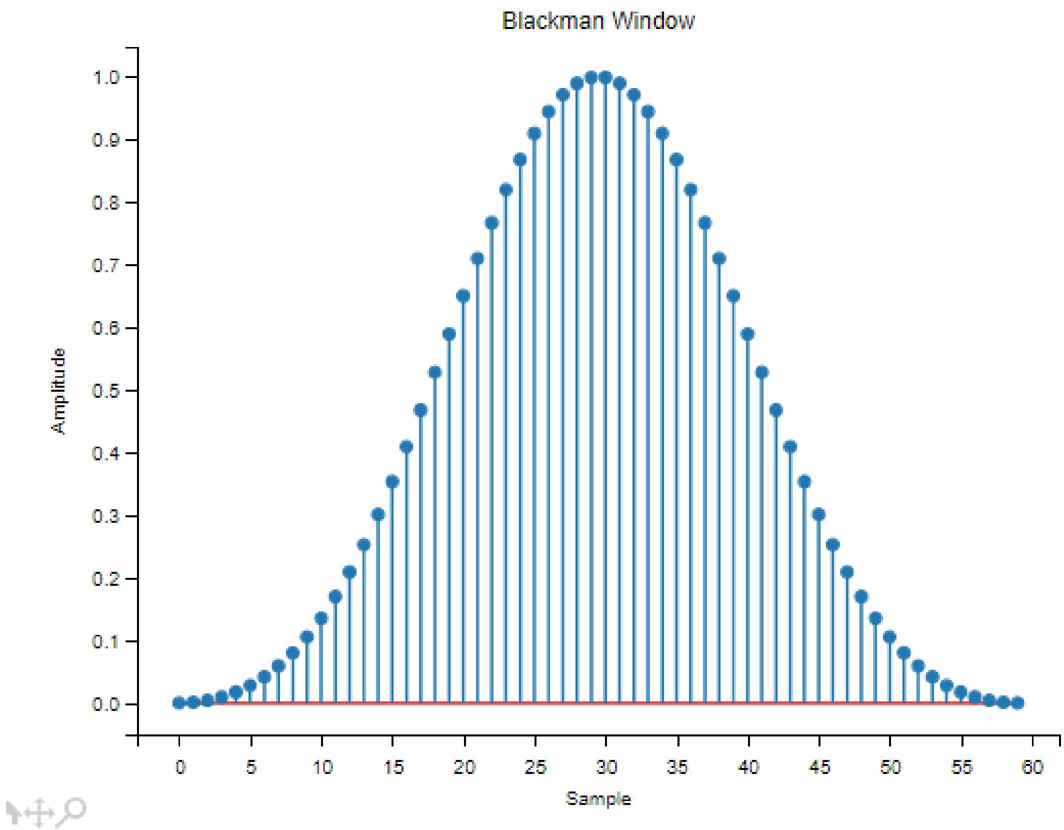
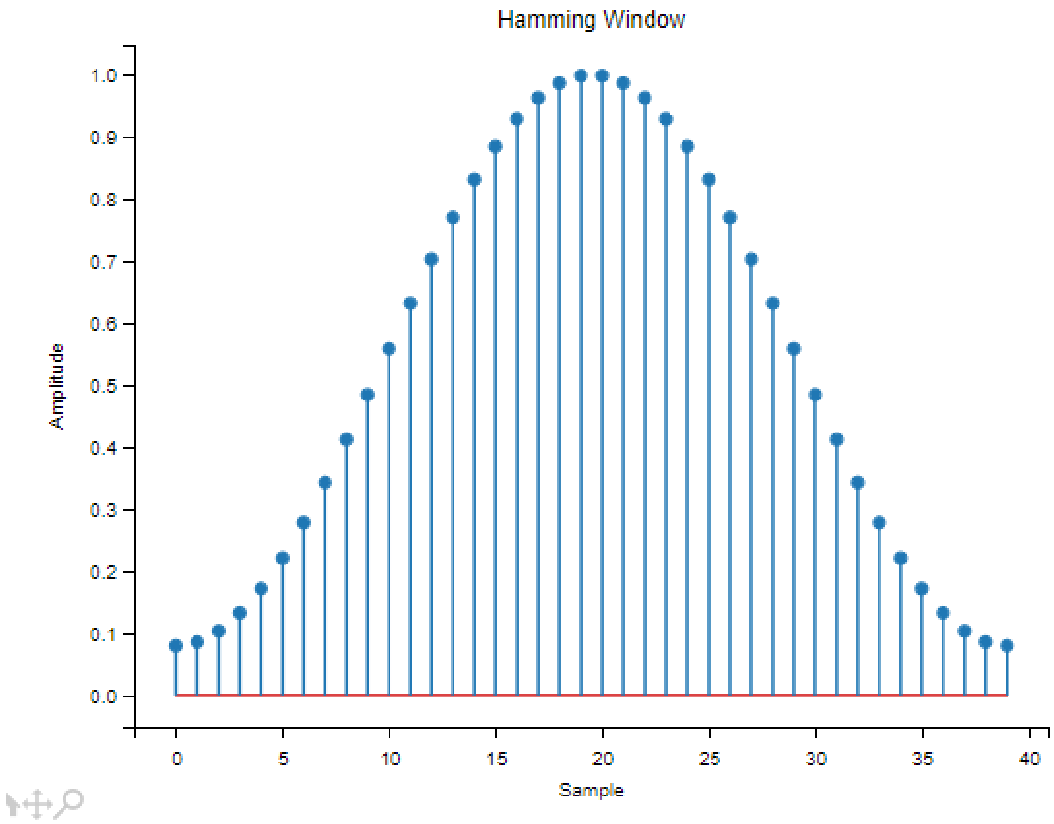
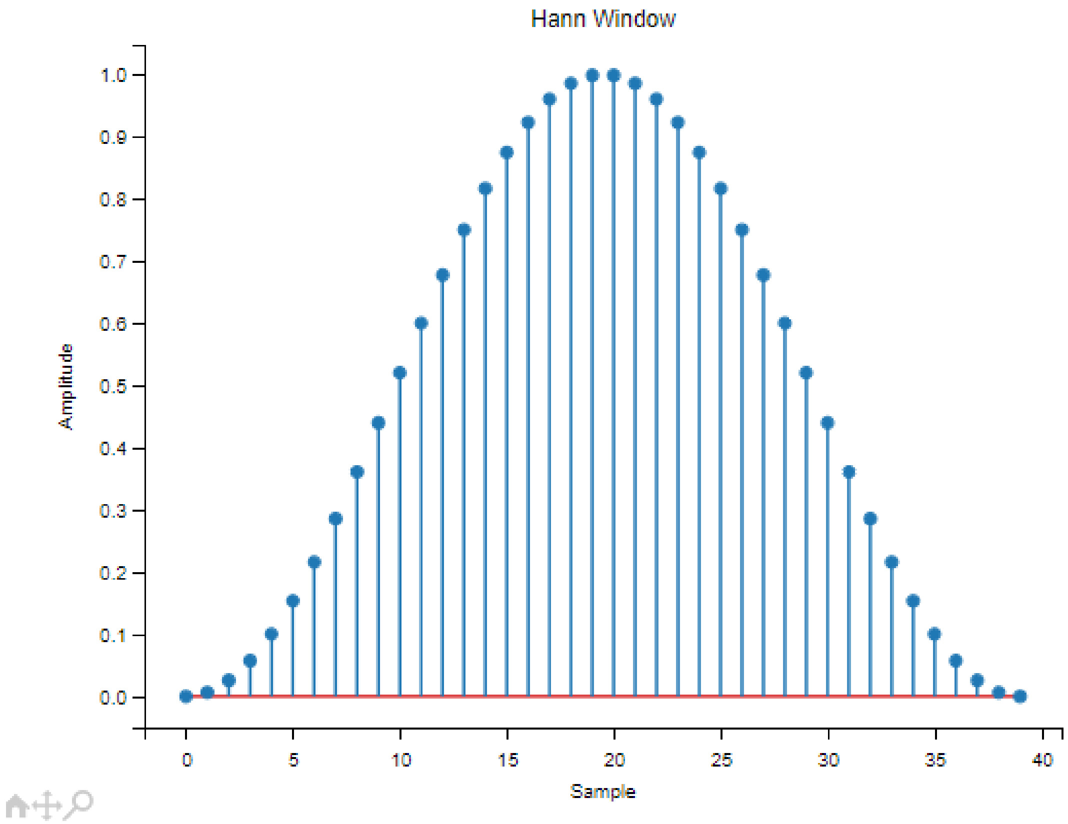
# Project ΨΕΣ Βασίλης Πίσχος(03175), Γιώργος Βαλσάμης(03259)

## Άσκηση 1η

A1) Μας ζητείται να σχεδιάσουμε ένα φίλτρο διακριτού χρόνου χρησιμοποιώντας 3 διαφορετικά παράθυρα (παράθυρο Hann, Blackman και Hamming) με passband [0.4π, 0.6π] και stopband [0, 0.2π] U [0.8π, π]. Ας αναλύσουμε την πορεία που ακολουθήσαμε με σκοπό την επίλυση του προβλήματος. Αρχικά δημιουργήσαμε δύο πίνακες ώστε να κρατάνε τις τιμές για το passband και το stopband και ύστερα με την χρήση αυτών των πινάκων βρήκαμε τις τιμές Δω που χρειαζόμαστε για να υπολογίσουμε το μέγεθος του κάθε παραθύρου μέσω του τύπου για το Δω που έχουμε διδαχθεί και στην θεωρία για κάθε παράθυρο ξεχωριστά. Ύστερα σχεδιάσαμε το κάθε παράθυρο χρησιμοποιώντας το Matplotlib.pyplot library της python (η πρώτη άσκηση έγινε σε γλώσσα python). Προέκυψαν τα 3 παρακάτω διαγράμματα.




Hamming Window
Amplitude
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
5
10
15
20
25
30
35
40
Sample

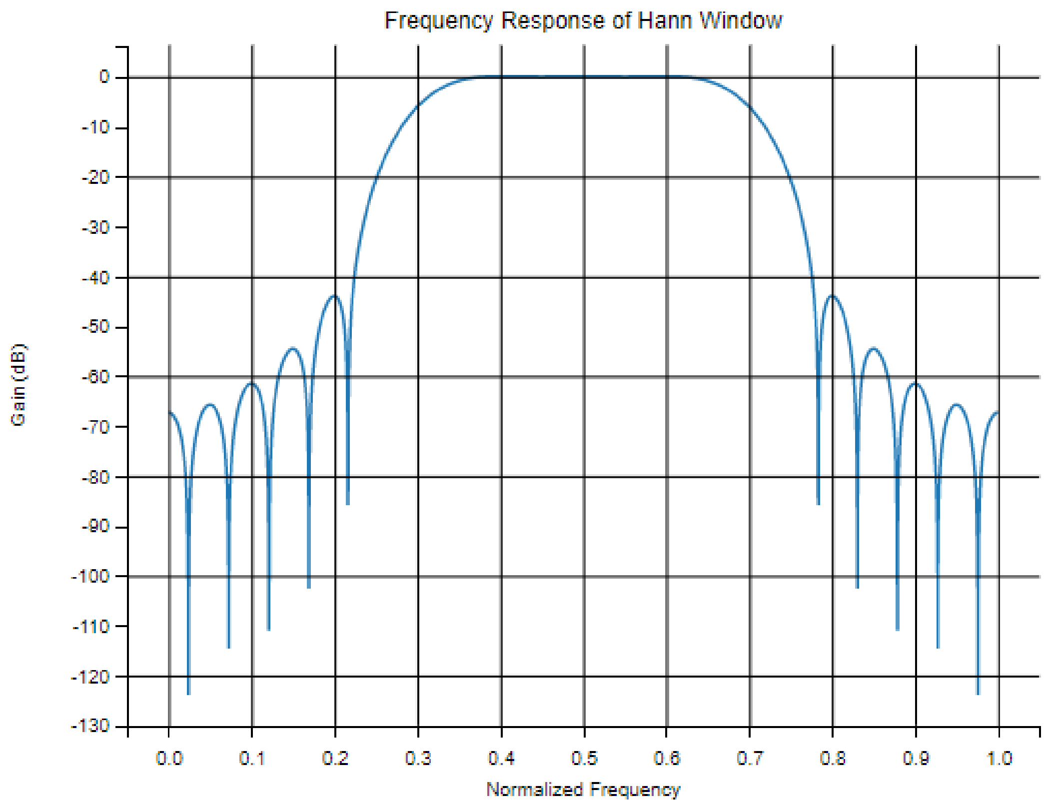

Hann Window
Amplitude
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
5
10
15
20
25
30
35
40
Sample

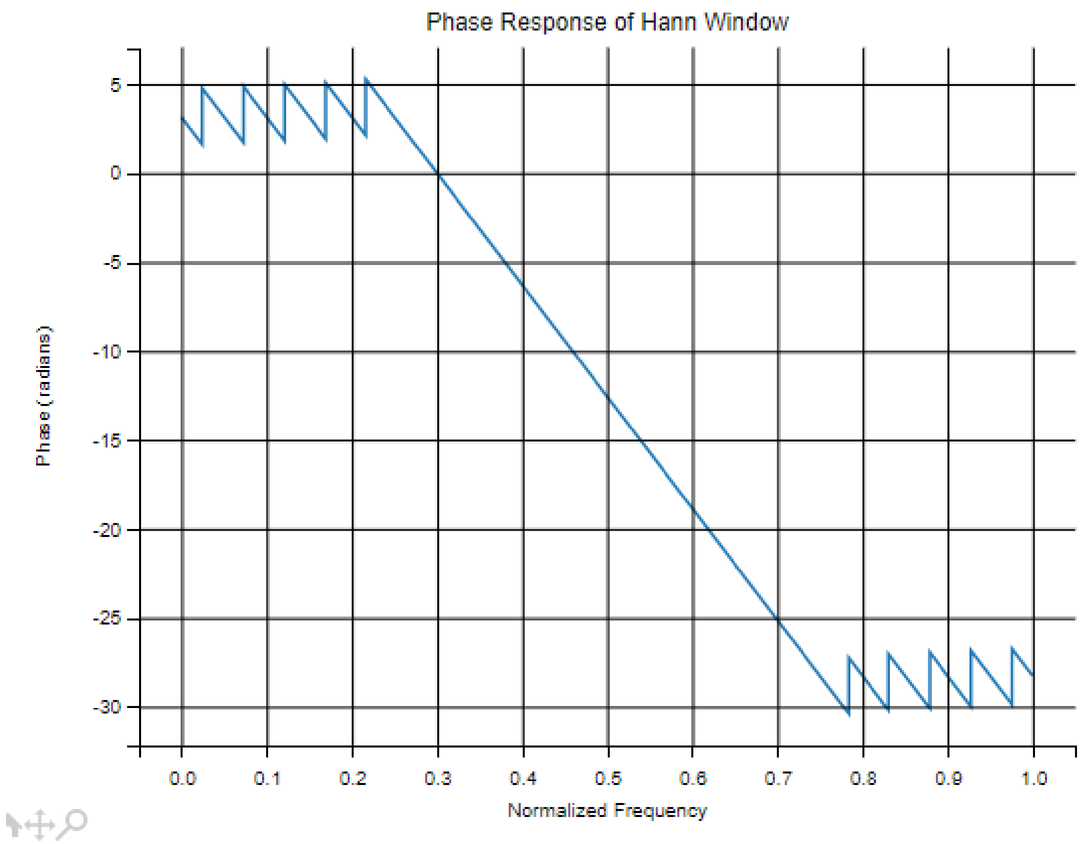
```python
import numpy as np
import matplotlib.pyplot as plt

# Define the values for the first table (stopband table)
stopband = np.array([[0, 0.2 * np.pi], [0.8 * np.pi, 1 * np.pi]])

# Define the values for the second table (passband table)
passband = np.array([[0.4 * np.pi, 0.6 * np.pi]])

# Calculate the "Δω" and use the minimum in order to find M
Delta_omega1 = passband[0, 0] - stopband[0, 1]
print("Delta_omega1:", Delta_omega1)

Delta_omega2 = stopband[1, 0] - passband[0, 1]
print("Delta_omega2:", Delta_omega2)

Delta_omega_final = np.minimum(Delta_omega1, Delta_omega2)
print("Delta_omega_final:", Delta_omega_final)

# for hamming
Hamming_window_length = int(8*np.pi / Delta_omega_final)
print("Hamming_window_length:", Hamming_window_length)
Hamming_window = np.hamming(Hamming_window_length)

# Plot the Hamming window
plt.figure()
plt.stem(Hamming_window)
plt.title("Hamming Window")
plt.xlabel("Sample")
plt.ylabel("Amplitude")

# for Hann
Hann_window_length = int(8*np.pi / Delta_omega_final)
print("Hann_window_length:", Hann_window_length)
Hann_window = np.hanning(Hann_window_length)

# Plot the Hann window
plt.figure()
plt.stem(Hann_window)
plt.title("Hann Window")
plt.xlabel("Sample")
plt.ylabel("Amplitude")

# for Blackman
Blackman_window_length = int(12*np.pi / Delta_omega_final)
print("Blackman_window_length:", Blackman_window_length)
Blackman_window = np.blackman(Blackman_window_length)

# Plot the Blackman window
plt.figure()
plt.stem(Blackman_window)
plt.title("Blackman Window")
plt.xlabel("Sample")
plt.ylabel("Amplitude")

plt.show()
```

Κώδικας για το ερώτημα A1 της 1ης άσκησης

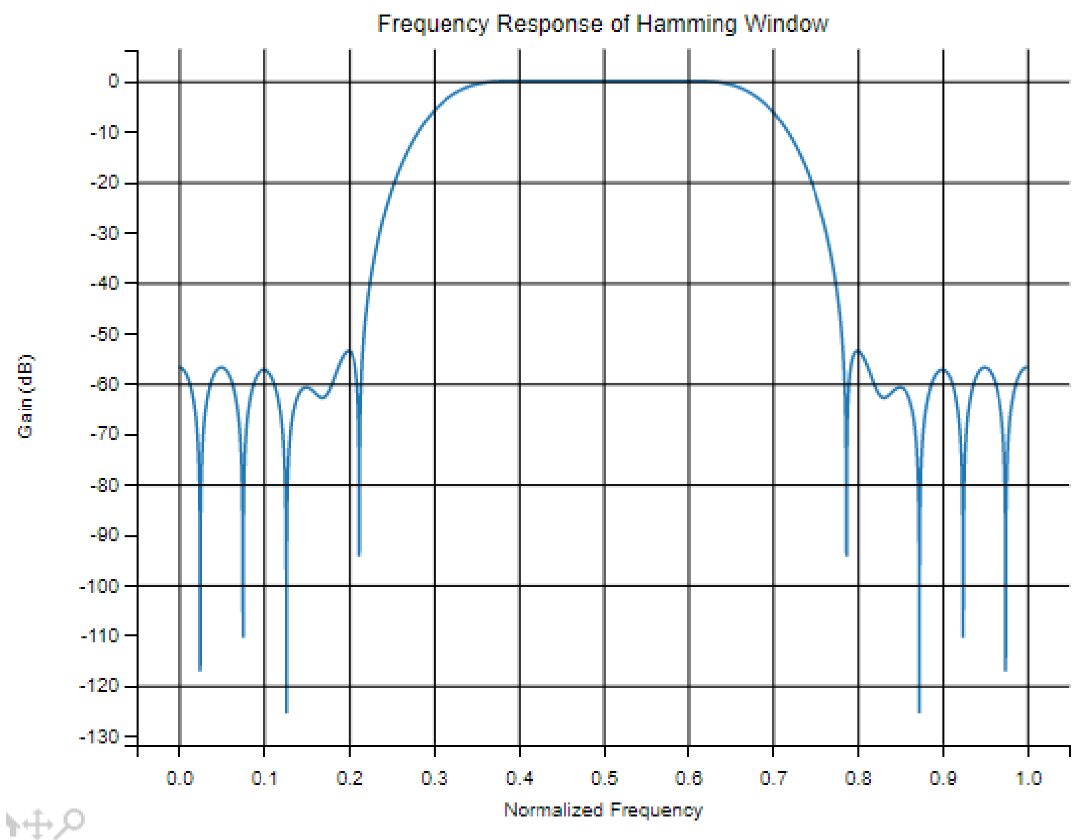
Τα μεγέθη των παραθύρων ήταν M = 40 για Hann και Hamming και 60 για το Blackman.

A2) Στο ερώτημα αυτό μας ζητείται να βρούμε το ζητούμενο φίλτρο είτε πολλαπλασιάζοντας την ιδανική κρουστική απόκριση παλμού με το παράθυρο είτε χρησιμοποιώντας την εντολή fir1 του MATLAB. Εμείς αφού λύσαμε την άσκηση σε python χρησιμοποιήσαμε την αντίστοιχη εντολή στην python η οποία είναι η εντολή firwin της βιβλιοθήκης scipy.signal. Στη συνέχεια σχεδιάσαμε την απόκριση συχνότητας κάθε φίλτρου καθώς και την φάση του χρησιμοποιώντας την εντολή freqz της ίδιας βιβλιοθήκης scippy.signal, όπως φαίνεται στα σχήματα παρακάτω.

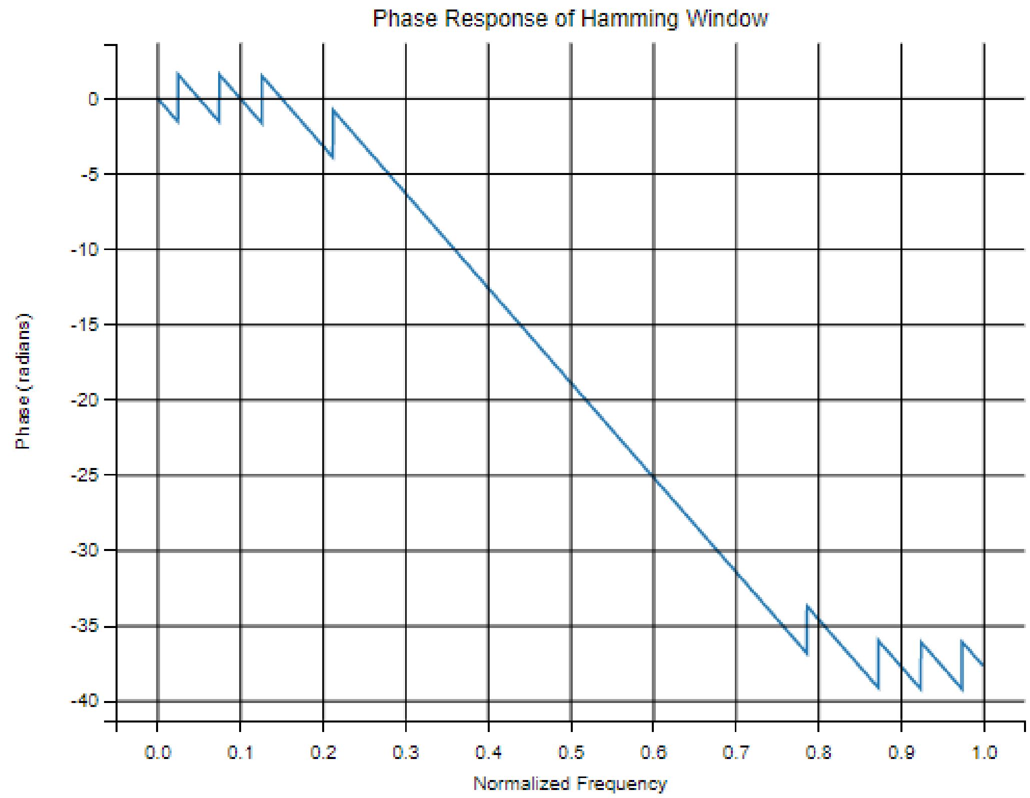


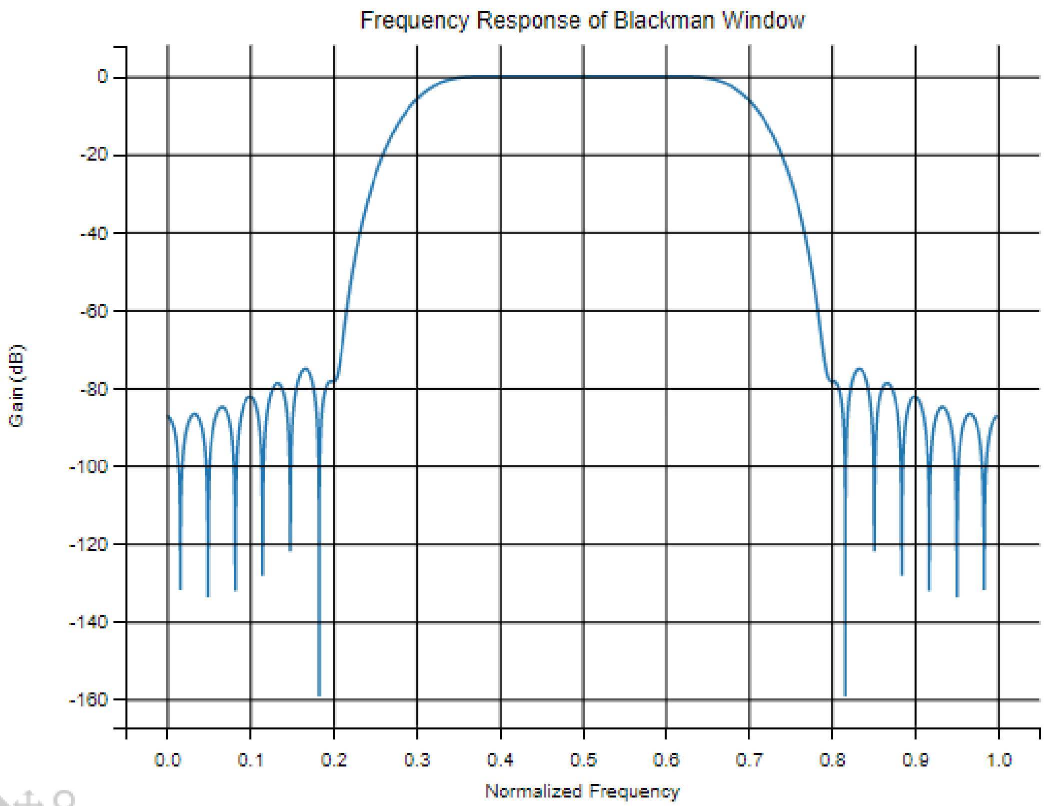


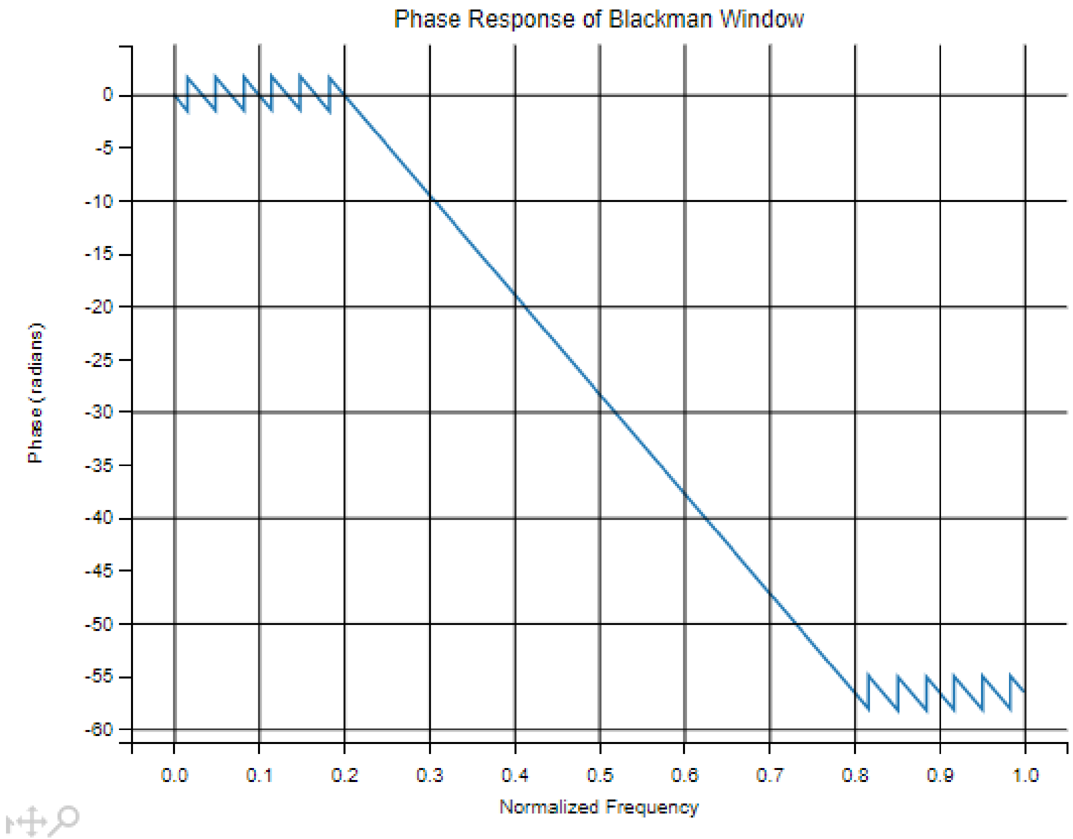
Phase Response of Hann Window
Phase (radians)
5
0
-5
-10
-15
-20
-25
-30
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Normalized Frequency

Frequency Response of Hamming Window
Gain (dB)
0
-10
-20
-30
-40
-50
-60
-70
-80
-90
-100
-110
-120
-130
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Normalized Frequency


Phase Response of Hamming Window
Phase (radians)
0
-5
-10
-15
-20
-25
-30
-35
-40
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Normalized Frequency

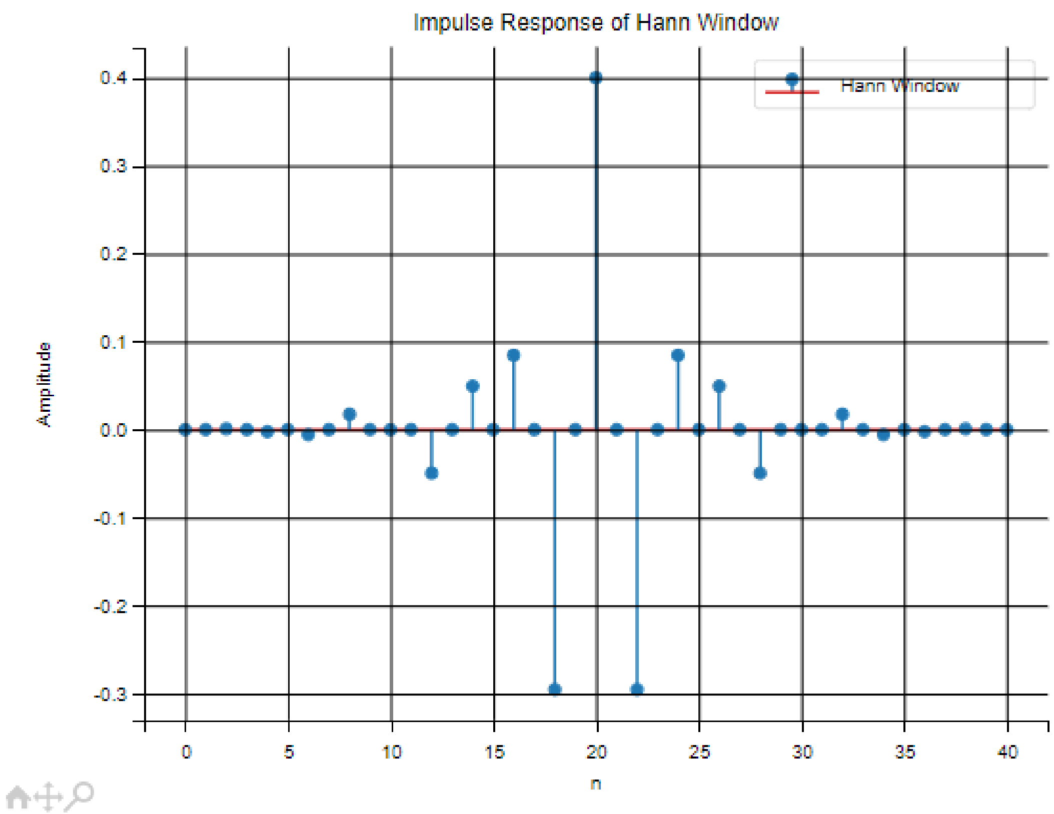

Frequency Response of Blackman Window
Gain (dB)
0
-20
-40
-60
-80
-100
-120
-140
-160
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Normalized Frequency

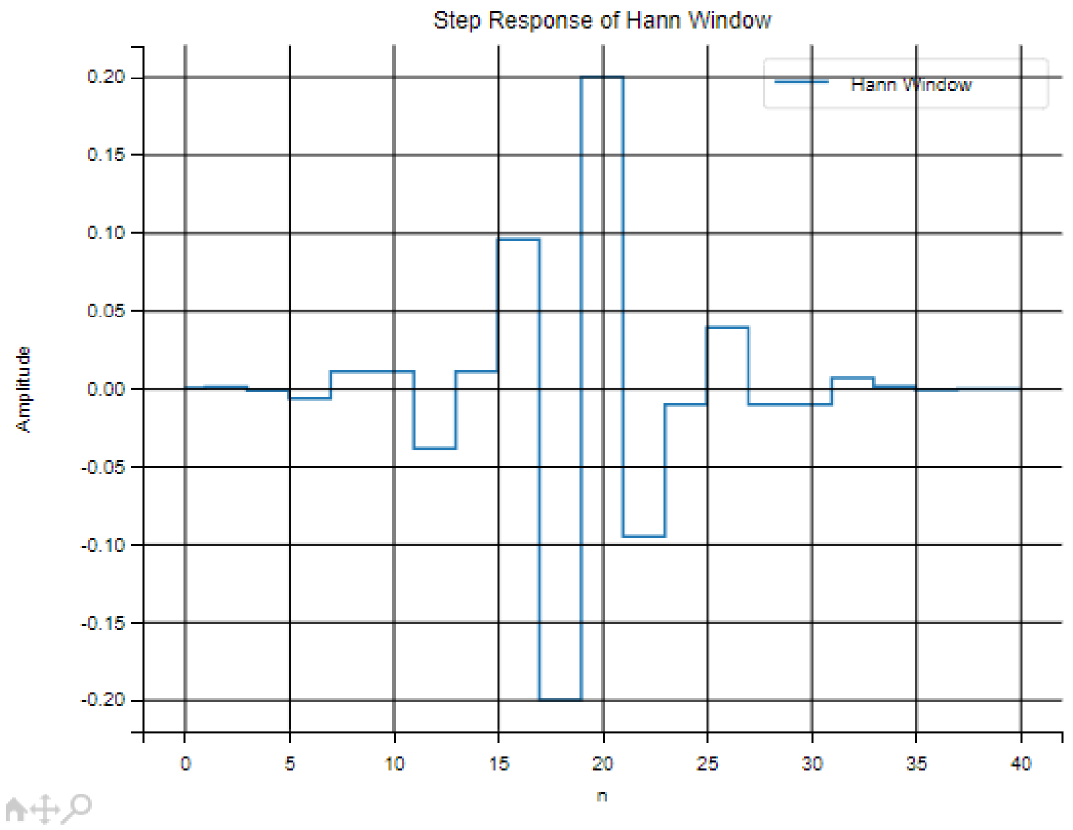


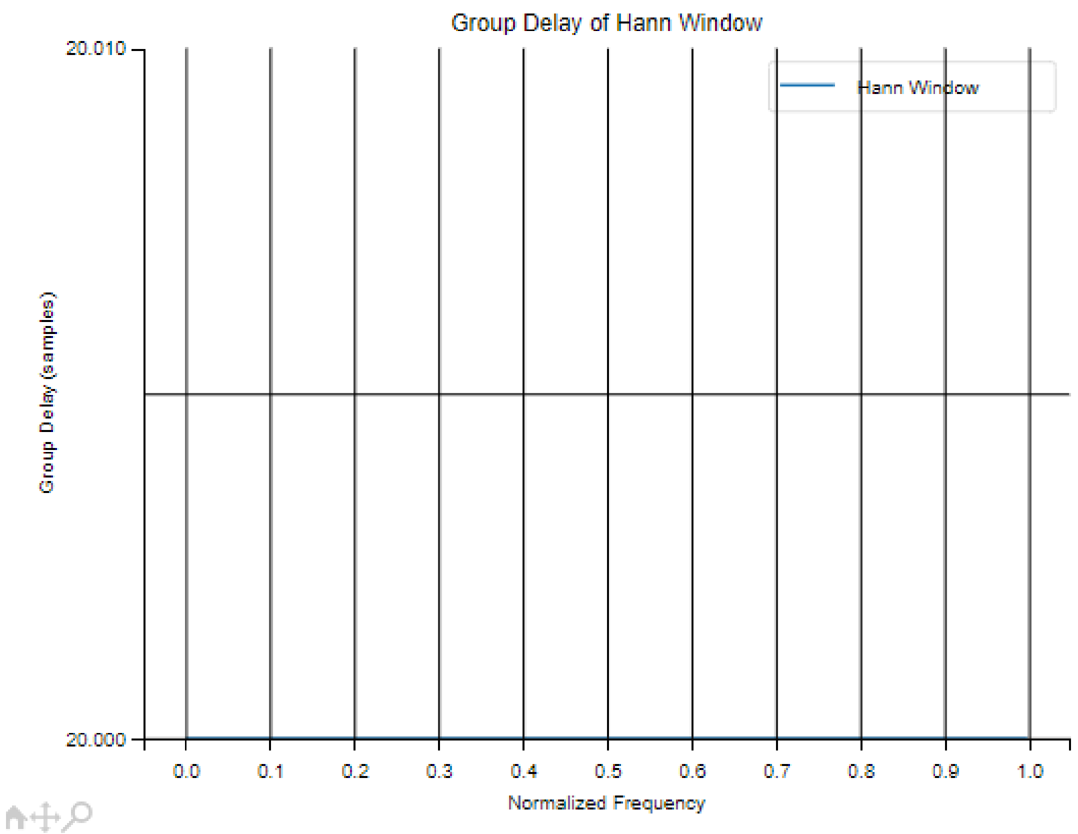

Ακολουθεί ο κώδικας για το δεύτερο υποερώτημα της πρώτης άσκησης στην επόμενη σελίδα.

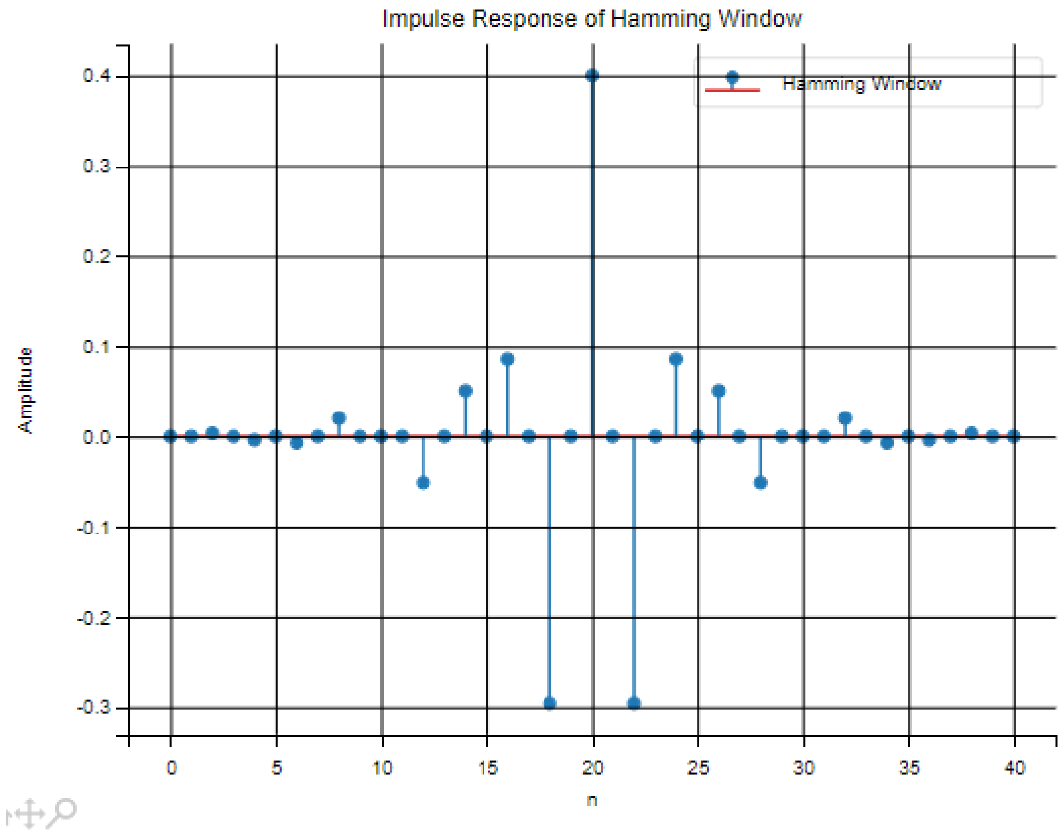
```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import firwin, freqz


# function to plot the frequency response of each filter
def plot_frequency_response(w, h_freq, window_name):
    plt.figure(figsize=(8, 6))
    plt.plot(w / np.pi, 20 * np.log10(abs(h_freq)))
    plt.title(f'Frequency Response of {window_name} Window')
    plt.xlabel('Normalized Frequency')
    plt.ylabel('Gain (dB)')
    plt.grid(True)
    plt.show()


# function to plot the phase response of each filter
def plot_phase_response(w, h_freq, window_name):
    plt.figure(figsize=(8, 6))
    plt.plot(w / np.pi, np.unwrap(np.angle(h_freq)))
    plt.title(f'Phase Response of {window_name} Window')
    plt.xlabel('Normalized Frequency')
    plt.ylabel('Phase (radians)')
    plt.grid(True)
    plt.show()


# Define the size of each window (found previously)
M_hann = 40
M_hamm = 40
M_blac = 60

# Frequency range
freq_range = [(0.20 + 0.40) / 2, (0.60 + 0.80) / 2]

# Design FIR filters using firwin
h_hann = firwin(M_hann + 1, freq_range, pass_zero=False, window='hann', scale=True)
h_hamm = firwin(M_hamm + 1, freq_range, pass_zero=False, window='hamming',
scale=True)
h_blac = firwin(M_blac + 1, freq_range, pass_zero=False, window='blackman',
scale=True)

# Frequency response for Hann Window
w_hann, h_freq_hann = freqz(h_hann, worN=8000, fs=2 * np.pi)
plot_frequency_response(w_hann, h_freq_hann, 'Hann')

# Phase response for Hann Window
plot_phase_response(w_hann, h_freq_hann, 'Hann')

# Frequency response for Hamming Window
w_hamm, h_freq_hamm = freqz(h_hamm, worN=8000, fs=2 * np.pi)
plot_frequency_response(w_hamm, h_freq_hamm, 'Hamming')

# Phase response for Hamming Window
plot_phase_response(w_hamm, h_freq_hamm, 'Hamming')

# Frequency response for Blackman Window
w_blac, h_freq_blac = freqz(h_blac, worN=8000, fs=2 * np.pi)
plot_frequency_response(w_blac, h_freq_blac, 'Blackman')

# Phase response for Blackman Window
plot_phase_response(w_blac, h_freq_blac, 'Blackman')
```

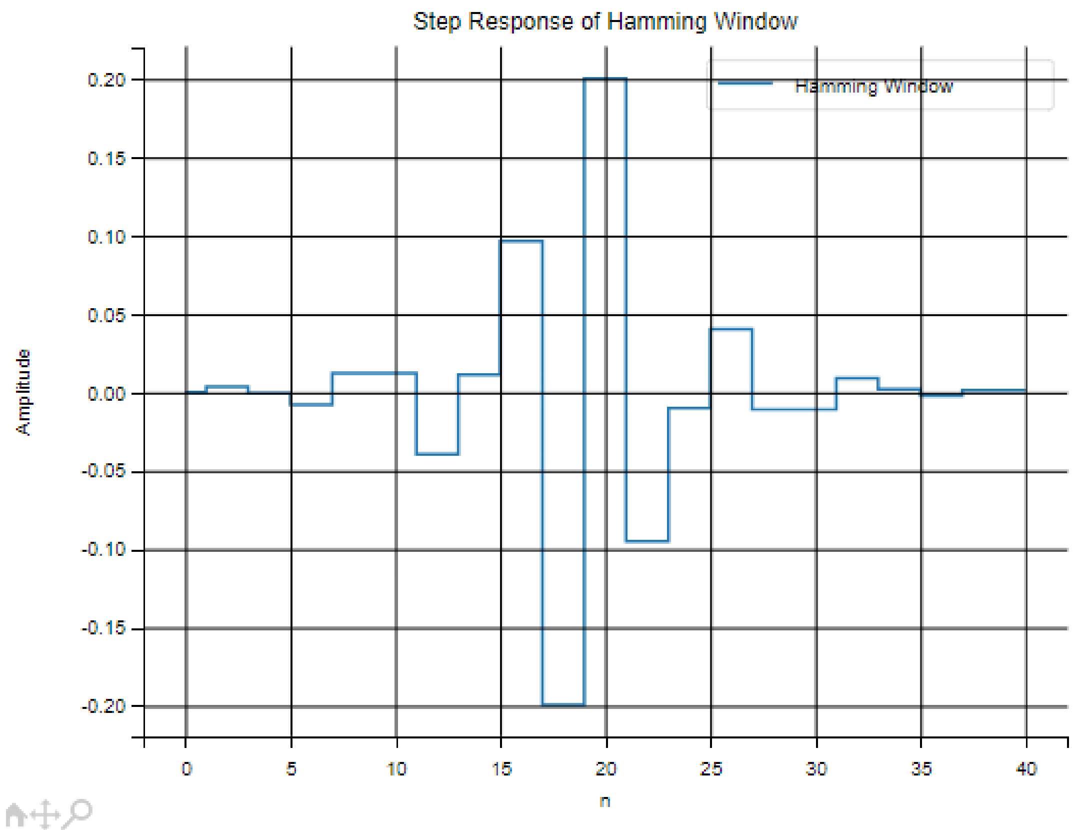
A3) Στο ερώτημα αυτό σχεδιάζουμετην κρουστική απόκριση του φίλτρου, την βηματική απόκριση αυτού, το μέτρο της απόκρισης συχνότητάς του σε λογαριθμική κλίμακα (σε dB), και την καθυστέρηση ομάδας του.Δημιουργούμε custom συναρτήσεις python σε συνδυασμό με συναρτήσεις της βιβλιοθήκης spicy της python όπως η συνάρτηση group_delay. Οι μεταβλητές f1 και f2 χρησιμοποιούνται για τον υπολογισμό των συχνοτήτων αποκοπής του φίλτρου και ορίζονται ως το μισό του αθροίσματος των συχνοτήτων του stopband και του passband divided by π. Το παράθυρο Hann υπολογίζεται πολλαπλασιάζοντας το αποτέλεσμα του υπολογισμού (- f1*sinc(f1*k)+f2*sinc(f2*k)) με το παράθυρο Hann W_hann του οποίου την συνάρτηση παίρνουμε έτοιμη από τη συνάρτηση hann της spicy.signal.windows βιβλιοθήκη που έχει αντίστοιχες συναρτήσεις και για το παράθυρο hamming και για το blackman. Ομοίως, ενεργούμε και για τα παράθυρα Hamming και Blackman. Παρουσιάζουμε τα αποτελέσματα στις γραφικές παραστάσεις παρακάτω.

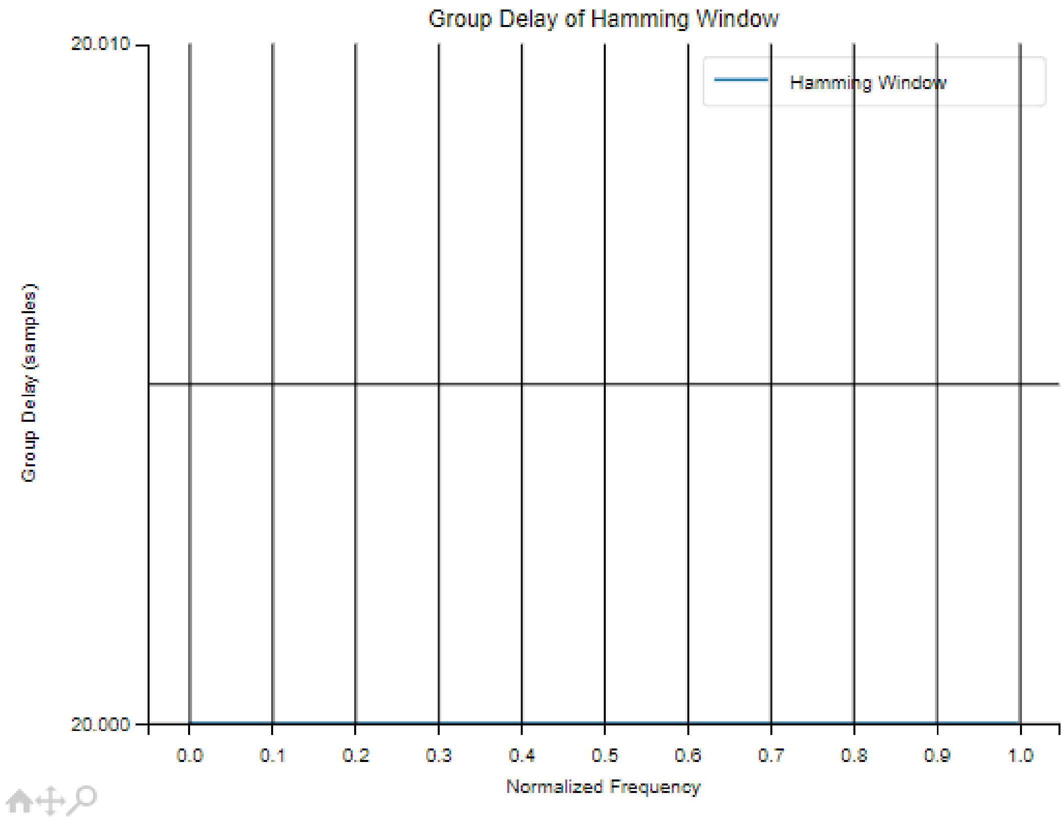


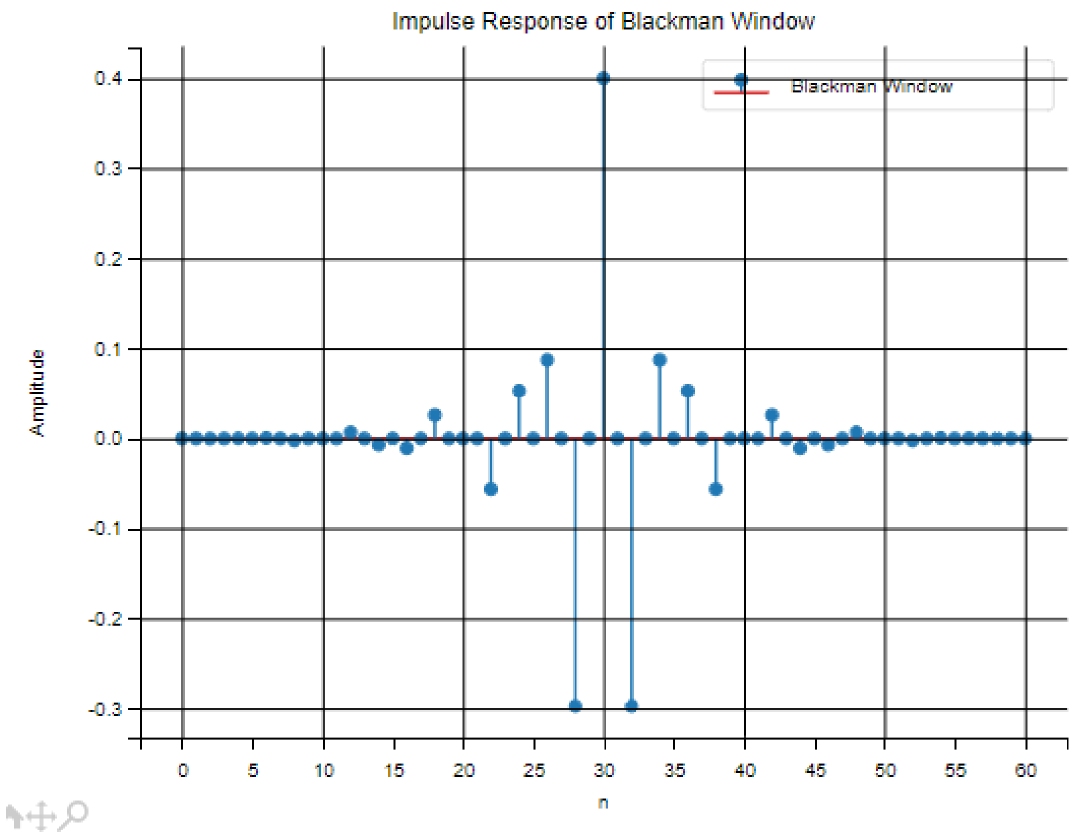

Step Response of Hann Window
Hann Window
Amplitude
0.20
0.15
0.10
0.05
0.00
-0.05
-0.10
-0.15
-0.20
0
5
10
15
20
25
30
35
40
n

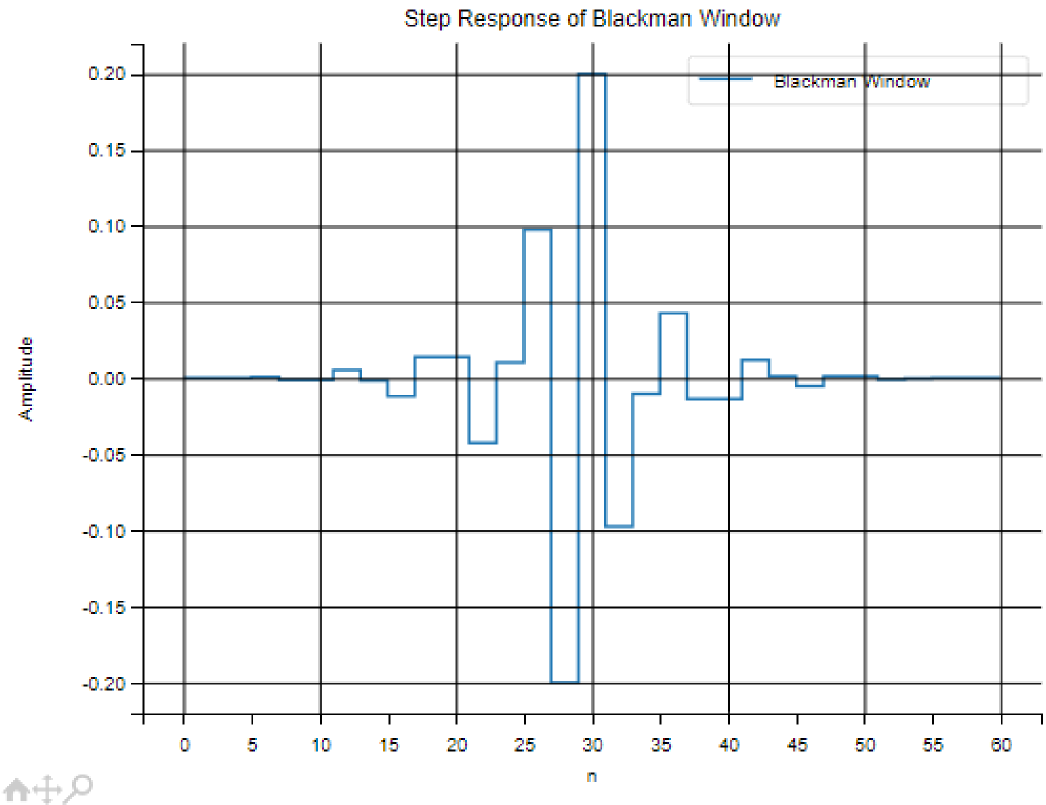

Group Delay of Hann Window
Hann Window
Group Delay (samples)
20.010
20.000
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Normalized Frequency


Impulse Response of Hamming Window
Hamming Window
Amplitude
n
0.4
0.3
0.2
0.1
0.0
-0.1
-0.2
-0.3
0
5
10
15
20
25
30
35
40

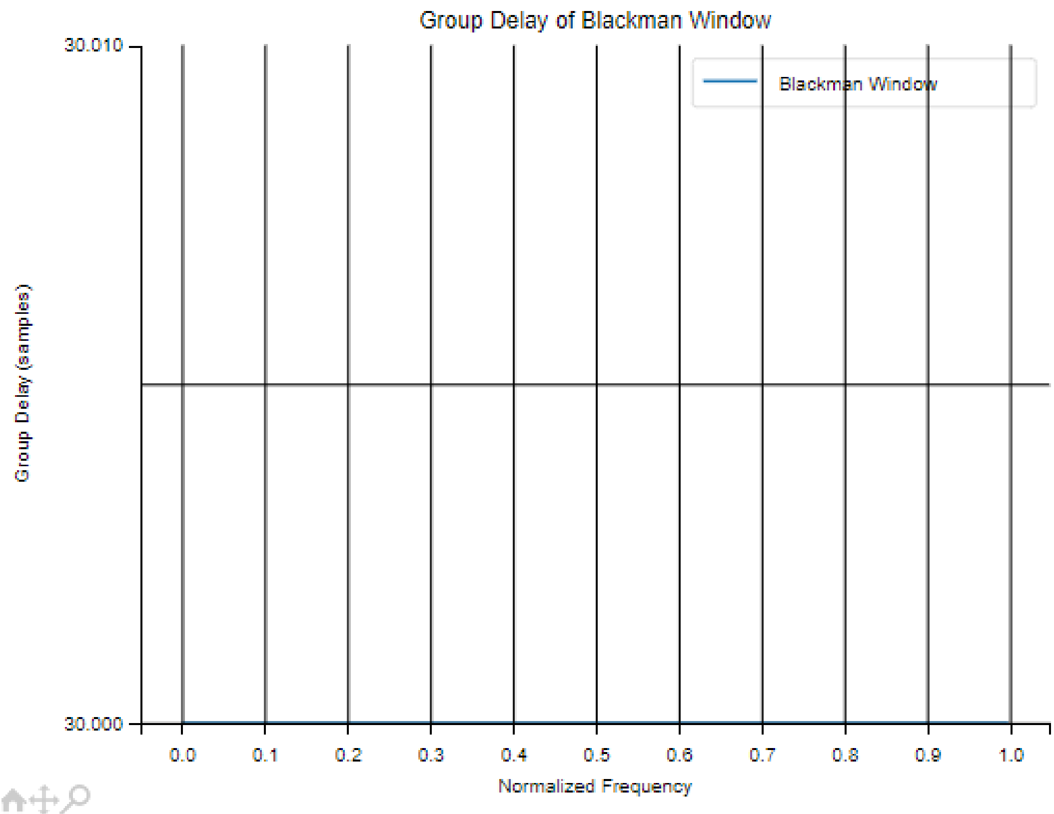

Step Response of Hamming Window
Hamming Window
Amplitude
n
0.20
0.15
0.10
0.05
0.00
-0.05
-0.10
-0.15
-0.20
0
5
10
15
20
25
30
35
40


Group Delay of Hamming Window
Hamming Window
Group Delay (samples)
20.010
20.000
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Normalized Frequency


Impulse Response of Blackman Window
Blackman Window
Amplitude
0.4
0.3
0.2
0.1
0.0
-0.1
-0.2
-0.3
0
5
10
15
20
25
30
35
40
45
50
55
60
n



Step Response of Blackman Window
Blackman Window
Amplitude
0.20
0.15
0.10
0.05
0.00
-0.05
-0.10
-0.15
-0.20
0
5
10
15
20
25
30
35
40
45
50
55
60
n

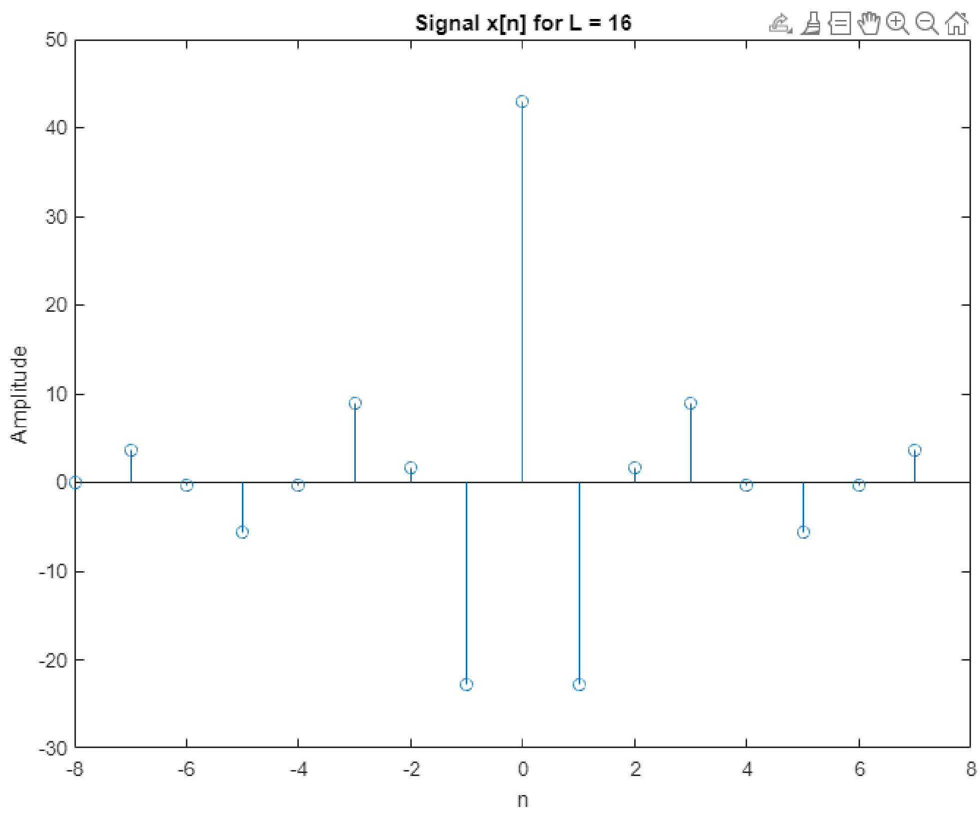
Group Delay of Blackman Window
Blackman Window
Group Delay (samples)
30.010
30.000
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
Normalized Frequency

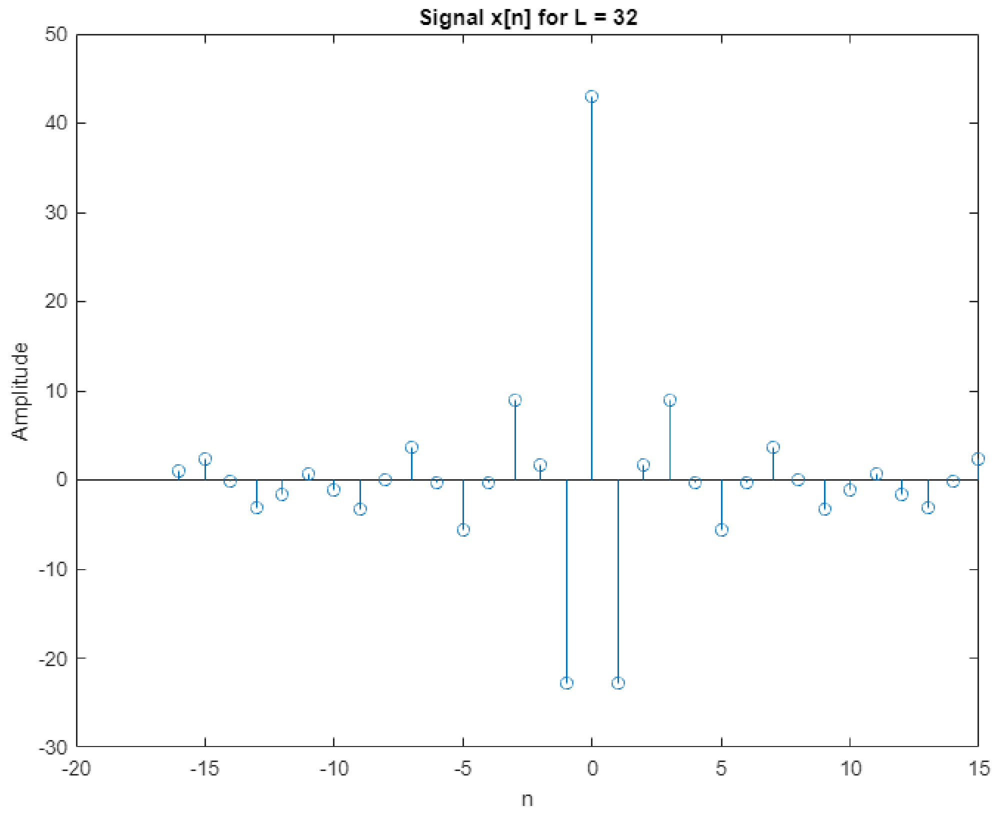
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import group_delay
from scipy.signal.windows import hann, hamming, blackman

# Define parameters passband and stopband
p_a = 0.20 * np.pi
p_b = 0.40 * np.pi
s_a = 0.60 * np.pi
s_b = 0.80 * np.pi

# Calculate normalized frequencies
f1 = ((p_a + p_b) / 2) / np.pi
f2 = ((s_a + s_b) / 2) / np.pi

# Hann Window with M = 40
M_hann = 40
n_hann = np.arange(0, M_hann + 1)
k_hann = n_hann - M_hann / 2
W_hann = hann(M_hann + 1)
hhann = (-f1 * np.sinc(f1 * k_hann) + f2 * np.sinc(f2 * k_hann)) *
W_hann

# Hamming Window with M = 40
M_hamm = 40
n_hamm = np.arange(0, M_hamm + 1)
k_hamm = n_hamm - M_hamm / 2
W_hamm = hamming(M_hamm + 1)
hhamm = (-f1 * np.sinc(f1 * k_hamm) + f2 * np.sinc(f2 * k_hamm)) *
W_hamm

# Blackman Window with M = 60
M_blac = 60
n_blac = np.arange(0, M_blac + 1)
k_blac = n_blac - M_blac / 2
W_blac = blackman(M_blac + 1)
hblac = (-f1 * np.sinc(f1 * k_blac) + f2 * np.sinc(f2 * k_blac)) *
W_blac


# Impulse response plots function
def plot_impulse_response(h, window_name):
    plt.figure(figsize=(8, 4))
    plt.stem(np.arange(len(h)), h, label=f'{window_name} Window')
    plt.title(f'Impulse Response of {window_name} Window')
    plt.xlabel('n')
    plt.ylabel('Amplitude')
    plt.legend()
    plt.grid(True)
    plt.show()


# Step response plots function
def plot_step_response(h, window_name):
    plt.figure(figsize=(8, 4))
    plt.step(np.arange(len(h)), np.cumsum(h),
label=f'{window_name} Window')
    plt.title(f'Step Response of {window_name} Window')
    plt.xlabel('n')
    plt.ylabel('Amplitude')
    plt.legend()
```

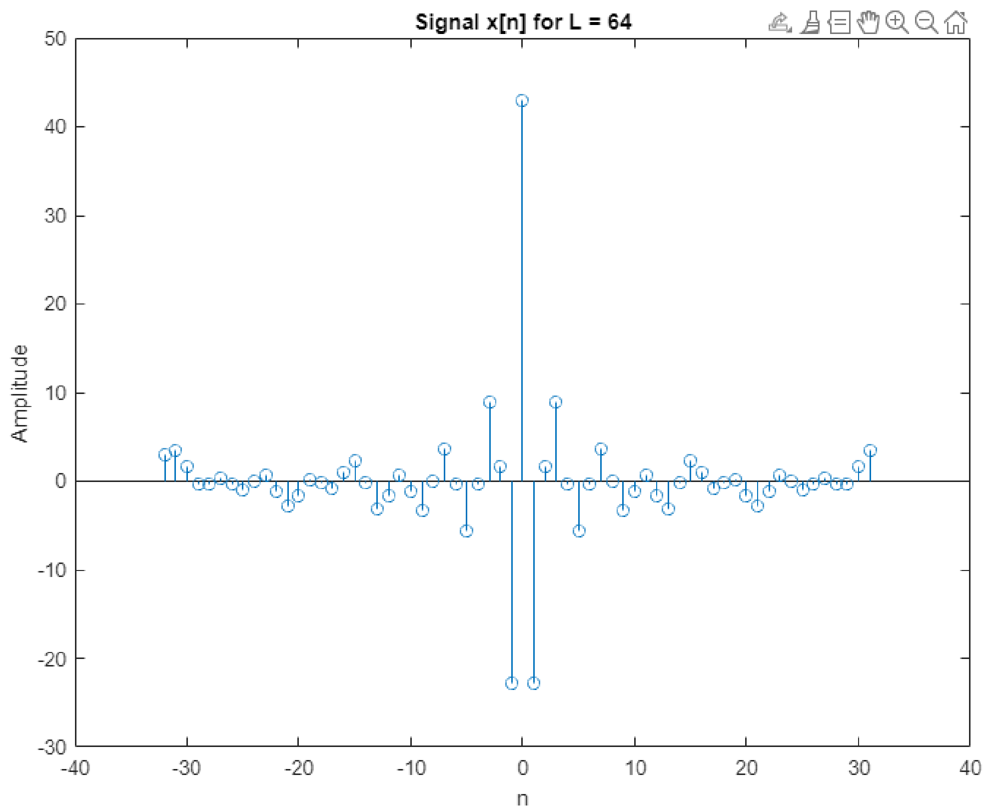
Παραπάνω φαίνεται ο κώδικας και για αυτό το ερώτημα.

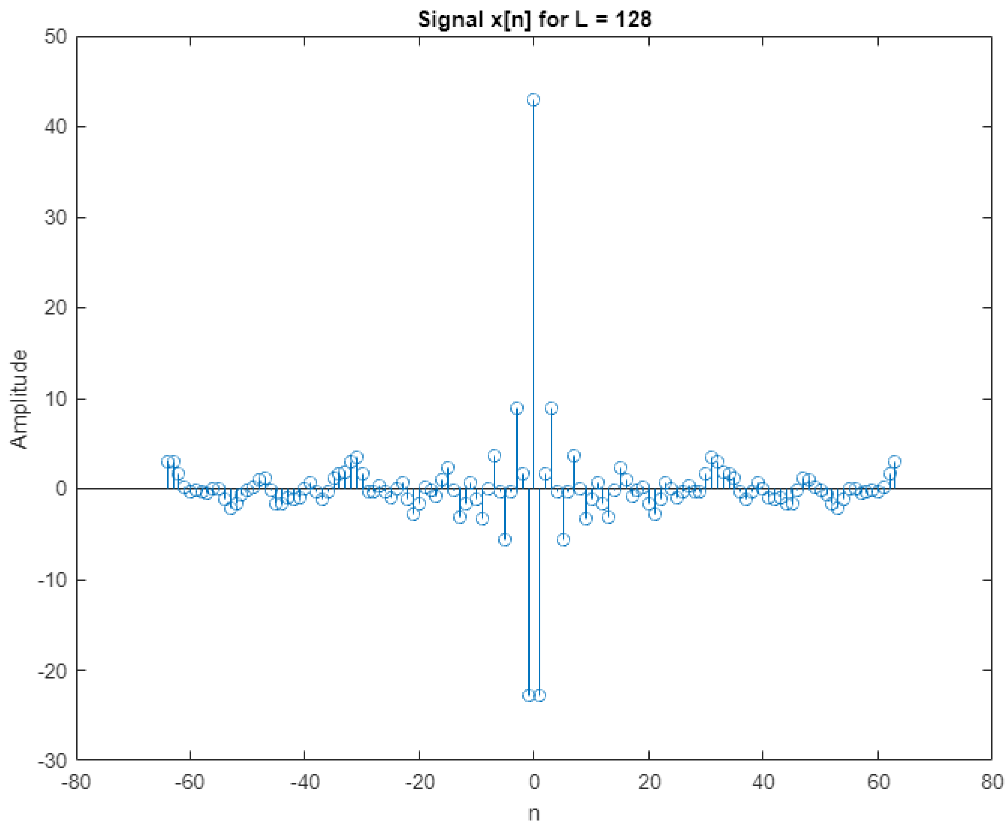
## Άσκηση $2^η$

B1) Στο ερώτημα αυτό μας ζητείται να παραθυρώσουμε το σήμα για κάθε L της εκφώνησης και ύστερα να το σχεδιάσουμε. Ο κώδικας μας εφαρμόζει ένα ορθογώνιο παράθυρο διαφόρων μηκών L στο διάστημα [-L/2, L/2 -1] στο σήμα εισόδου και υπολογίζει το σήμα που προκύπτει μετά την εφαρμογή του κάθε παραθύρου. Για κάθε τιμή του L το παράθυρο εφαρμόζεται στο αρχικό σήμα μετατοπίζοντας το κατά L/2 αριστερά και υπολογίζεται το σήμα που προκύπτει από την παραθύρωση. Ακολουθούν τα διαγράμματα που μας δείχνουν το αποτέλεσμα κάθε παραθύρωσης. (Η άσκηση αυτή υλοποιήθηκε σε MATLAB όχι σε python όπως η προηγούμενη).

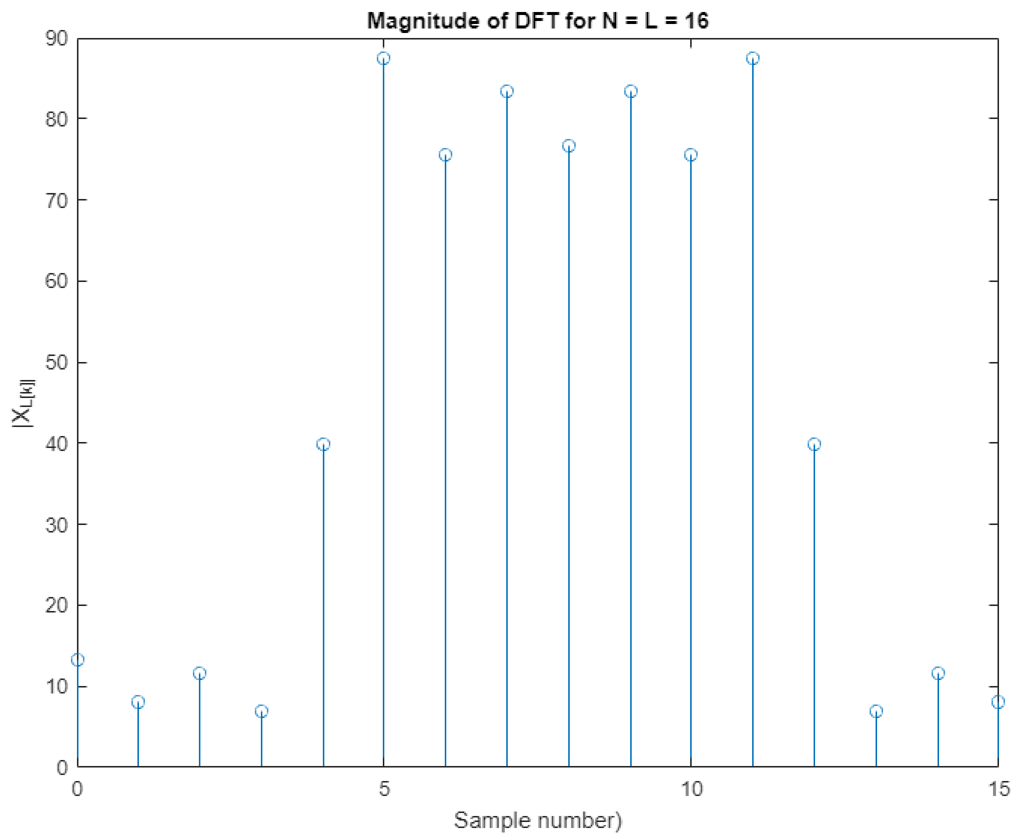


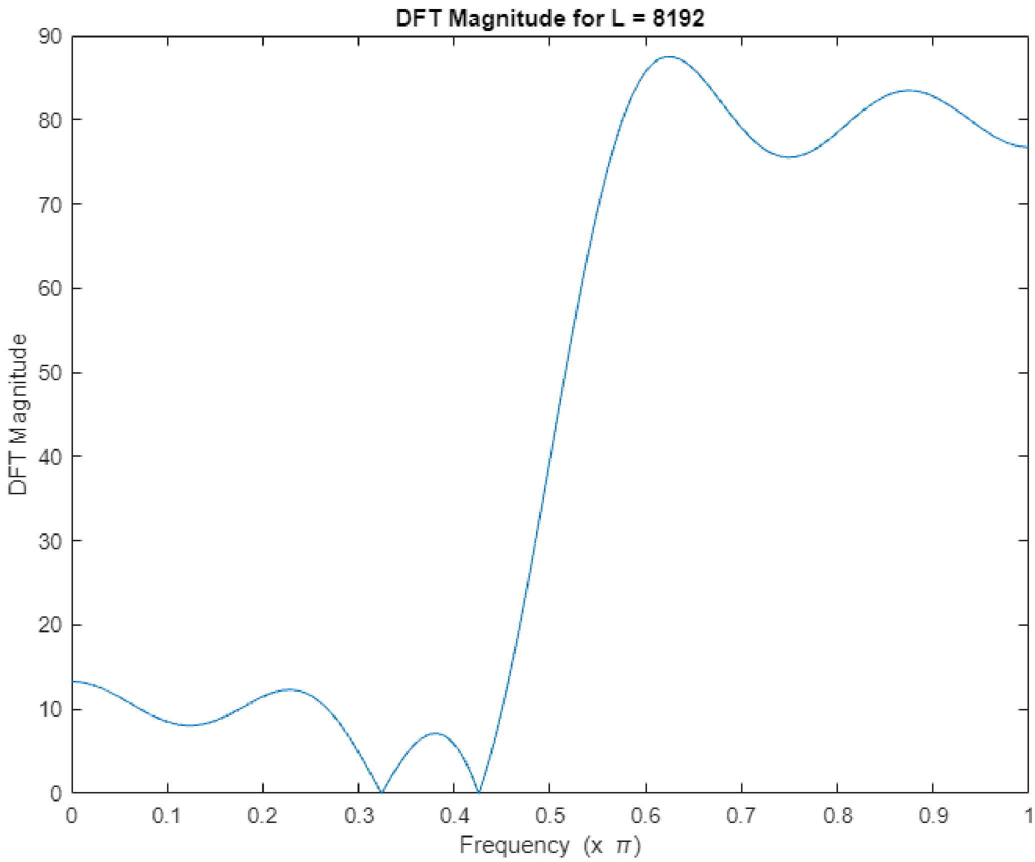

Signal x[n] for L = 32
Amplitude
n
50
40
30
20
10
0
-10
-20
-30
-20
-15
-10
-5
0
5
10
15

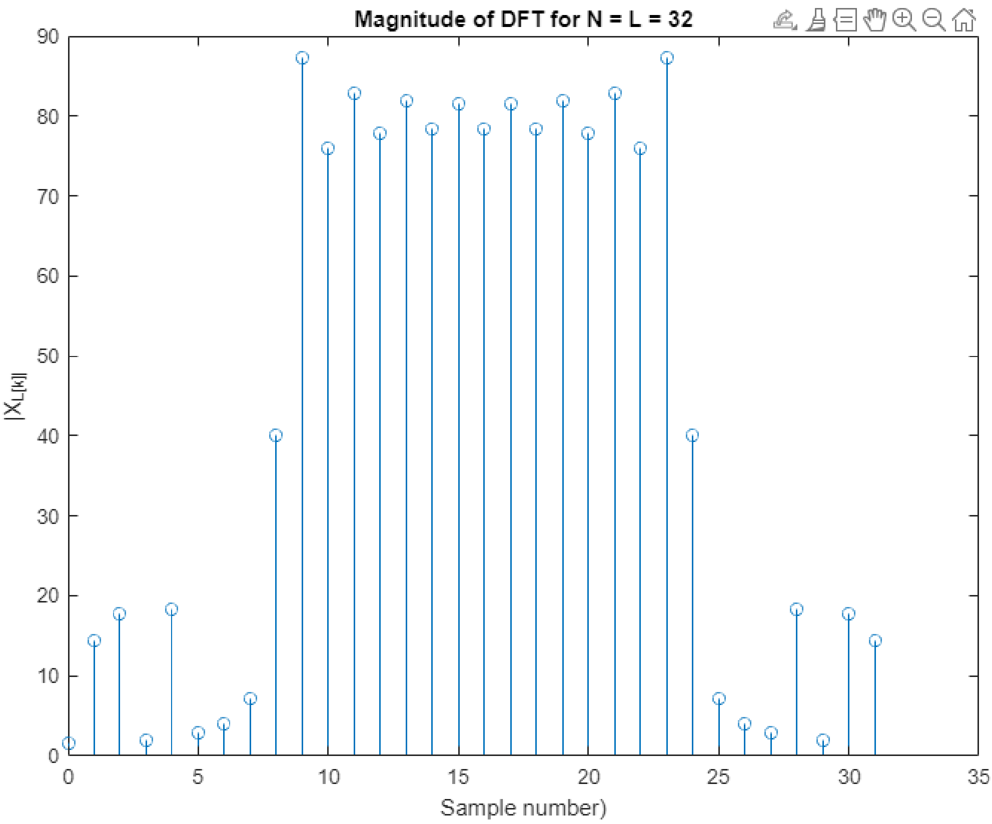

Signal x[n] for L = 64
Amplitude
n
50
40
30
20
10
0
-10
-20
-30
-40
-30
-20
-10
0
10
20
30
40

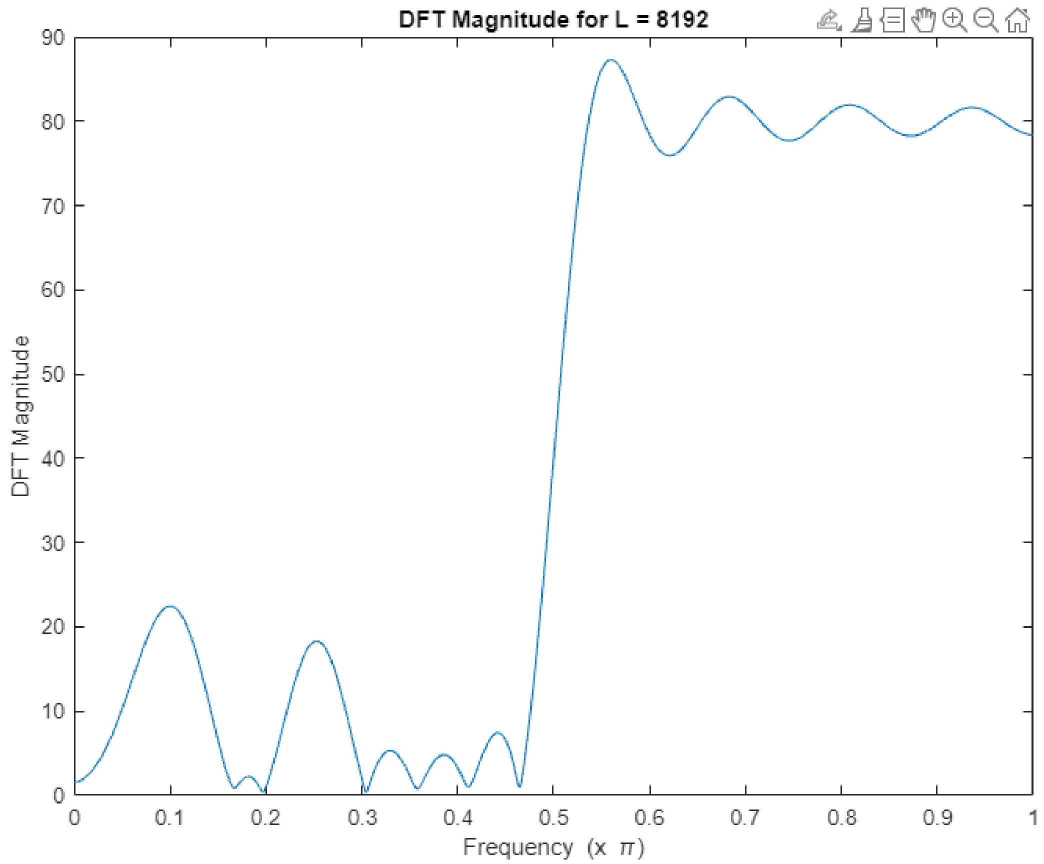


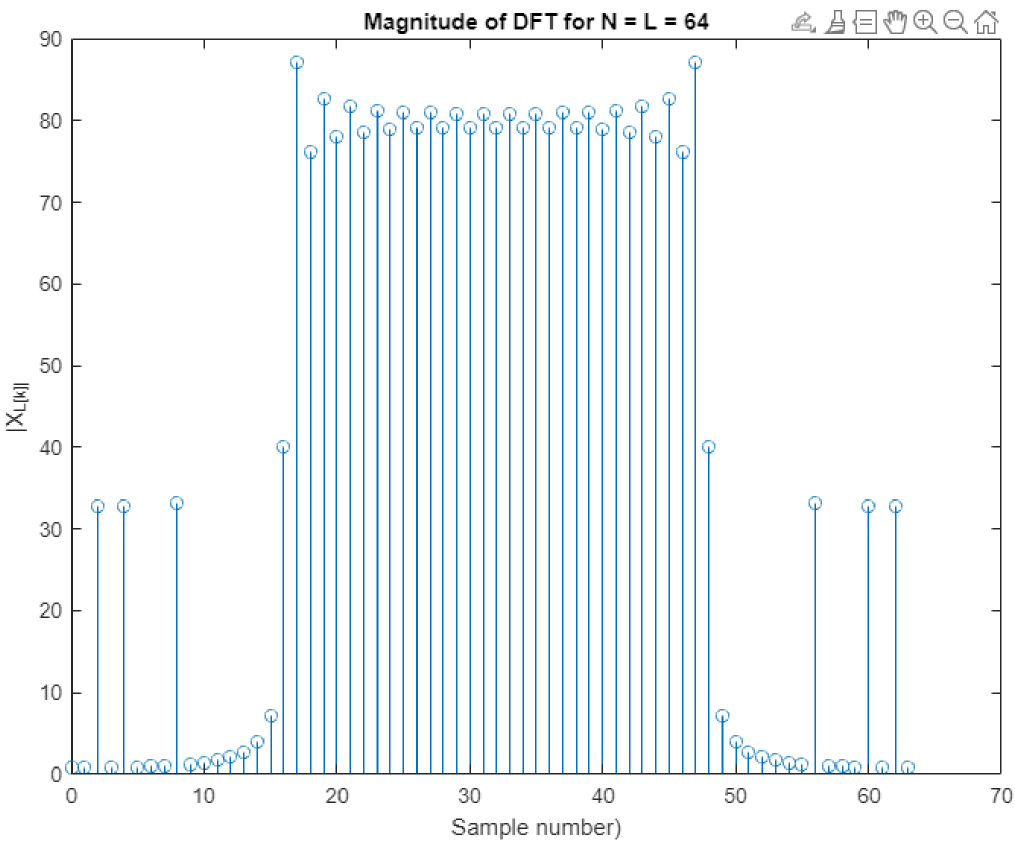

B2) Τα διαγράμματα για το παραθυρωμένο σήμα με υπολογισμό DFT είναι τα παρακάτω:

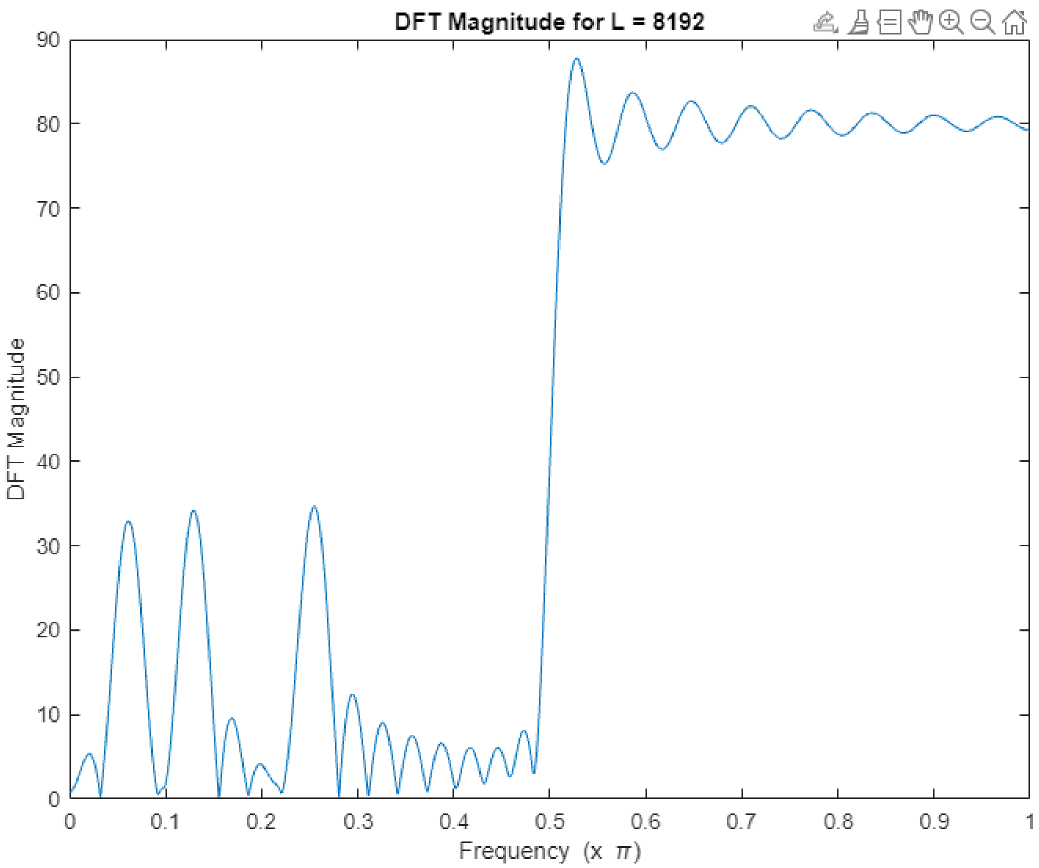


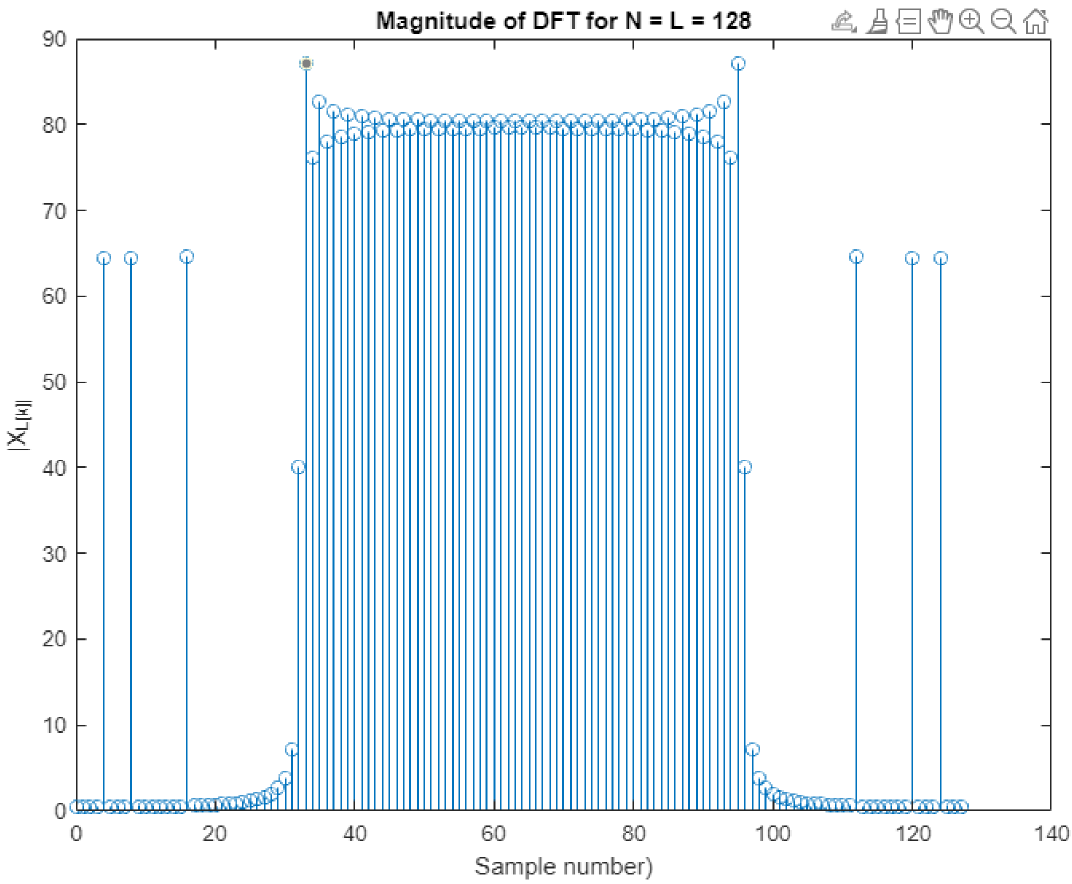

DFT Magnitude for L = 8192
DFT Magnitude
Frequency (x π)
0
10
20
30
40
50
60
70
80
90
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

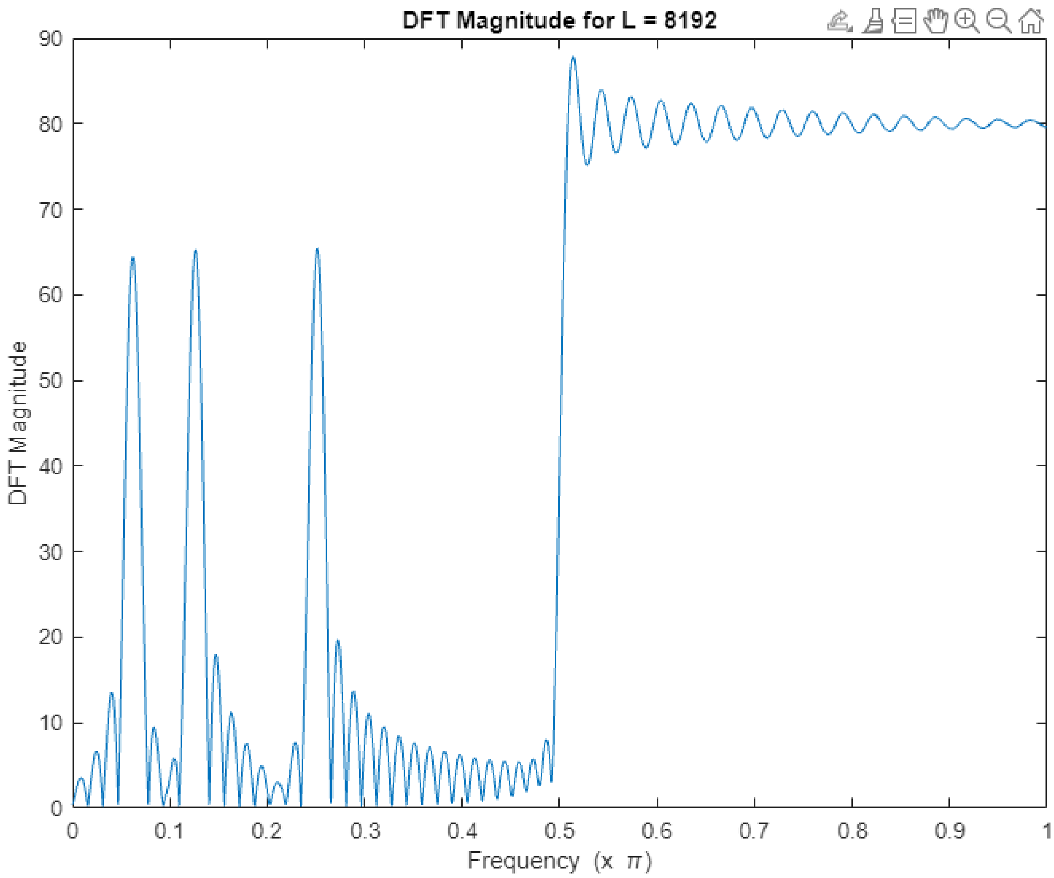

Magnitude of DFT for N = L = 32
|X_L[k]|
Sample number)
0
10
20
30
40
50
60
70
80
90
5
15
25
35


DFT Magnitude for L = 8192
DFT Magnitude
Frequency (x π)
0
10
20
30
40
50
60
70
80
90
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Magnitude of DFT for N = L = 64
|X_L[k]|
Sample number)
0
10
20
30
40
50
60
70
80
90


DFT Magnitude for L = 8192
DFT Magnitude
Frequency (x π)
0
10
20
30
40
50
60
70
80
90
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

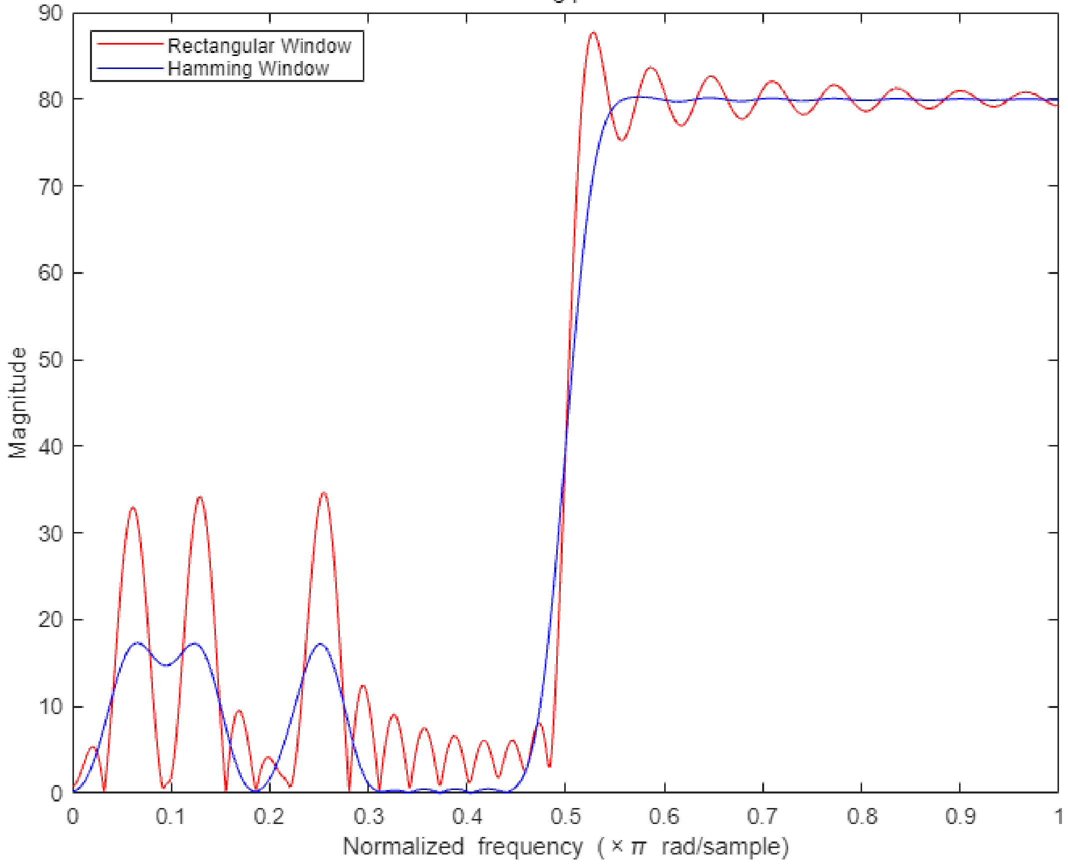

Magnitude of DFT for N = L = 128
$|X_{L[k]}|$
Sample number)
0
10
20
30
40
50
60
70
80
90
100
120
140



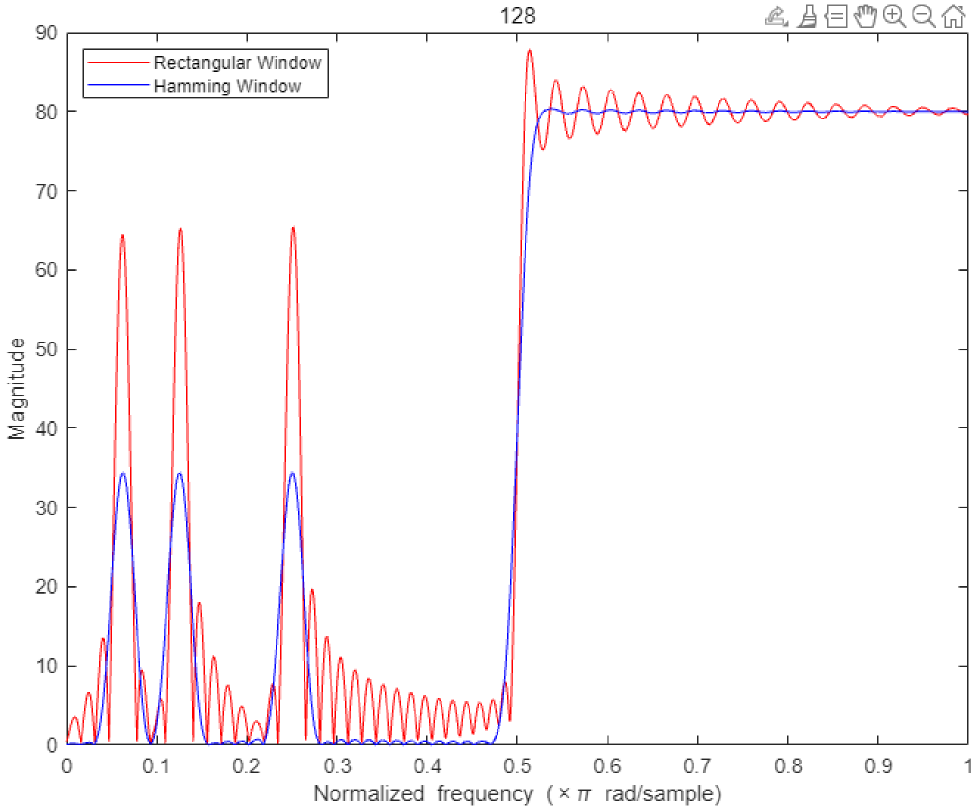

B3) Καθώς η τιμή του L αυξάνεται παρατηρούμε ότι η ανάλυση της γραφικής παράστασης  βελτιώνεται και είναι πιο ορατές οι λεπτομέρειες του σήματος. Ο κυματισμός και οι πλευρικοί λοβοί που παρατηρούμε στα σχήματά μας είναι αποτέλεσμα του πεπερασμένου μήκους του σήματος και μπορούν να επηρεάσουν αρνητικά την ακρίβεια της γραφικής παράστασης μεγέθους του DFT. Οι κορυφές στα διαγράμματα αντιστοιχούν στις συχνότητες που υπάρχουν στο σήμα συνεπώς το συχνοτικό περιεχόμενο του σήματος είναι εμφανές σε αυτά. Το συχνοτικό περιεχόμενο του σήματος γίνεται ξεκάθαρο και πιο ακριβές με την με την αύξηση του L.

B4) Σε αυτό το ερώτημα μας ζητήθηκε να εφαρμόσουμε ένα παράθυρο Hamming για L = 64 , 128 και N = 8192 και συγκρίναμε το φάσμα που προκύπτει από το παράθυρο hamming με αυτό που προκύπτει από το ορθογώνιο παράθυρο. Παραθέτουμε τα αποτελέσματα παρακάτω και ύστερα θα τα σχολιάσουμε περαιτέρω.

Comparison of Rectangular and Hamming Windows, L = 64

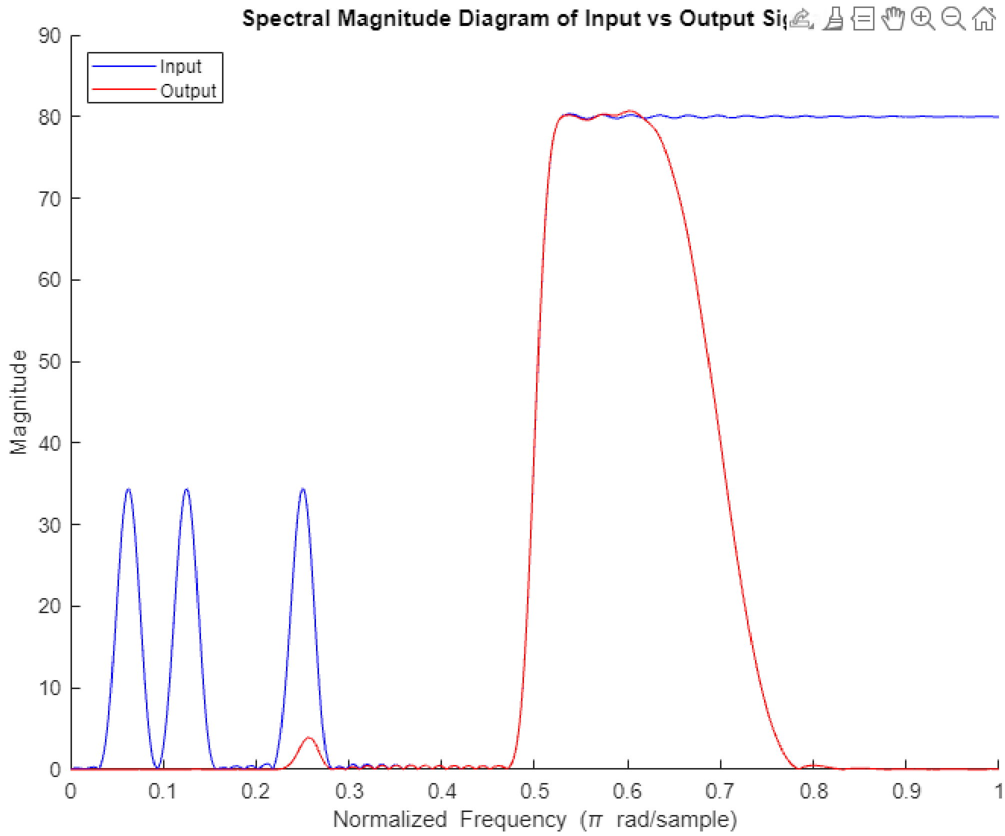



Comparison of Rectangular and Hamming Windows, L = 128



Από τα αποτελέσματα είναι εμφανές ότι το παράθυρο Hamming μειώνει σημαντικά τα επίπεδα κυματισμού και πλευρικού λοβού στο φάσμα του σήματος σε σχέση με το ορθογώνιο παράθυρο. Συγκεκριμένα για L = 64 τα επίπεδα κυματισμού μειώνονται κατά περίπου 12db και τα επίπεδα του πλευρικού λοβού μειώνονται κατά περίπου 8db (όπως φαίνεται στα διαγράμματα παραπάνω). Καταλήγουμε στο συμπέρασμα πως το παράθυρο Hamming αποτελεί χρήσιμο εργαλείο για την μείωση των επιπέδων κυματισμού και πλευρικού λοβού ενώ ταυτόχρονα το συχνοτικό περιεχόμενο του σήματος παραμένει σχετικά το ίδιο.

B5) Το φίλτρο διέλευσης ζωνών που εφαρμόζουμε στο σιγκεκριμένο ερώτημα φιλτράρει το σήμα με συντελεστές φίλτρου που βασίζονται στο παράθυρο Hann. Η γραφική παράσταση που ακολουθεί δείχνει το φασματικό μέγεθος του σήματος εισόδου (μπλε) και του φιλτραρισμένου σήματος (κόκκινο). Το φιλτραρισμένο σήμα έχει μικρότερο μέγεθος συχνοτήτων που βρίσκονται στην ζώνη διακοπής (stopband). Αυτό σημαίνει ότι έχουν απορριφθεί επιτυχώς από το φίλτρο. Οι συχνότητες όμως που βρίσκονται στη ζώνη διέλευσης (passband) έχουν είναι σχεδόν ίδιες με το αρχικό σήμα φαινόμενο που μας αποδεικνύει ότι αυτές έχουν διατηρηθεί.




Ακολουθεί στην επόμενη σελίδα ολόκληρος ο κώδικας MATLAB που χρησιμοποιήθηκε για την άσκηση 2.

```
%PROJECT PSES - VALSAMIS GEORGIOS (03259)-PISXOS VASILEIOS (03175)

% Define the lengths
L_values = [16, 32, 64, 128];

%erothma B1
for L = L_values

    %We give n the asked values and then apply it to the signal x[n]
    n = -L/2:1:(L/2)-1;

    x = -40*sinc(n/2) + cos(pi*n/16) + cos(pi*n/8) + cos(pi*n/4);
    x(L/2+1) = x(L/2+1) + 80;


    % Plot the zero-padded or truncated signal
    figure;
    stem(n, x);
    title(['Signal x[n] for L = ', num2str(L)]);
    xlabel('n');
    ylabel('Amplitude');
end
```

```matlab
%erothma B2

for L = L_values
    N = 8192;
    n = -L/2:1:L/2-1;
    x = -40*sinc(n/2) + cos(pi*n/16) + cos(pi*n/8) + cos(pi*n/4);
    x(L/2+1) = x(L/2+1) + 80;

    %DFT for N = L and N=8192
    X_L = fft(x, L);
    X_N = fft(x, N);

    %these are the frequency values for each L
    f_L = (0:1:L-1);
    f_N = (0:1/(N/2):1);

    %DFT for N = L
    figure;
    stem(f_L, abs(X_L));
    title(['Magnitude of DFT for N = L = ', num2str(L)]);
    xlabel('Sample number)');
    ylabel('|X_L[k]|');

    figure;
    plot(f_N, abs(X_N(1:N/2+1)));
    xlabel("Frequency (x \pi)");
    ylabel("DFT Magnitude");
    title("DFT Magnitude for L = 8192");


end

%erothma B4

L_new = [64, 128];

for L = L_new
    N = 8192;
    n = -L/2:1:L/2-1;
    x = -40*sinc(n/2) + cos(pi*n/16) + cos(pi*n/8) + cos(pi*n/4);
    x(L/2+1) = x(L/2+1) +80;
    X_N = fft(x,N);

    %Hamming window
    w = 0.54-0.46*cos(2*pi*(n+L/2)/(L-1));
    x2 = x.*w;
    X_hamming = fft(x2,N);

    figure;
    plot(0:1/(N/2):1,abs(X_N(1:N/2+1)),'r');
    hold on;
    plot(0:1/(N/2):1,abs(X_hamming(1:N/2+1)),'b');
    xlabel('Normalized frequency (\times\pi rad/sample)');
    ylabel('Magnitude');
    title('Comparison of Rectangular and Hamming Windows, L = ',
num2str(L));
    legend('Rectangular Window','Hamming Window',
'Location','northwest');
    hold off;
end
```

```
%erothma B5
N = 8192;
%apo askhsh 1
p_a = 0.20 * pi;
p_b = 0.40 * pi;
s_a = 0.60 * pi;
s_b = 0.80 * pi;

dp = p_b - p_a;
ds = s_b - s_a;
d = max([dp, ds]);

%tha xrhsimopoihsoume to parathyro Hann apo thn proth aslhsh gia auto to
erothma
M_hann = ceil(8*pi/d);
W_hann = hann(M_hann + 1)';

f1 = ((p_a+p_b)/2)/pi;
f2 = ((s_a+s_b)/2)/pi;

n = 0:1:M_hann;
k = n - M_hann/2;
hHann = (-f1*sinc(f1*k)+f2*sinc(f2*k)).*W_hann;

filter_output = conv(x2,hHann);
magni_spec_output = abs(fft(filter_output, N));

figure;
hold on;
plot(0:1/(N/2):1, abs(X_hamming(1:N/2+1)), 'b');
plot(0:1/(N/2):1, magni_spec_output(1:N/2+1), 'r');
hold off;
xlabel('Normalized Frequency (\pi rad/sample)');
ylabel('Magnitude');
title('Spectral Magnitude Diagram of Input vs Output Signal');
legend('Input', 'Output', 'Location','northwest');
```