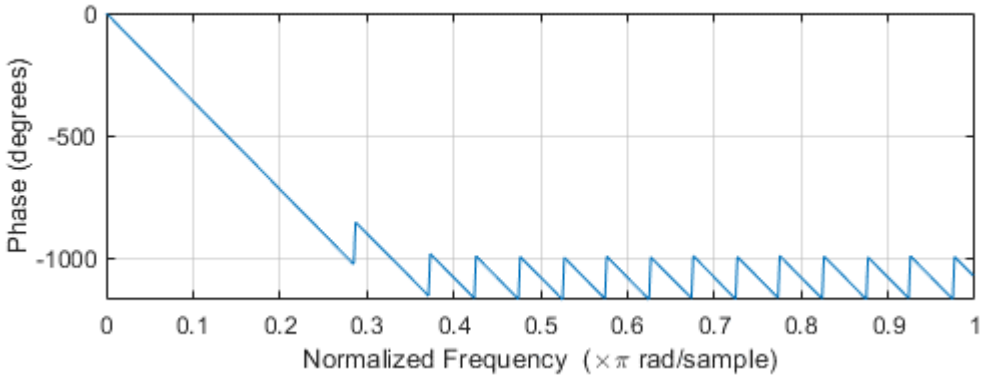
Σημείωση: Στις 2 τελευταίες περιπτώσεις το n είναι ο αριθμός των σημείων επαλήθευσης. Για καλύτερα αποτελέσματα πρέπει να είναι μεγαλύτερος από την τάξη του φίλτρου.

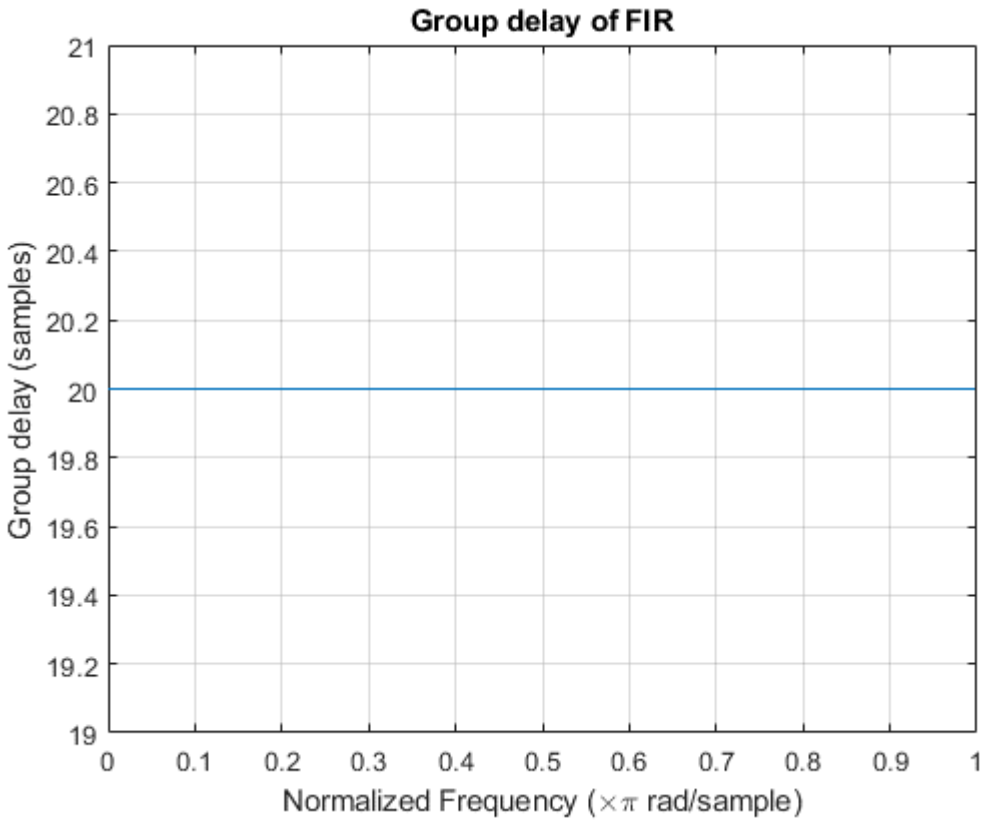
<u>Παραγόμενα διαγράμματα (από MATLAB):</u>




Step response of FIR
Amplitude
n (samples)
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
0
5
10
15
20
25
30
35
40


Frequency response of FIR
Magnitude (dB)
0
-50
-100
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Normalized Frequency (×π rad/sample)


Phase (degrees)
0
-500
-1000
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Normalized Frequency (×π rad/sample)



Σχεδίαση ως IIR με Butterworth και bilinear transform:

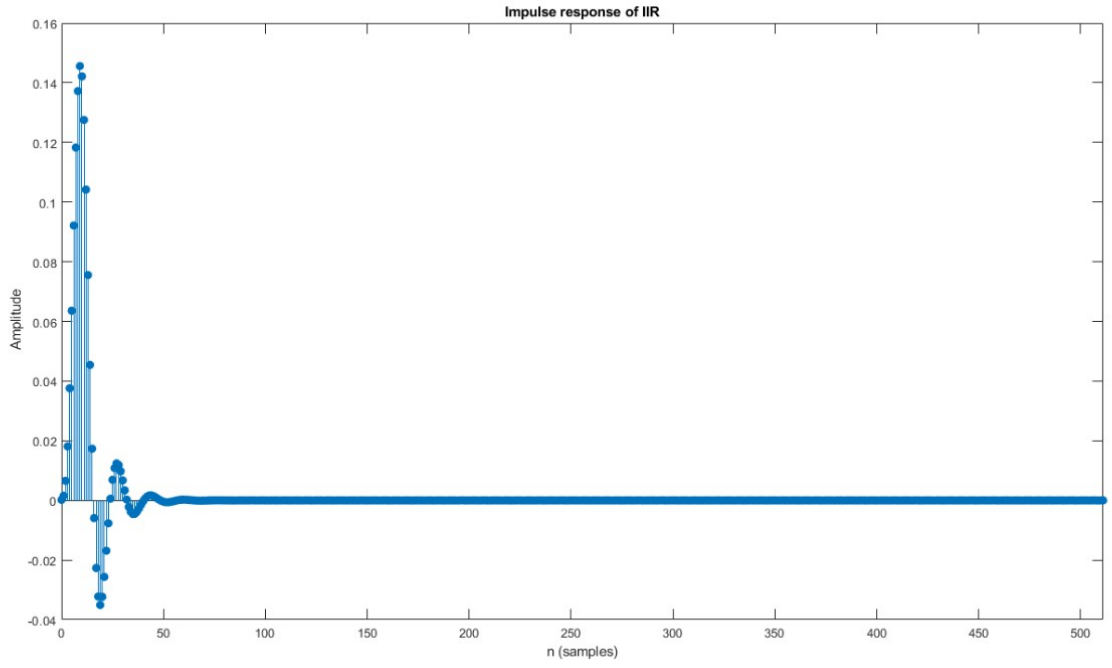
Δίνεται ο κώδικας της σχεδίασης.

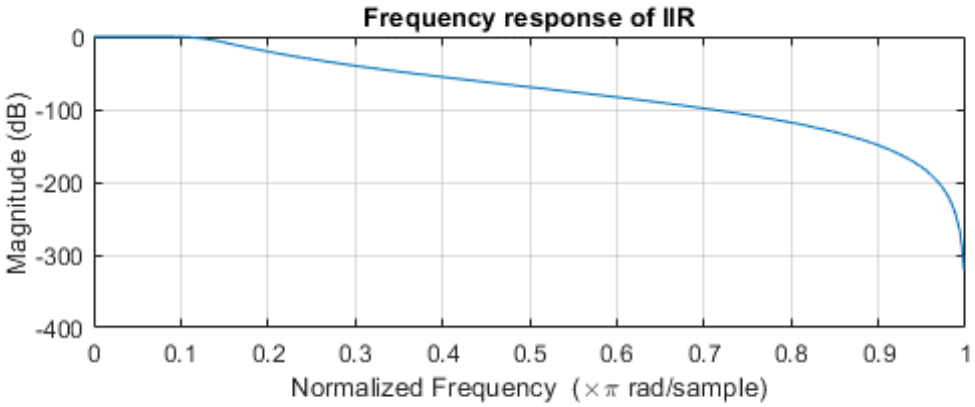
```
function [b, a] = IIR()
Max_ripple_pass = 1; % (dB)
passband_frq = 0.1; % *pi

Max_ripple_stop = 40; % dB
stopband_frq = 0.3; % *pi


[order, cutoff] = buttord(passband_frq, stopband_frq, Max_ripple_pass, Max_ripple_stop);

%% b arithmhths kai a paronomastis

[b, a] = butter(order, cutoff); %% Bilinear logw tou algorithmou tou

figure(1); impz(b, a, 512); title('Impulse response of IIR');
figure(2); stepz(b, a, 512); title('Step response of IIR');
figure(3); freqz(b,a, 512); title('Frequency response of IIR');
figure(4); grpdelay(b, a, 512); title('Group delay of IIR');
figure(5); zplane(b, a); title('Poles and zeros of IIR');
```

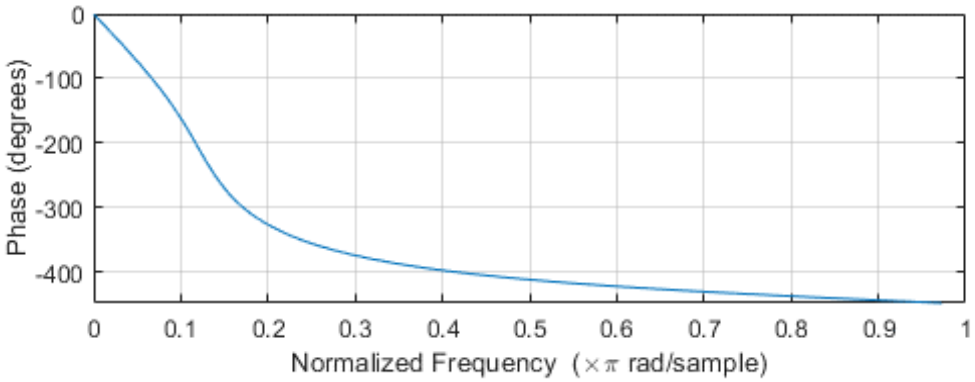
Επιπλέον συναρτήσεις:

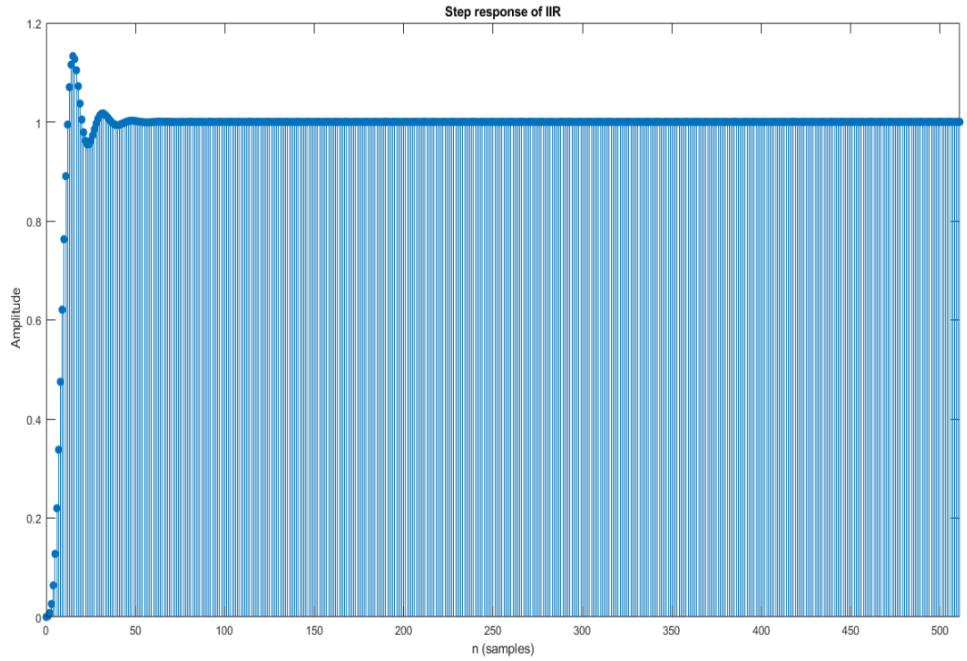
- [n, Wn] = buttord(Wp, Ws, Rp, Rs): Επιστρέφει τη χαμηλότερη τάξη n και τη συχνότητα αποκοπής του Butterworth φίλτρου με μέγιστο κυματισμό στη ζώνη διάβασης σε dB ορισμένο από Rp και απόσβεση τουλάχιστον Rs. Τα Wp, Ws ορίζουν τις συχνότητες διάβασης και αποκοπής αντίστοιχα.
- [b, a] = butter(n, Wn): Επιστρέφει τον αριθμητή και τον παρονομαστή b, a ενός Butterworth τάξης n και συχνότητας αποκοπής Wn. Χρησιμοποιεί στον αλγόριθμο του διγραμμικό μετασχηματισμό.
- zplane(b, a): Αφού βρει τις ρίζες των διανυσμάτων b, a σχεδιάζει το διάγραμμα πόλων και μηδενικών της συνάρτησης μεταφοράς που ορίζεται από τα b, a θεωρώντας τα διανύσματα ως αριθμητή και παρονομαστή όπως περιεγράφηκε στην fir1.

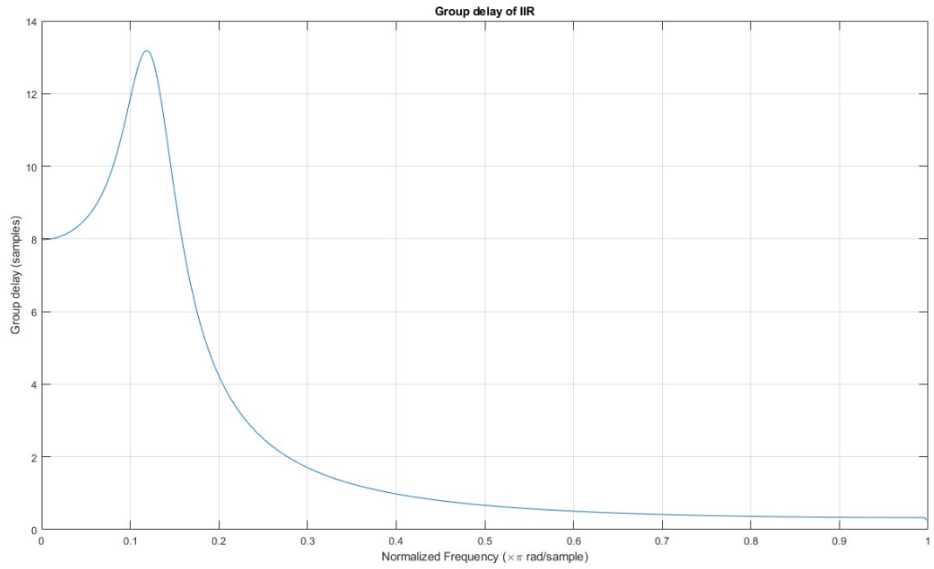
<u>Παραγόμενα διαγράμματα (από MATLAB):</u>

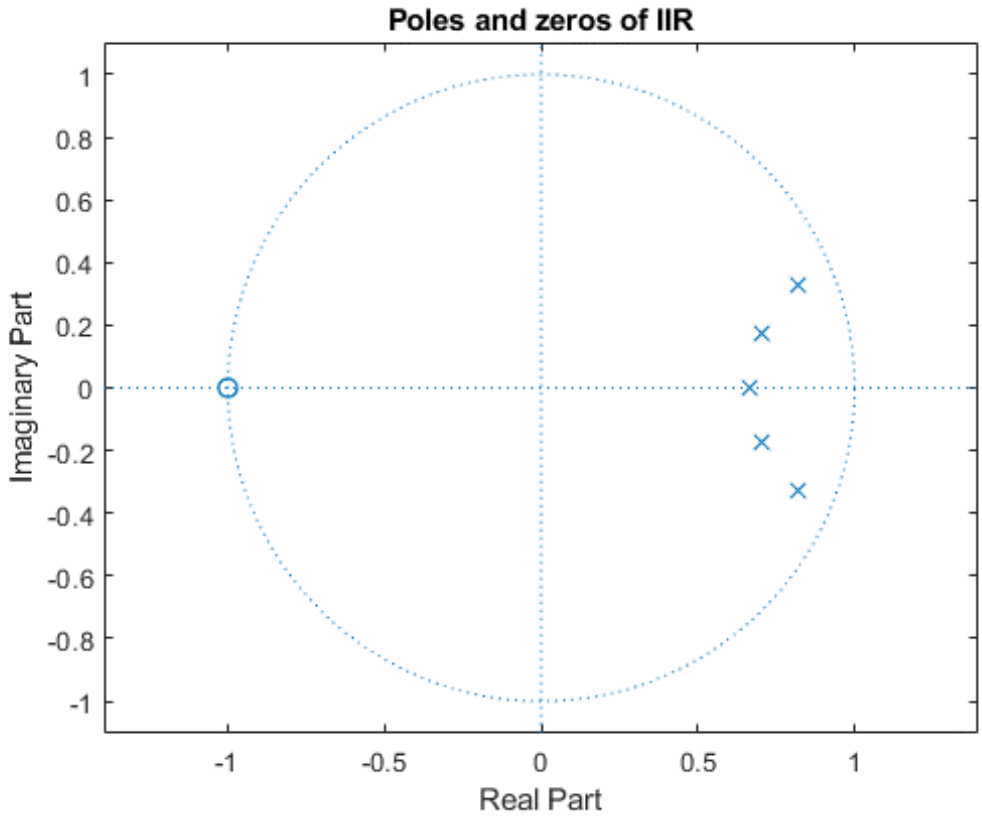



Frequency response of IIR
Magnitude (dB)
0
-100
-200
-300
-400
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Normalized Frequency (×π rad/sample)


Phase (degrees)
0
-100
-200
-300
-400
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Normalized Frequency (×π rad/sample)


Step response of IIR
Amplitude
1.2
1
0.8
0.6
0.4
0.2
0
0
50
100
150
200
250
300
350
400
450
500
n (samples)


Group delay of IIR
Group delay (samples)
Normalized Frequency (×π rad/sample)


Poles and zeros of IIR
Imaginary Part
Real Part

ΜΕΡΟΣ Β: ΣΗΜΑΤΑ ΜΕ DFT

B1.1:

Αρχικά θα εξεταστούν οι περιπτώσεις όπου N = L. Για το σήμα της x[n] = $A_1\cos(\omega_1 n)$ + $A_2\cos(\omega_2 n)$, με $A_1 = 1$ και $A_2 = 0.50$ και $\omega_1 = \pi \bmod((10/7.5) * (\max(A) / (l_1 + l_2)), 1)$ και $\omega_2 = \bmod(\omega_1 + \pi / 4, \pi)$, το οποίο παραθυρώνεται με ορθογώνιο παράθυρο για τα μήκη με L = 16, 64, 512 ζητείται ο DFT για μήκη N = L, 2^{64}.

Δίνεται ο κώδικας της εύρεσης του DFT και του σχεδιασμού του μέτρου του DFT (όλα τα διαγράμματα σχεδιάστηκαν σε dB). Επίσης ο σχεδιασμός έγινε με συνεχή γραμμή για καλύτερη κατανόηση των αποτελεσμάτων, καθώς το μέτρο αποτελεί δειγματοληψία σημείων του διαγράμματος ανά $\frac{\pi}{N-1}$ (ουσιαστικά σχεδιάστηκε το μέτρο του DTFT και το μέτρο του DFT μαζί σε dB). Με μπλέ συνεχή γραμμή δίνεται το μέτρο του DTFT και με κόκκινα σημεία το μέτρο του DFT σε decibels :

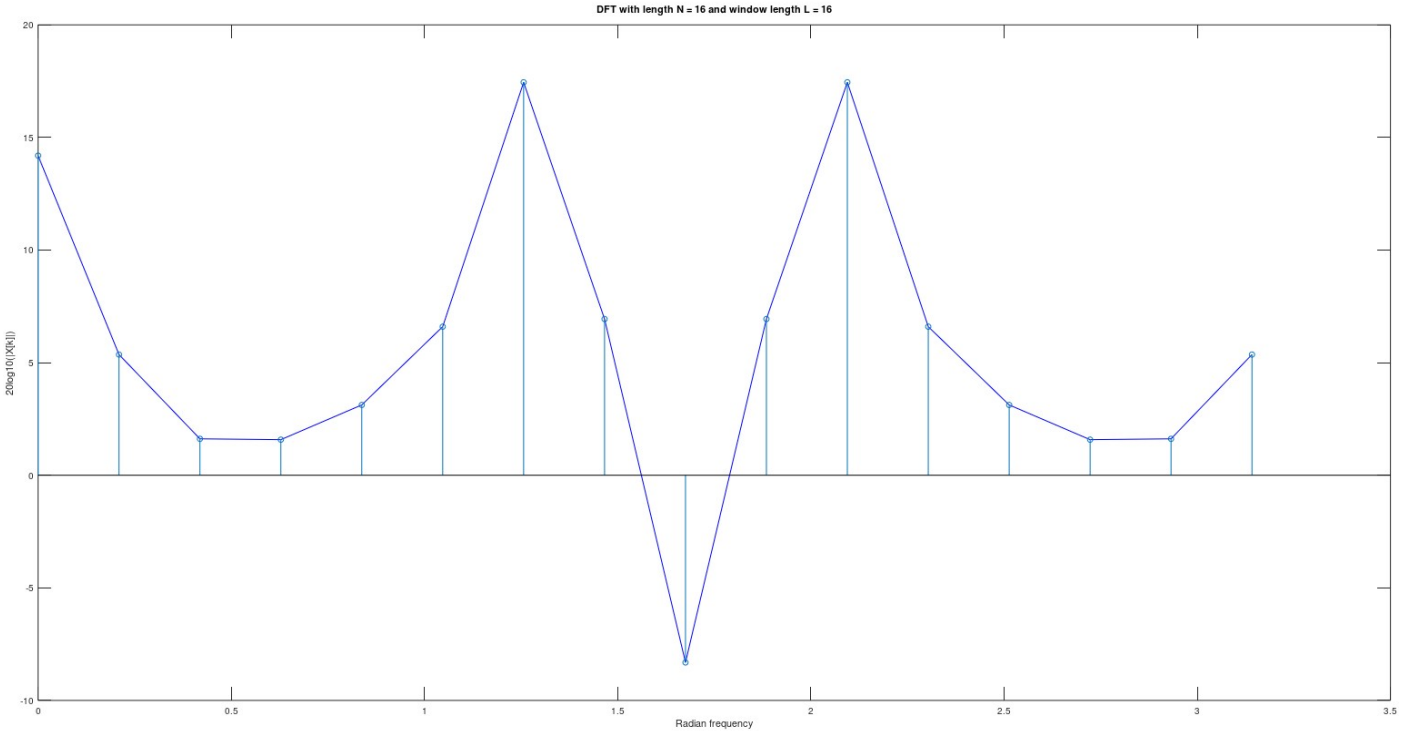
```
%% Latiniko onoma: Dimitrios (9 grammata)
%% Latiniko epitheto: Koukougiannis (13)

function signalB = DFT()
  %% Orismos parametrwn

L = 16; %% Allazei
A1 = 1; A2 = 0.5;
l1 = 9; l2 = 13;

A = [l1 l2]; %% Pinakas gia ton arithmo twn grammatwn
frq1 = pi*mod((10/7.5) * (max(A) / (l1 + l2)), 1);
frq2 = mod(frq1 + (pi / 4), pi);

N = L; %% allazei

signalA = inline('A1*cos(frq1*n) + A2*cos(frq2*n)'); %% x[n]
xconf = linspace(0, L - 1, L);
signalB = signalA(A1, A2, frq1, frq2, xconf).*rectwin(L)'; %% Parathiromeno shma L stoixeia


%% .* Shmainei oti pollaplasiazoume stoixeio me stoixeio(x[1]*x[1] x[2]*x[2] etc])
fouriert = fft(signalB, N);
xlin = linspace(0, pi, N);
plot(xlin, 20*log10(abs(fouriert)), 'b-') ;hold on;
stem(xlin, 20*log10(abs(fouriert)));xlabel('Radian frequency'); ylabel('20log10(|X[k]|)');
title(['DFT with length N = ', num2str(N), ' and window length L = ', num2str(L)]);
```

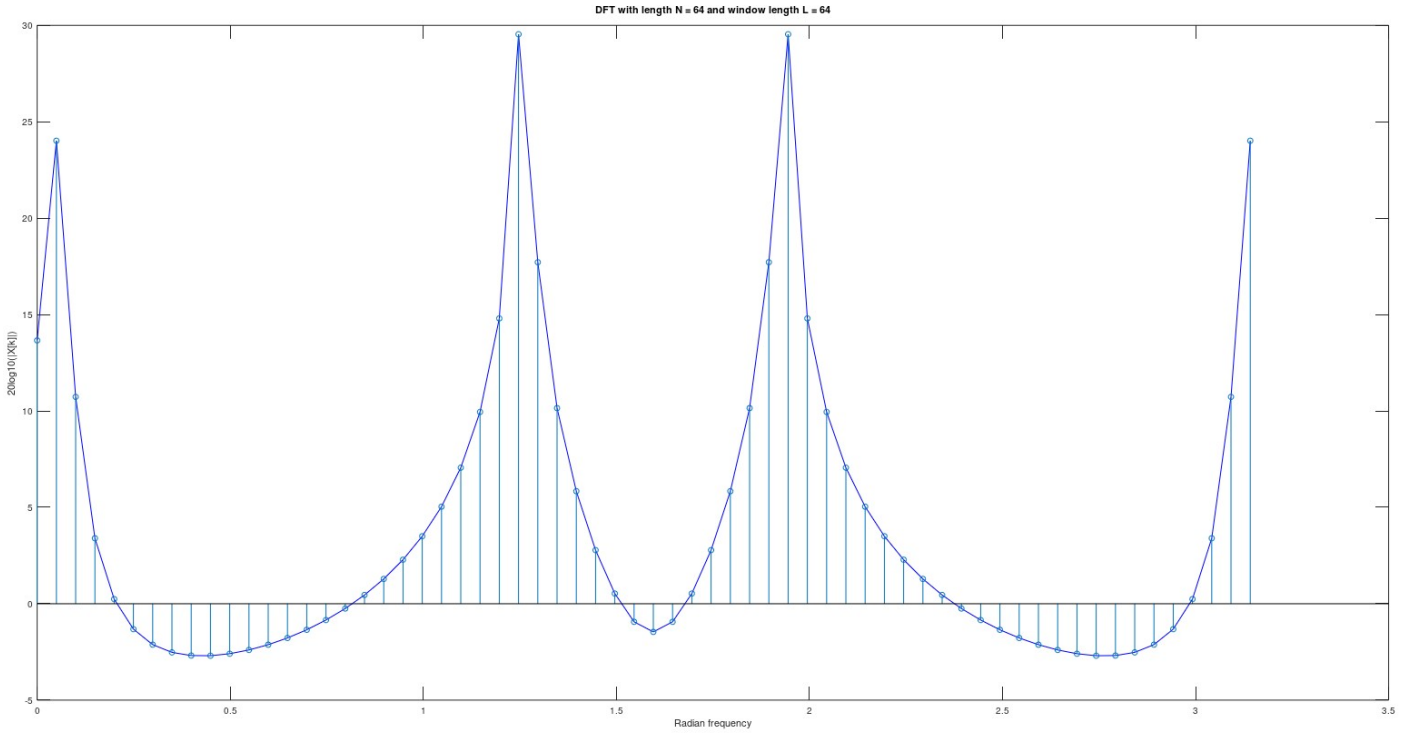
Συναρτήσεις:

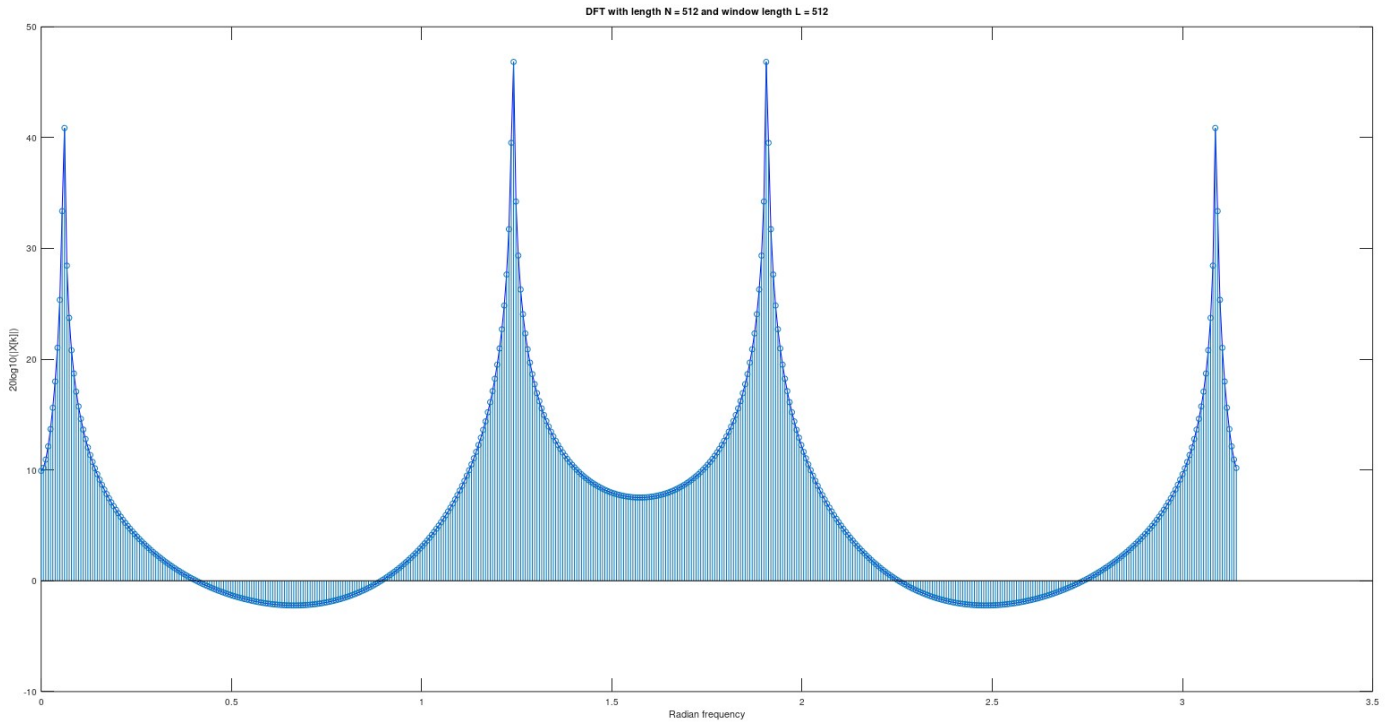
- inline(string): Χρησιμεύει για ορισμό του σήματος x[n] (ως inline object)
- rectwin(L): Επιστροφή ενός ορθογώνιο παράθυρο μήκους L ($0 \leq n \leq L - 1$)
- Y = fft(X, n): Χρησιμοποιώντας έναν αλγόριθμο Fast Fourier Transform επιστρέφει στο Y τον DFT μήκους n του διανύσματος X και τον επιστρέφει στο Y. Αν length(X) < n τότε συμπληρώνει με μηδενικά (χρησιμεύει για την περίπτωση $L = 2^{14}$).

<u>Σημείωση: Οι τιμές για το μέτρο του DFT κυμαίνονται από 0 έως N – 1.</u>

<u>Παραγόμενα διαγράμματα (από Octave):</u>





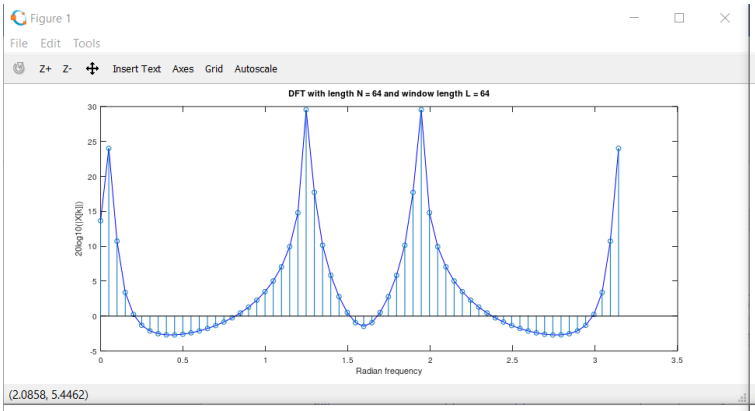


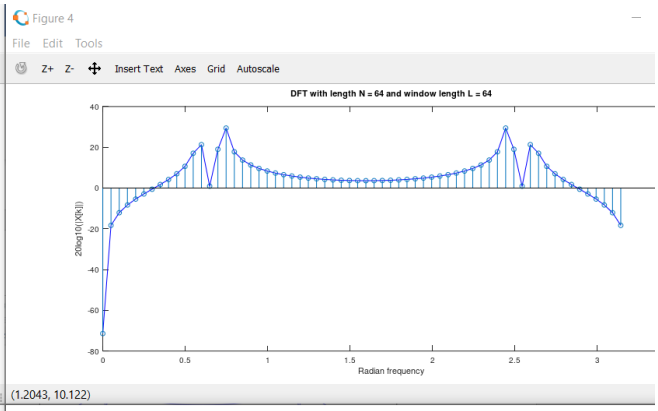
Στα διαγράμματα φαίνεται ότι αυξάνοντας το μήκος του παραθύρου (κατά συνέπεια και το N, αφού εξετάζουμε την περίπτωση N = L), το σήμα γίνεται πιο smooth (τα δείγματα έρχονται πιο κοντά). Κοιτώντας τα τρία διαγράμματα παρατηρώ ότι οι γραμμές του διαγράμματος για L = 16 λειαίνονται σταδιακά. Επίσης οι κορυφές (τοπικά μέγιστα) φτάνουν πιο ψηλά. Παράλληλα τα κοίλα του διαγράμματος φαίνονται ότι αυξάνονται σε μήκος και μετατοπίζονται προς τα κάτω από 16 σε 64. Η αλλαγή αυτή δε διακρίνεται και στην αλλαγή του L από 64 σε 512. Στο τελευταίο διάγραμμα τα κοίλα φαίνονται μετατοπισμένα προς τα πάνω, όπως κι οι κορυφές.

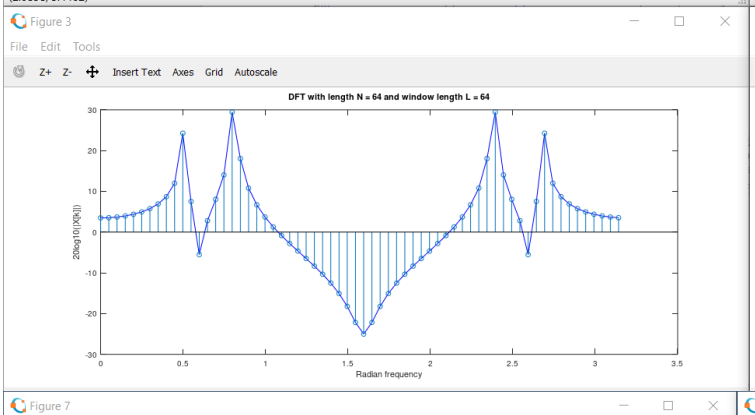
Τώρα θα επικεντρωθούμε στην περίπτωση όπου L = 64. Αλλάζοντας τον κώδικα μετά τους ορισμούς των παραμέτρων όπως φαίνεται παρακάτω:

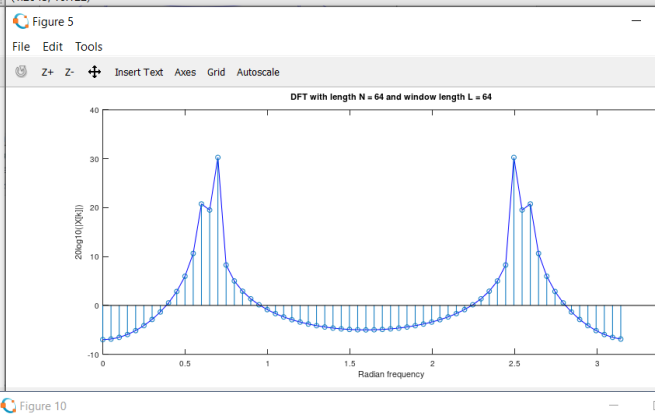
```
N = L; %% allazei

  for i = 1:1:100
  signalA = inline('A1*cos(frq1*n) + A2*cos(frq2*n)'); %% x[n]
  xconf = linspace(0, L - 1, L);
   signalB = signalA(A1, A2, frq1, frq2, xconf).*rectwin(L)'; %% Parathiromeno shma L stoixeia


  %% .* Shmainei oti pollaplasiazoume stoixeio me stoixeio(x[1]*x[1] x[2]*x[2] etc])
  fouriert = fft(signalB, N);
  xlin = linspace(0, pi, N);
  plot(xlin, 20*log10(abs(fouriert)),  'b-',  xlin,  20*log10(abs(fouriert)),'ro')

  xlabel('Radian frequency'); ylabel('20log10(|X[k]|)');
  title(['DFT with length N = ', num2str(N), ' and window length L = ', num2str(L)]);

  center = (frq1 + frq2) / 2;
  frq1 = (frq1 + center) / 2;
  frq2 = (frq2 + center) / 2;

  disp('Iterations: '); disp(i);
  pause(3);
  end
```

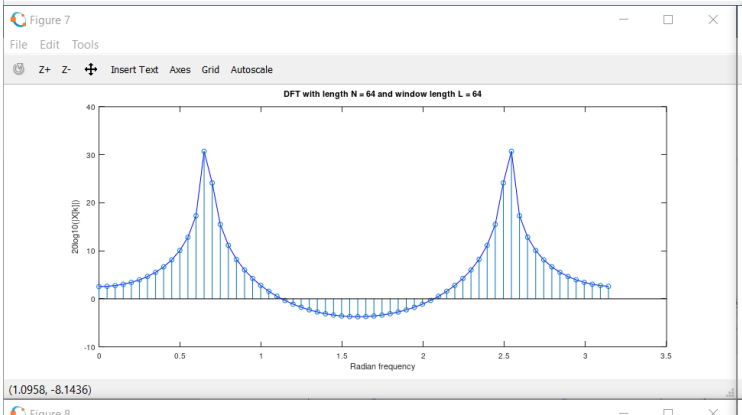
Τρέχουμε τον κώδικα και παρακολουθούμε τα αποτελέσματα. Στην επανάληψη 13 το διάγραμμα σταματάει να αλλάζει. Παρακάτω δίνονται επιλεκτικά κάποια διαγράμματα από το τρέξιμο του script. Όπου figure n προέρχεται από την κάθε επανάληψη. Το script εκτελέστηκε σε Octave.

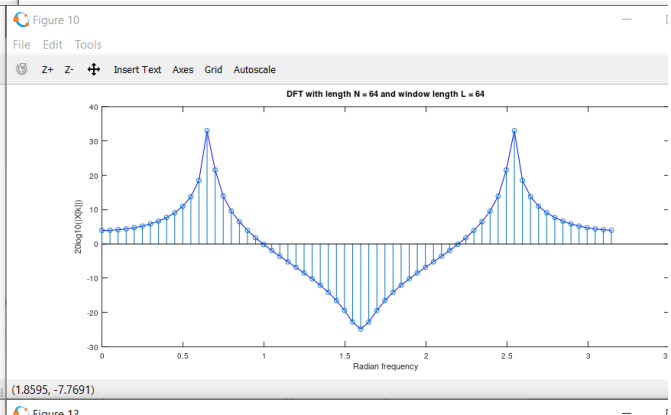


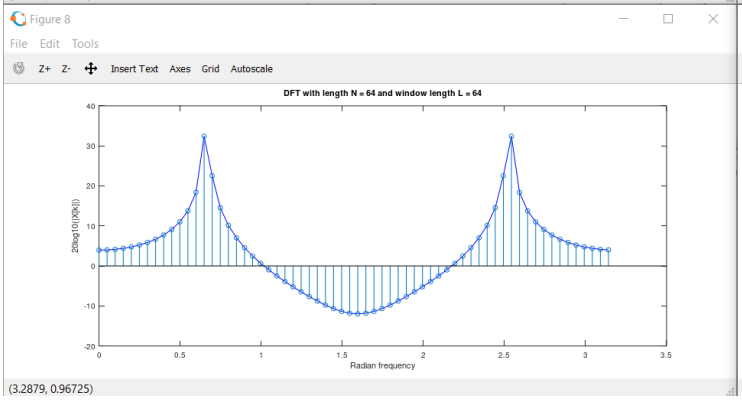


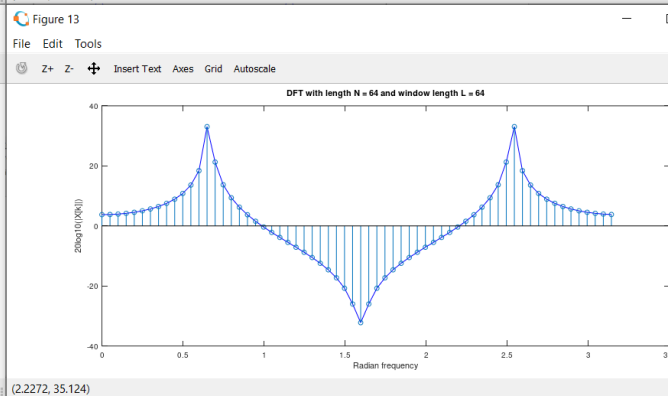






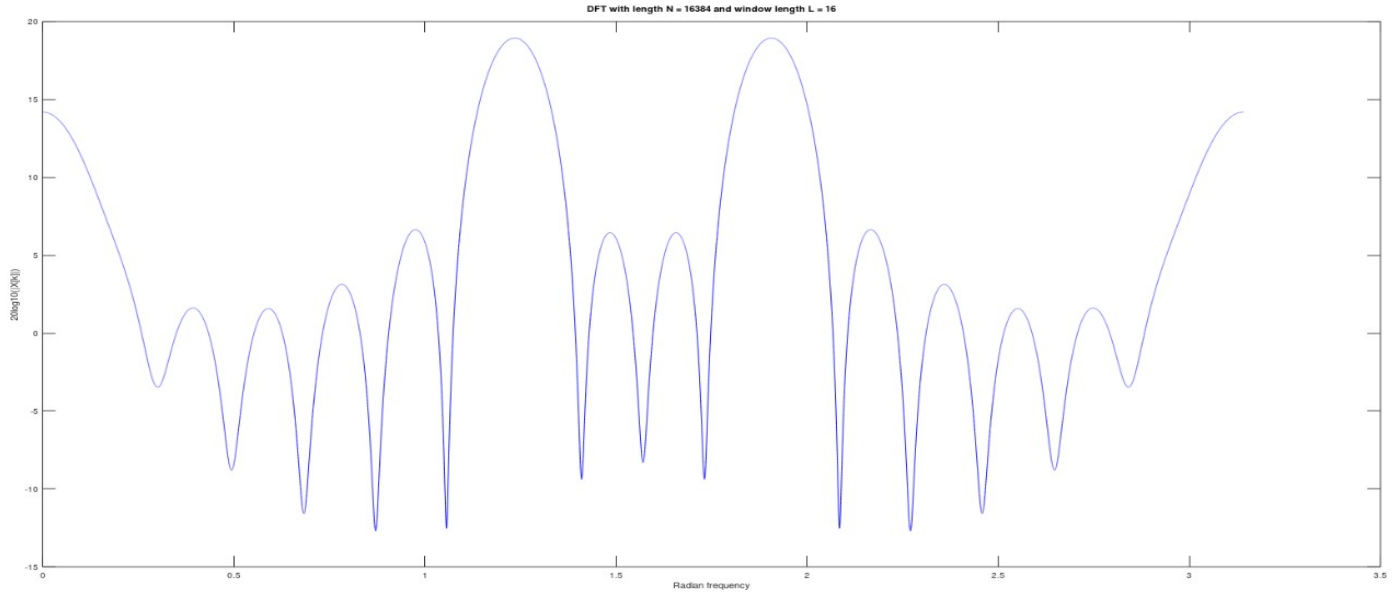


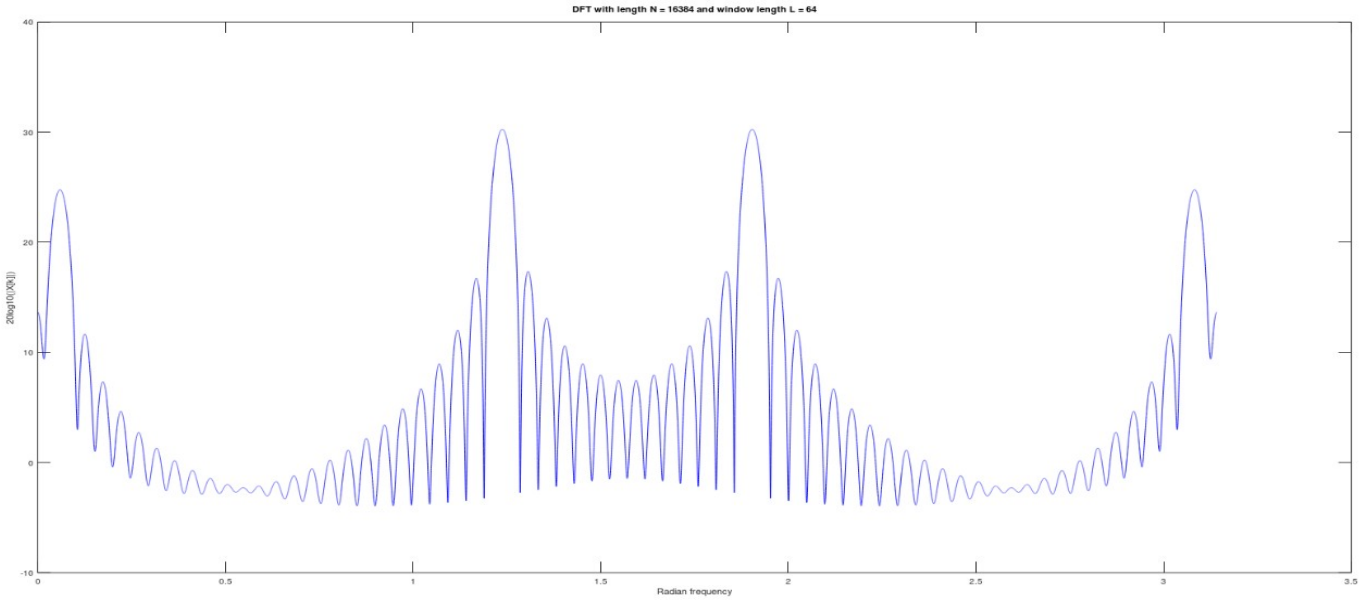


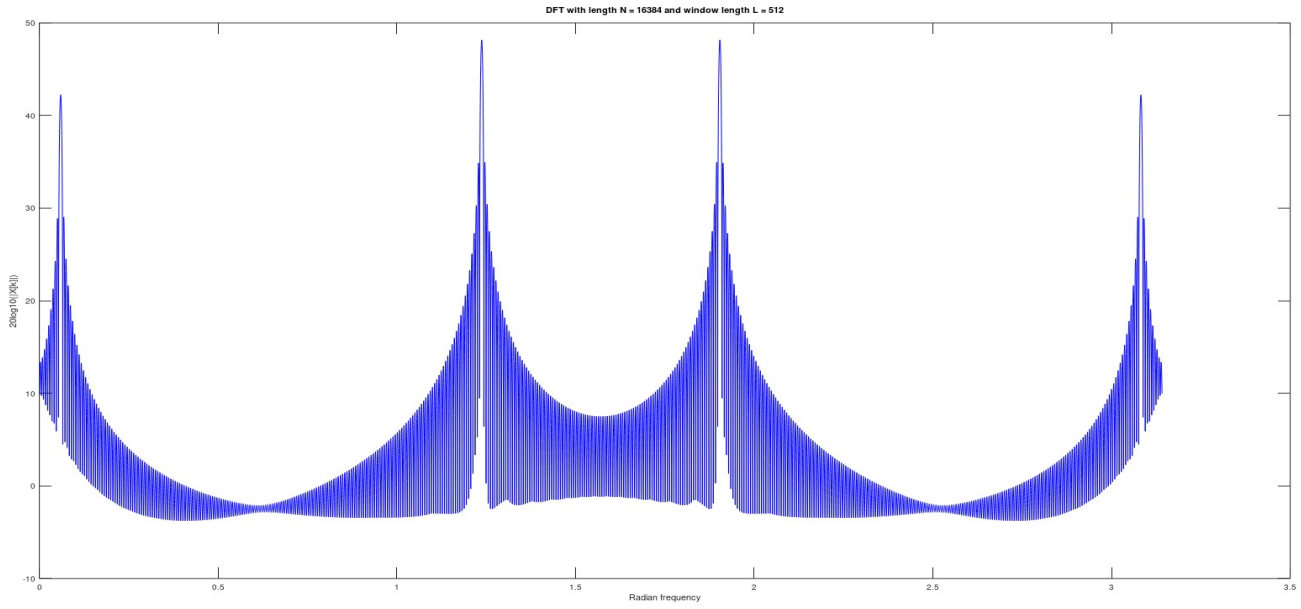


Κοιτώντας τα διαγράμματα είναι εμφανές ότι όσο πλησιάζουν τα ω το κέντρο τότε τόσο πιο πολύ <<ανοίγει>> το διάγραμμα. Δηλαδή ανοίγει στο κέντρο και αριστερότερα και δεξιότερα αυτού το διάγραμμα συμπιέζεται. Τελικά κοιτώντας το figure 13 παρατηρούμε ότι δεξιά κι αριστερά της <<κεντρικής καμπύλης>> έχει μεταβληθεί πολύ το διάγραμμα αλλά εντός αυτής δεν υπάρχει μεγάλη διαφορά. Είναι δηλαδή σα να επιμηκύνθηκε από αριστερά κι από δεξιά.

Παρακάτω δίνονται τα διαγράμματα για $\underline{N = 2^{14}}$. Εδώ σχεδιάστηκε μόνο το μέτρο του DTFT επειδή τα δείγματα βρίσκονται πολύ κοντά μεταξύ τους.







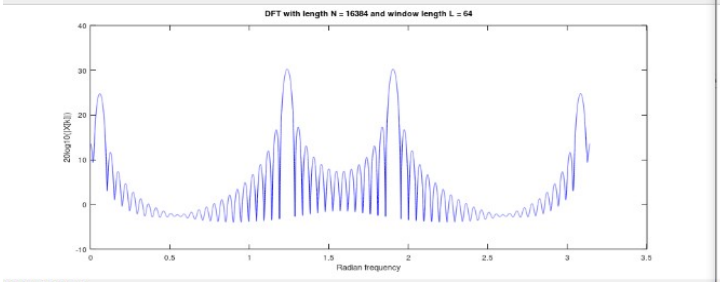
Σε αυτή την περίπτωση παρατηρούμε τα ίδια αποτελέσματα για N = L. Ωστόσο το σήμα δε <<λειαίνεται>> με την αύξηση του μήκους του παραθύρου αλλά αντίθετα οι διαδοχικές καμπύλες πλησιάζουν πολύ γρήγορα μεταξύ τους. Τελικά όταν L = 512 το σήμα γίνεται θόρυβος.

Για L = 64 ο DFT συμπεριφέρεται με τρόπο παρόμοιο όπως περιεγράφηκε στην προηγούμενη μελέτη. Το διάγραμμα συμπιέζεται αριστερότερα των δύο καμπυλών που φαίνονται στο figure 1 εντός του δε μεταβάλλεται πολύ.

Figure 1

File Edit Tools

Z+ Z- Insert Text Axes Grid Autoscale

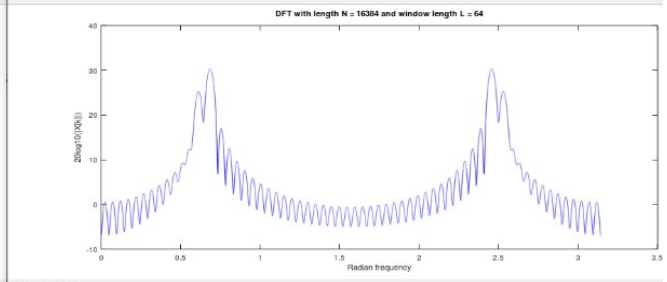


(0.21902, 8.5882)

Figure 5

File Edit Tools

Z+ Z- Insert Text Axes Grid Autoscale

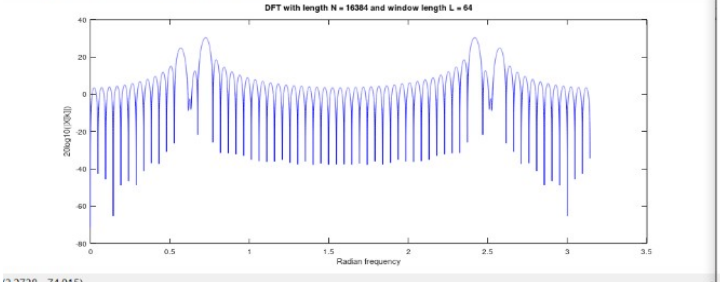


(2.9391, -5.594)

Figure 4

File Edit Tools

Z+ Z- Insert Text Axes Grid Autoscale

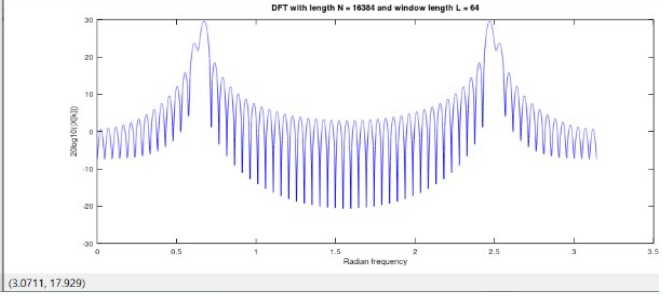


(3.2738, -74.015)

Figure 6

File Edit Tools

Z+ Z- Insert Text Axes Grid Autoscale

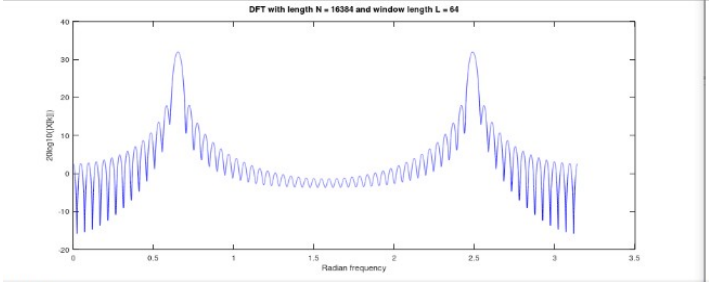


(3.0711, 17.929)

Figure 7

Edit Tools

Z+ Z- Insert Text Axes Grid Autoscale

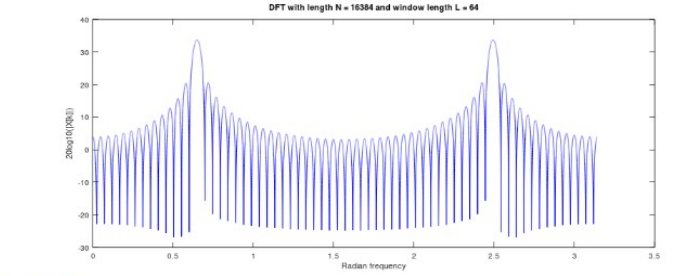


986, -18.728)

Figure 10

File Edit Tools

Z+ Z- Insert Text Axes Grid Autoscale

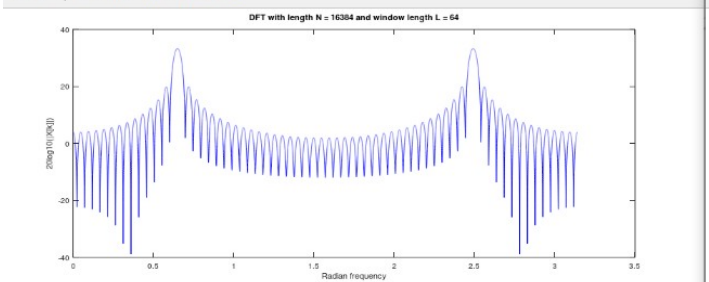


(3.2219, -6.6537)

Figure 8

Edit Tools

Z+ Z- Insert Text Axes Grid Autoscale

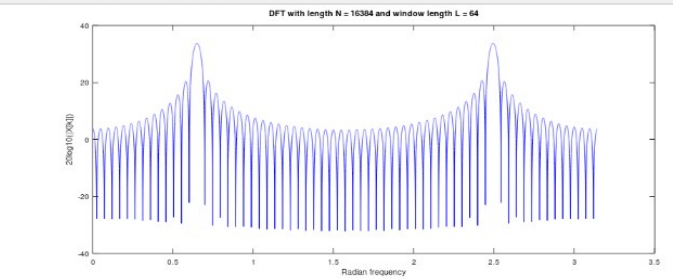


354, -21.987)

Figure 13

File Edit Tools

Z+ Z- Insert Text Axes Grid Autoscale

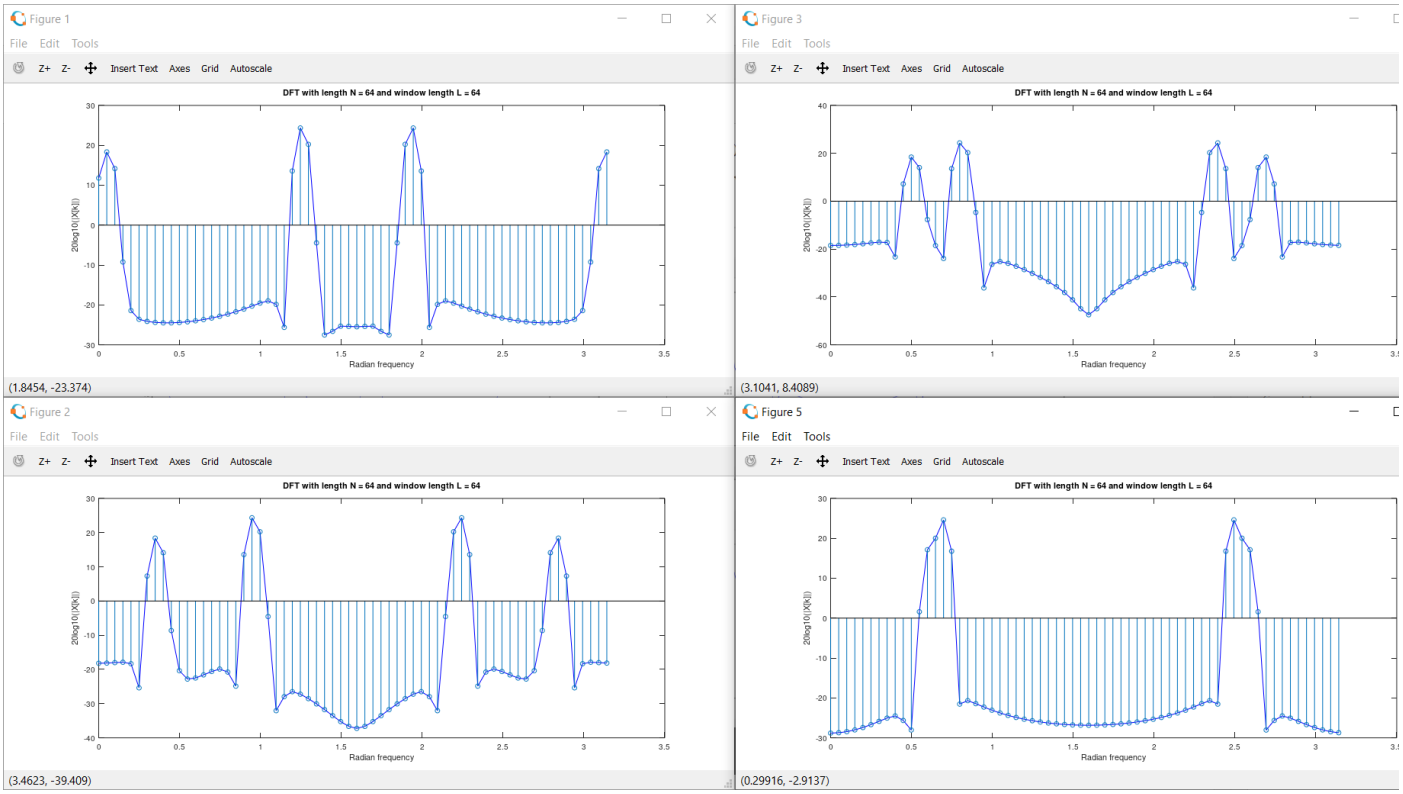


(1.1147, 34.614)

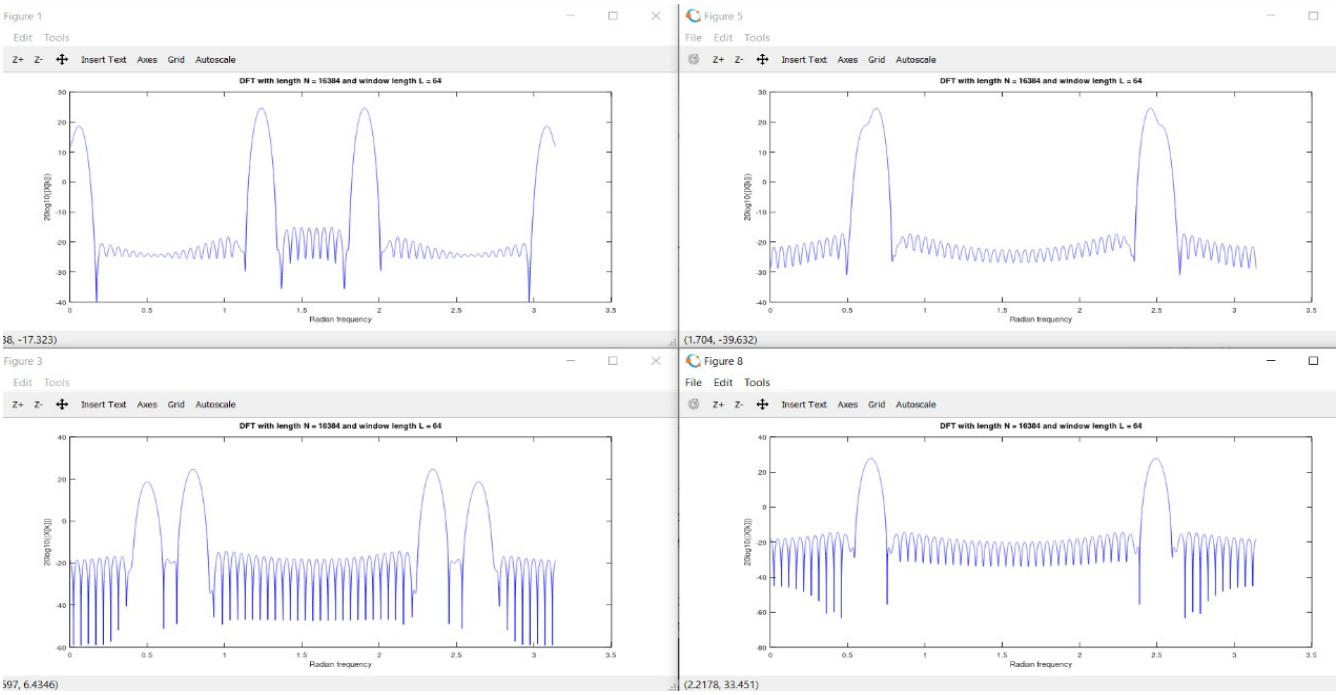
B1.2:

Απλά αλλάζουμε το rectwin σε hamming.

Κατά την αλλαγή για N = L έχουμε αρκετά μεγάλη αλλαγή στο διάγραμμα. Οι επαναλήψεις επιδρούν στο διάγραμμα με τρόπο που περιεγράφηκε πριν.



Χρησιμοποιώντας παράθυρο Hamming μήκους L = 64 για N = 2^{14} παρατηρώ ότι το διάγραμμα (figure1) μοιάζει με αυτό του B.1.1, με τη διαφορά ότι είναι λίγο πιο απλωμένο. Τα αποτελέσματα των επαναλήψεων μεταβάλλουν με τρόπο παρόμοιο με πριν το διάγραμμα.



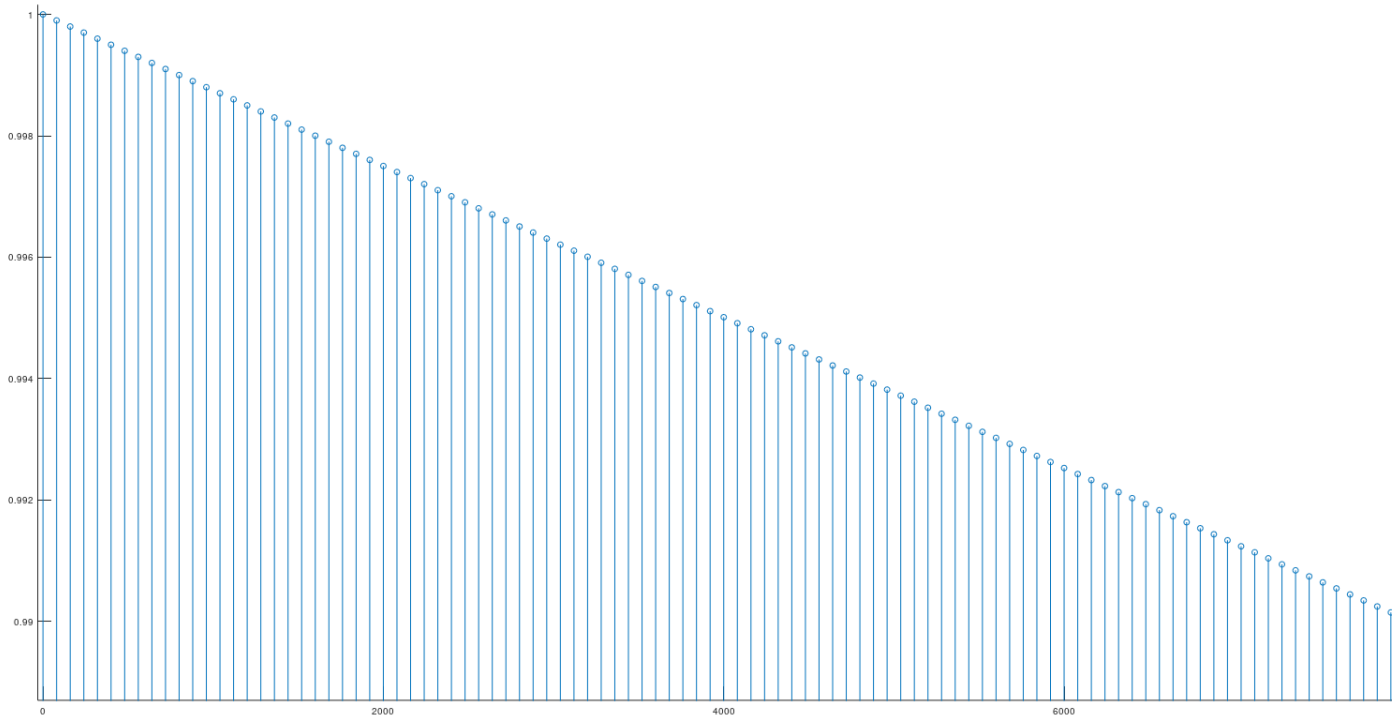
ΜΕΡΟΣ Γ: ΔΗΜΙΟΥΡΓΙΑ ΣΥΝΘΕΤΙΚΗΣ ΦΩΝΗΣ

Παρακάτω δίνεται ο κώδικας που χρησιμοποιήθηκε για την παραγωγή των διαγραμμάτων. Για τις ανάγκες τις εφαρμογής το σήμα p[n] σχεδιάστηκε μεταξύ 0 και 7999 (8000 μήκος), το φάσμα $P(e^{j\omega})$ μεταξύ 0 και π και προκύπτει από τον DFT μήκους 8000, ο οποίος αποτελεί δειγματοληψία του P.

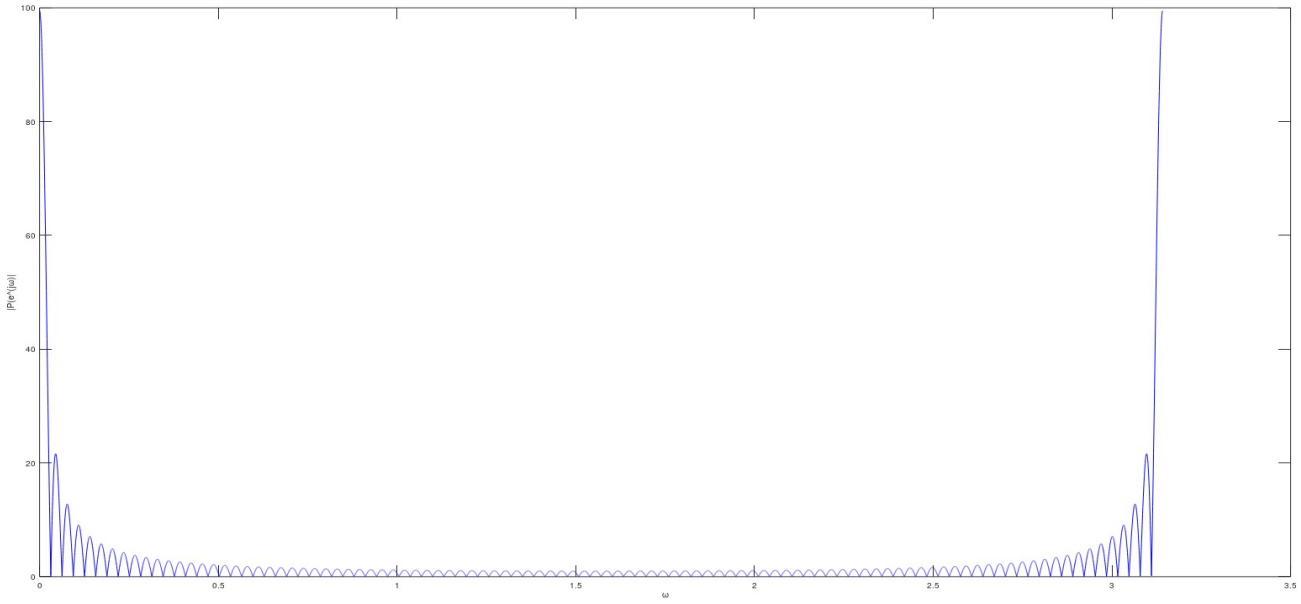
```
%%Synthetiko fonien

function plotting = fonien()

  pn = inline('0.9999^k');

  for i = 1:1:100
    plotting(1, i) = pn(i - 1);
  endfor

  figure(1);
  xplot = [0:80:7999];
  stem(xplot, plotting);

  dft = fft(plotting, 8000);
  xlin = linspace(0, pi, 8000);

  %% DFT apotelei deigmatolipsia DTFT omega = (2pi / N)*k 0 <= k <= N - 1

  figure(2);
  plot(xlin, abs(dft), 'b-');
  ylabel('|P(e\^(jω)|');
  xlabel('ω');
```

```
figure(3);
paranomastis(1, 1) = 1;
paranomastis(2:80, 1) = 0;
paranomastis(81, 1) = -0.9999;
zplane(1, paranomastis);
```

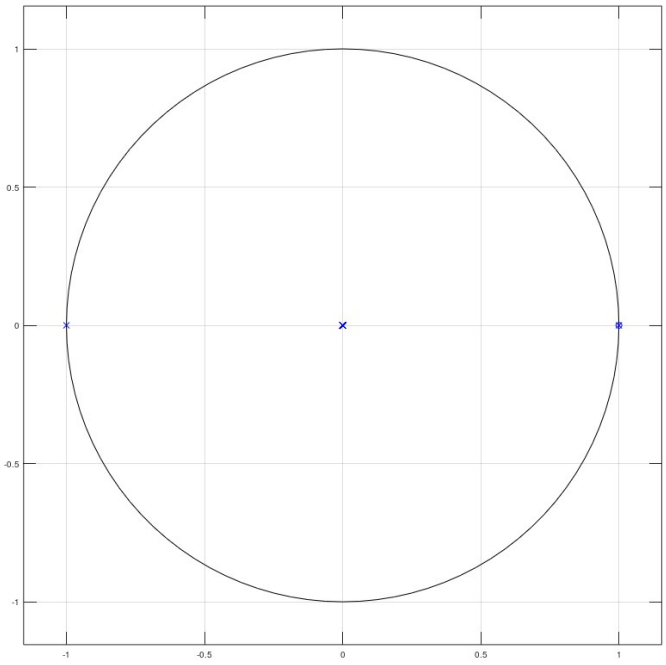
Παραγόμενα διαγράμματα (από Octave):



Εικόνα 3.1.1 – Σχεδιασμός του p[n] μεταξύ 0 και 7999



Εικόνα 3.1.2 – Μέτρο του $P(e^{j\omega})$. Προκύπτει από τον DFT μήκους 8000.



Εικόνα 3.1.3 – Διάγραμμα πόλων μηδενικών του P(z).