Σήματα και Συστήματα – Εργασία MATLAB

Επώνυμο:	**Εμμανουηλίδης**
Όνομα:	**Εμμανουήλ**
ΑΜ:	**03119435**
Εξάμηνο:	**3ο**

Άσκηση 1 – Σχεδίαση Φίλτρων:

1.1 Σχεδίαση φίλτρων ηχούς και αντήχησης (Echo, Reverb):

α) Το φίλτρο που χρησιμοποιείται για την προσομοίωση του echo – effect περιγράφεται από την ακόλουθη εξίσωση:

$$y[n] = cx[n] + (1-c)x[n-P]$$

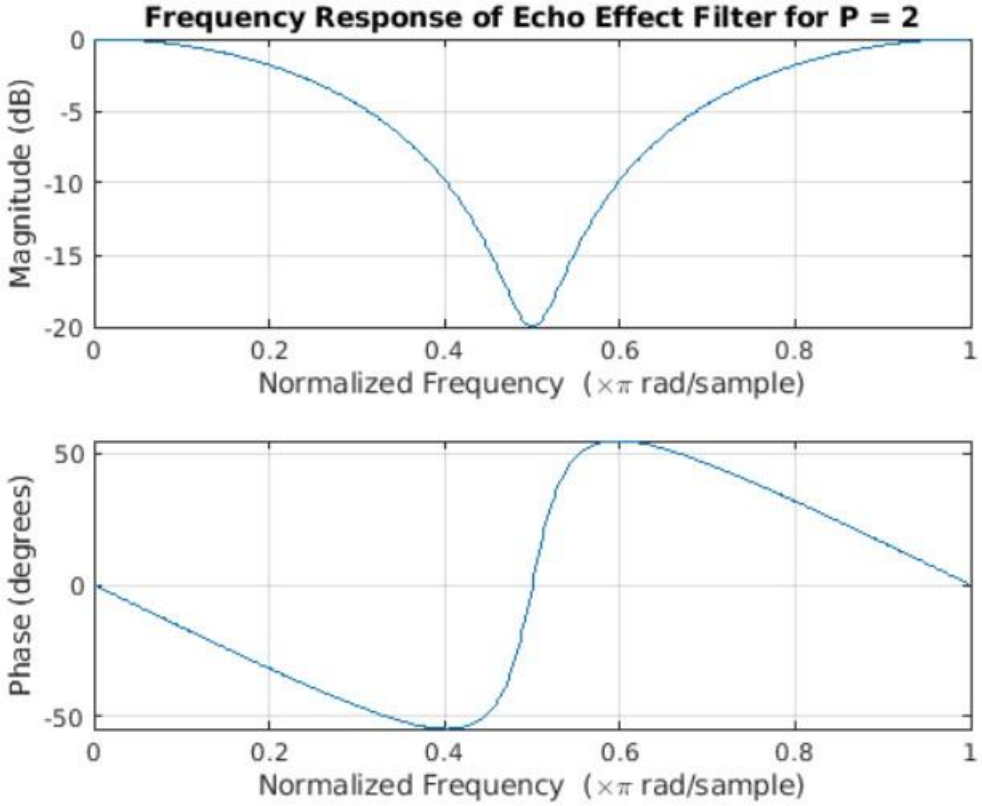
Για c = 0.55 και για P = 2 και P = 5 η εξίσωση γίνεται αντίστοιχα:

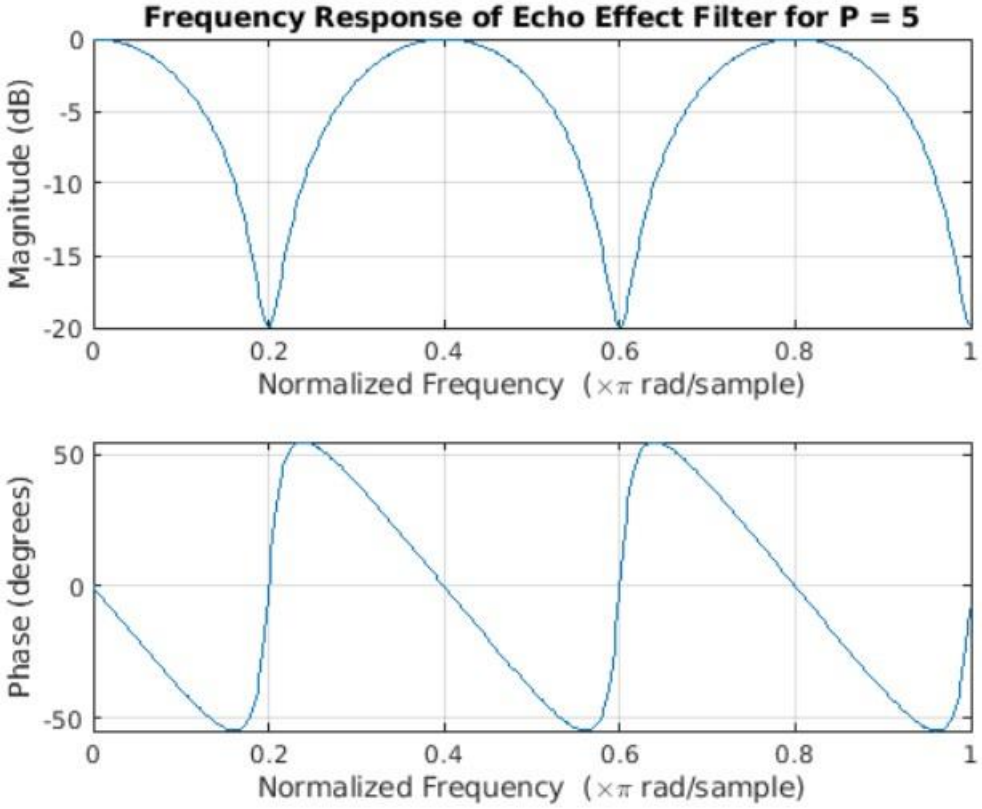
- $y[n] = 0.55x[n] + 0.45x[n-2]$
- $y[n] = 0.55x[n] + 0.45x[n-5]$

Από παρατήρηση των εξισώσεων και από τη σχέση (1) (της εκφώνησης) προκύπτουν:

- a = [a_i] = 1
- b2 = [b_i] = [0.55 0 0.45] για P = 2 και
- b5 = [b_i] = [0.55 0 0 0 0 0.45] για p = 5

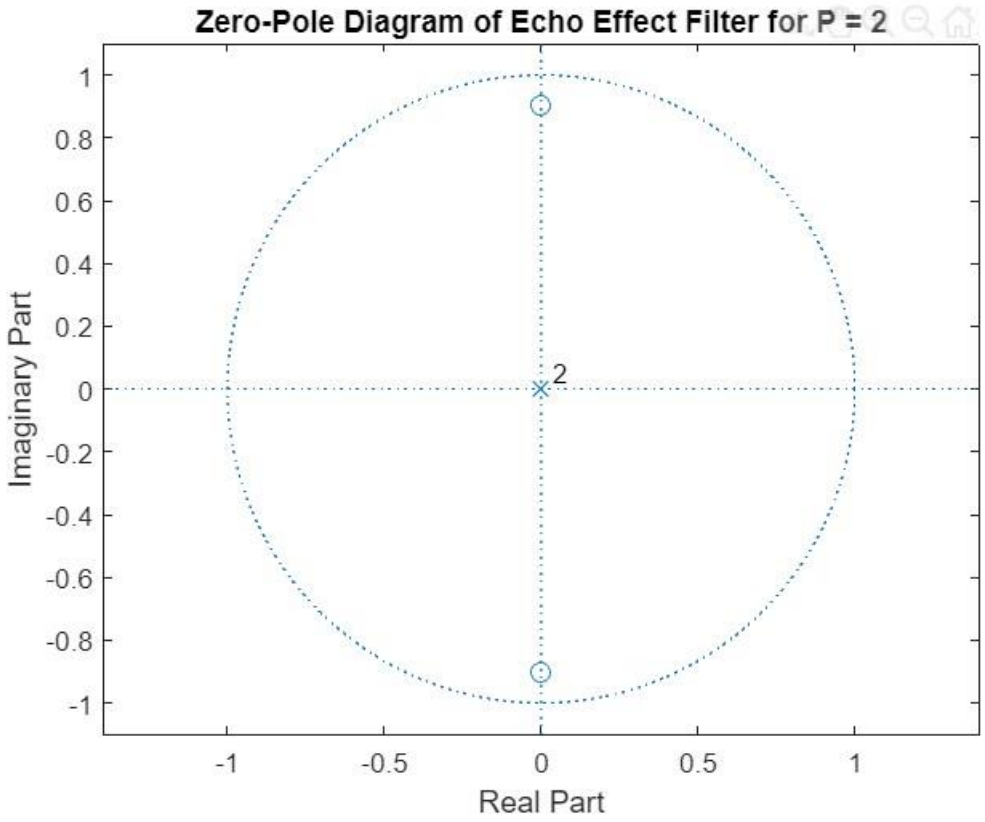
β) Χρησιμοποιώντας την εντολή **freqz()** με παραμέτρους τα a & b2 (για P = 2) και a & b5 (για P = 5) παίρνουμε την παρακάτω απόκριση πλάτους και φάσης για κάθε φίλτρο:

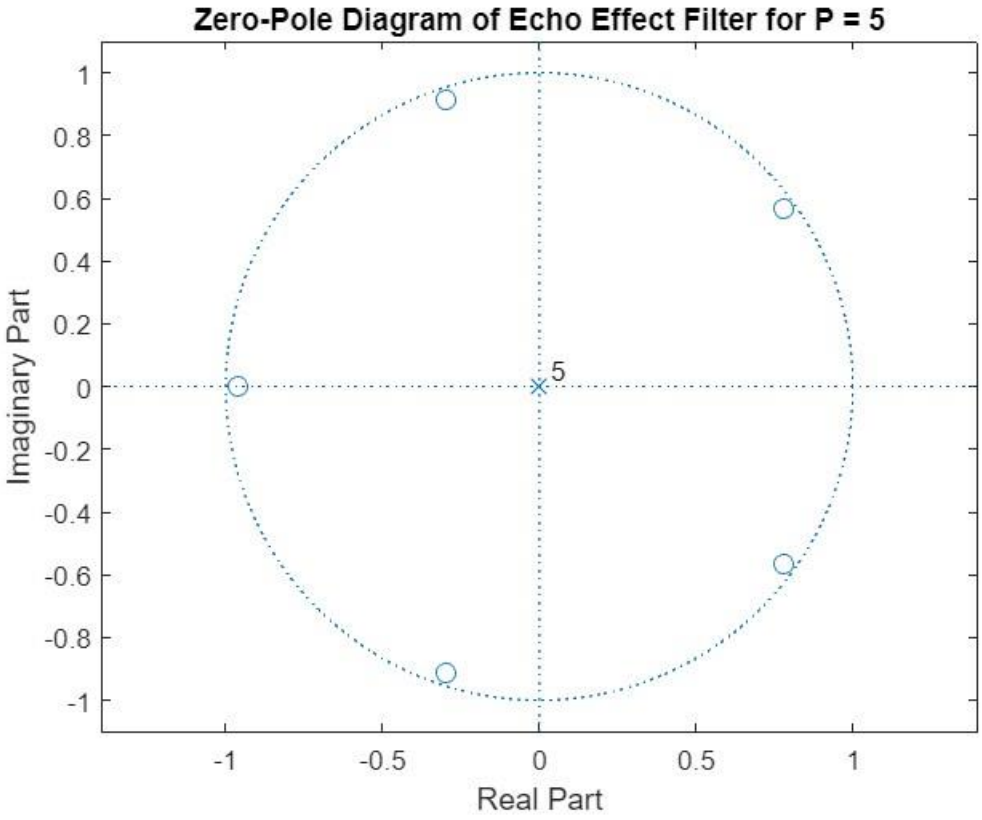

Frequency Response of Echo Effect Filter for P = 2
Magnitude (dB)
0
-5
-10
-15
-20
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)
Phase (degrees)
50
0
-50
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)


Frequency Response of Echo Effect Filter for P = 5
Magnitude (dB)
0
-5
-10
-15
-20
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)
Phase (degrees)
50
0
-50
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)

Παρατηρώντας τα παραπάνω διαγράμματα, βλέπουμε πως η απόκριση πλάτους είναι όμοια και στις 2 πειρπτώσεις, με μόνη διαφορά τη συχνότητα η οποία για P = 5 είναι 2.5 φορές μεγαλύτερη από τη συχνότητα για P = 2, ενώ οι μέγιστες και ελάχιστες τιμές παραμένουν ίδιες. Τα ίδια ακριβώς ισχύουν και για την απόκριση φάσης.

γ) Με χρήση της εντολής **zplane()** σχεδιάζουμε τα διαγράμματα πόλων και μηδενικών των δύο φίλτρων:





Για P = 2 έχουμε έναν διπλό πόλο στο (0, 0) και 2 μηδενικά στα σημεία (0, 0.9045) και (0, -0.9045), δηλαδή ζεύγος συζηγών μιγαδικών μηδενικών.

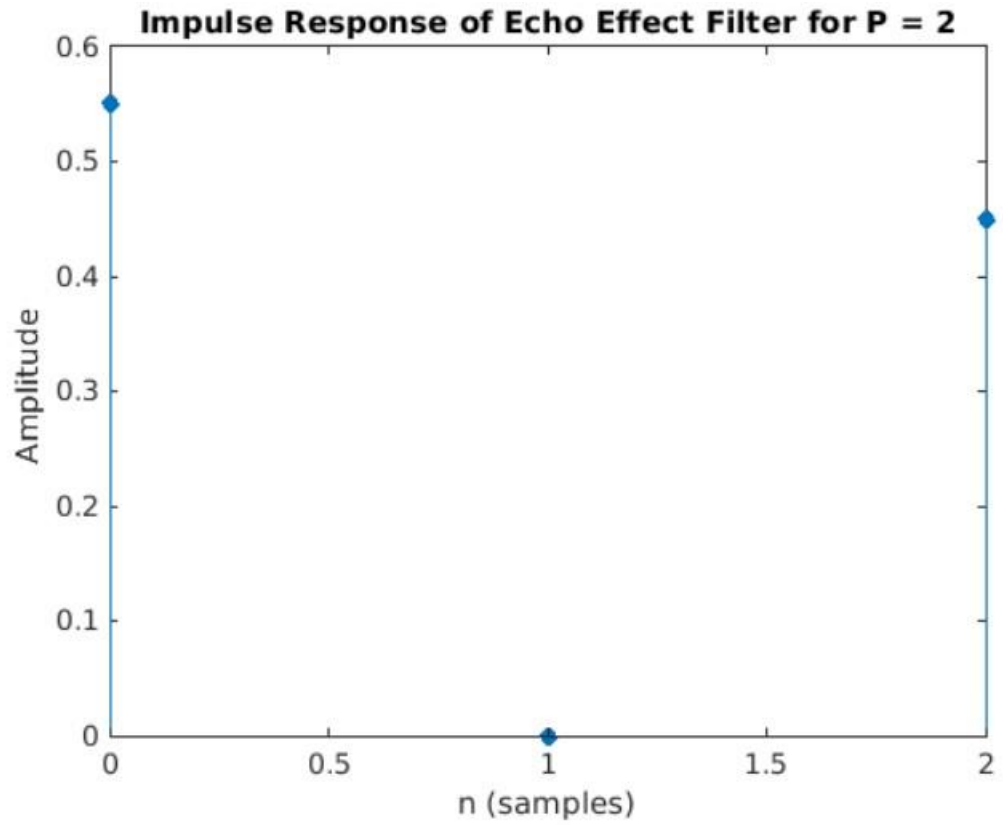
Για P = 5 έχουμε 5 πόλους (= έναν πενταπλό) στο (0, 0) ένα μηδενικό στο σημείο (-0.9607, 0) και 2 ζεύγη συζηγών μιγαδικών μηδενικών στα σημεία:

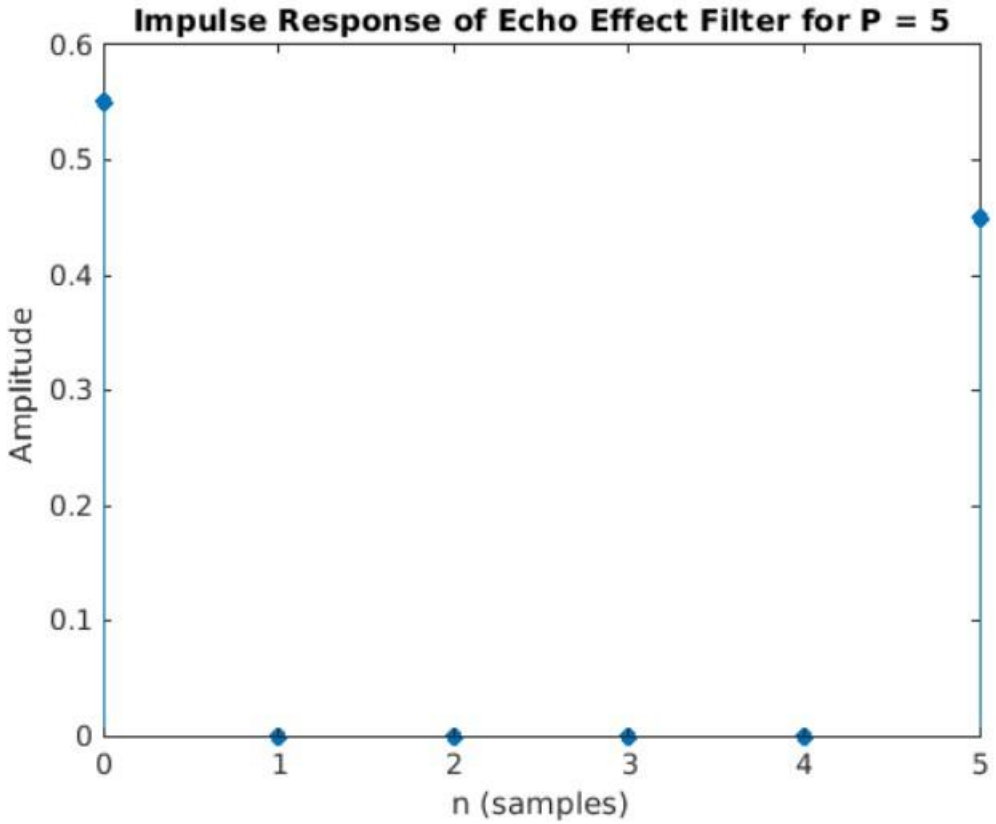
- (-0.2969, 0.9136) , (-0.2969, -0.9136)
- (0.7772, 0.5647) , (0.7772, -0.5647)

Εφόσον, και στα δύο φίλτρα οι πόλοι βρίσκονται στο (0, 0), αυτό σημαίνει πως η περιοχή σύγκλισης είναι όλο το R , το οποίο περιέχει τον μοναδιαίο κύκλο, οπότε το σύστημα είναι ευσταθές.

Για τη μεταφορά των φίλτρων σε μορφή πόλων – μηδενικών, χρησιμοποιήθηκε η εντολή **tf2zpk()** αντί της **tf2zp()** γιατί η τελευταία δε λειτουργεί καλά με αρνητικούς εκθέτες.

δ) Με χρήση της εντολής **impz()** προκύπτει η κρουστική απόκριση κάθε φίλτρου:





Εφόσον, a = [1 0 0 .. 0], και στα δύο φίλτρα, η κρουστική απόκριση θα είναι ίδια με το διάνυσμα b, πράγμα που φαίνεται και στα παραπάνω διαγράμματα.

ε) Αρχικά, με την εντολή **impz(b2, a)** υπολογίζουμε την κρουστική απόκριση h του αρχικού φίλτρου. Γνωρίζουμε (από την εκφώνηση) πως η δημιουργία ενός φίλτρου αντήχησης βαθμού 3 μπορεί να επιτευχθεί με χρήση 3 διαδοχικών (σε σειρά) φίλτρων δημιουργίας ηχούς. Έτσι, κάνοντας 3 φορές συνέλιξη, την απόκριση του φίλτρου ηχούς, με τον εαυτό της , παίρνουμε την κρουστική απόκριση του φίλτρου αντήχησης (h_reverb = h * h * h).

Προκύπτει: h_reverb = [0.1664 0 0.4084 0 0.3341 0 0.0911]'

Στη συνέχεια, αξιοποιώντας τις σχέσεις:

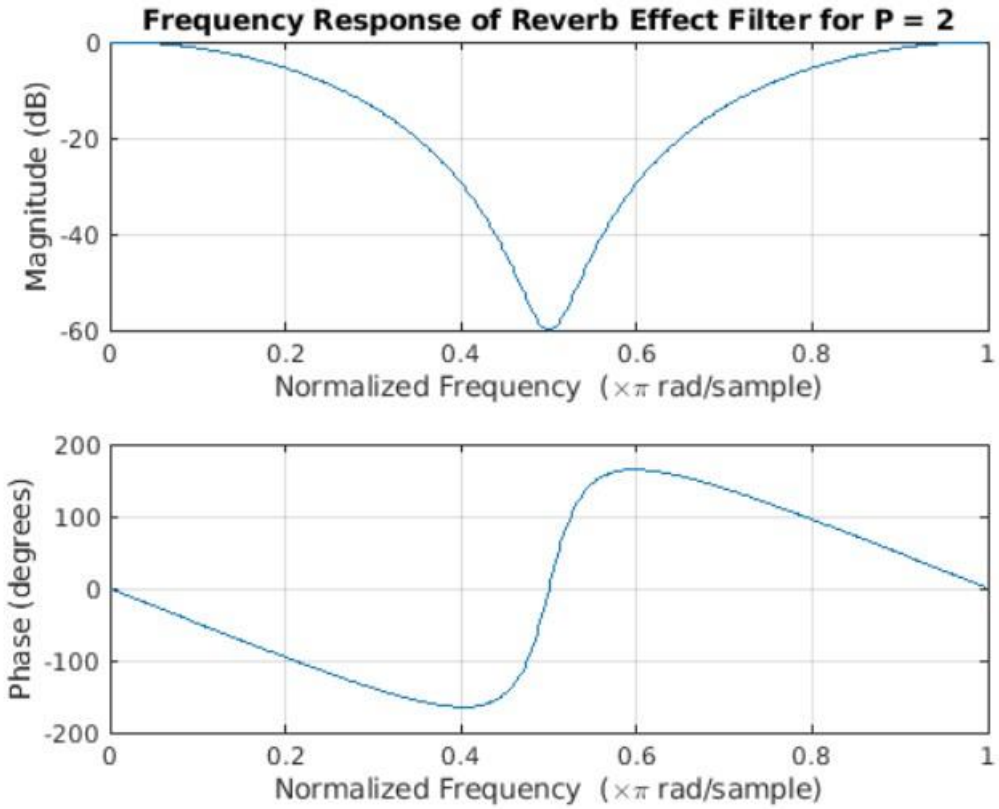
$$ai' = T^{-1}_{h_{reverb}} \cdot h'_{reverb}$$

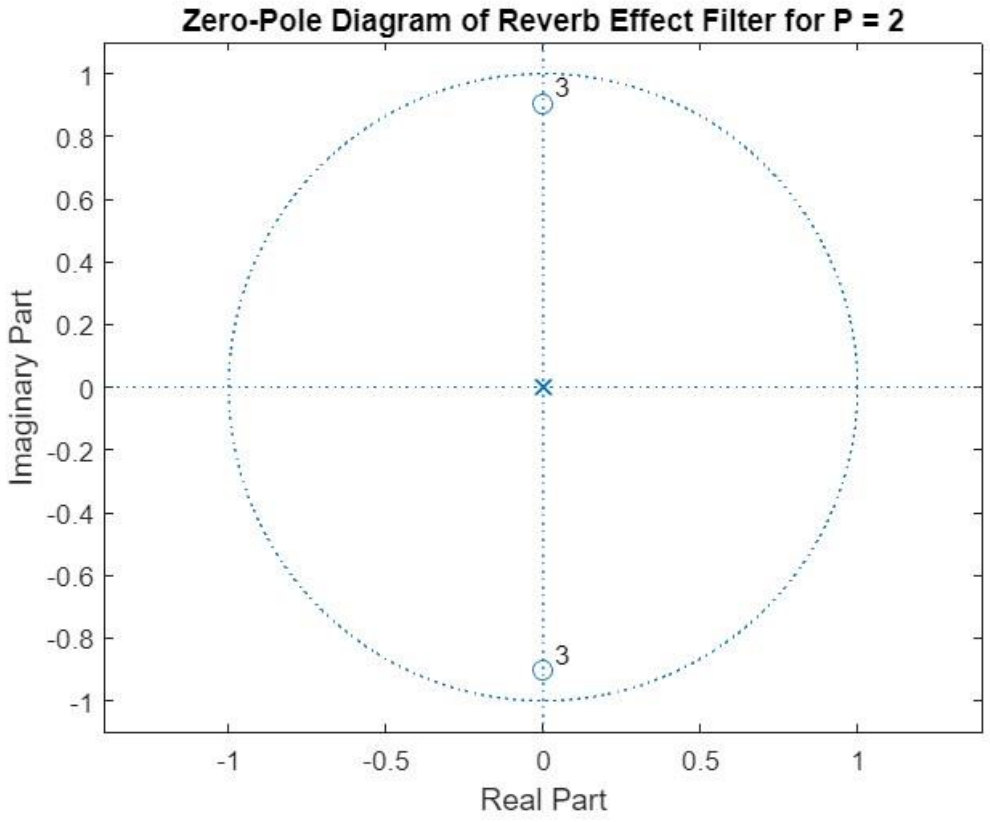
$$bi' = h'_{reverb} + T_{ho} \cdot ai' ,$$

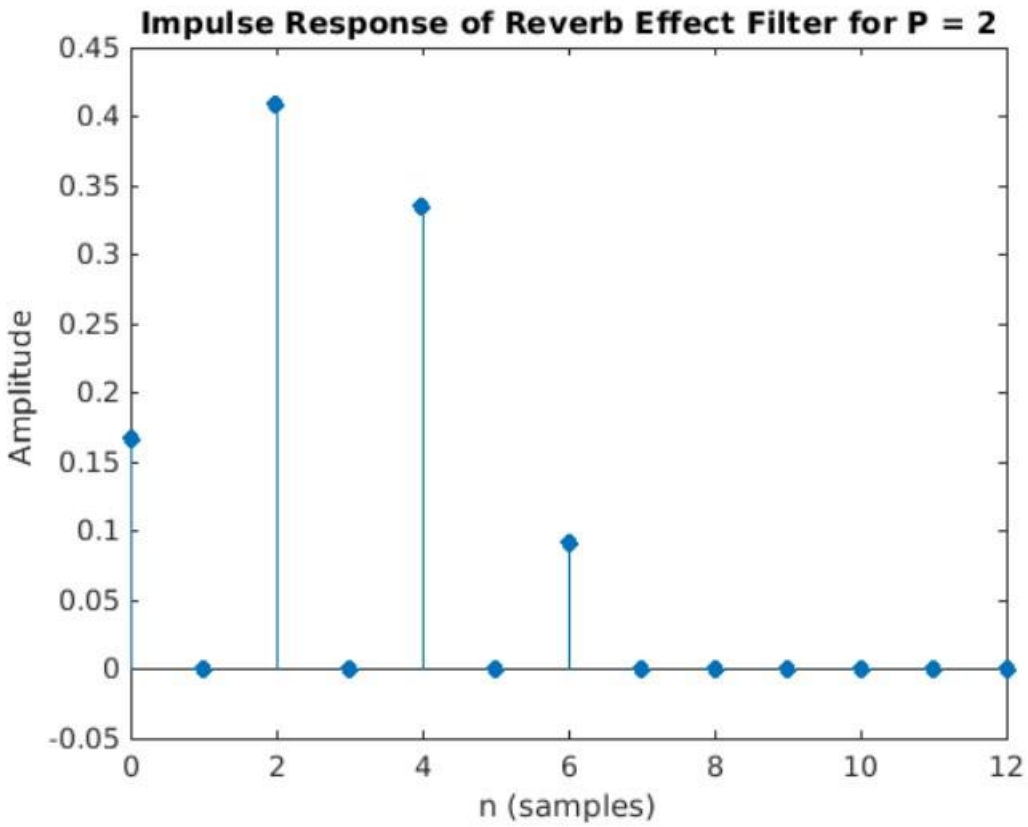
υπολογίζουμε τα διανύσματα ai & bi για το φίλτρο αντήχησης και παίρνουμε:

ai = 1 ⇒ bi = h_reverb = [0.1664 0 0.4084 0 0.3341 0 0.0911]'

Τέλος, με βάση τα διανύσματα ai, bi που υπολογίσαμε, επαναλαμβάνουμε τα ερωτήματα β) – δ) παίρνοντας τα εξής διαγράμματα:


Frequency Response of Reverb Effect Filter for P = 2
Magnitude (dB)
0
-20
-40
-60
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)
Phase (degrees)
200
100
0
-100
-200
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)


Zero-Pole Diagram of Reverb Effect Filter for P = 2
Imaginary Part
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
3
3
-1
-0.5
0
0.5
1
Real Part



Τα μηδενικά και οι πόλοι τώρα έχουν ως εξής:

Μηδενικά:

1 τριπλό μηδενικό στο σημείο (0, 0.9045) και ένα τριπλό στο (0, -0.9045) , (δηλαδή συζυγή).

Πόλοι:

(-0.0011, 0) , (0.0011, 0) , (-0.0005, 0.0009) , (-0.0005, -0.0009) ,

(0.0005 , 0.0009) και (0.0005, -0.0009)

στ) Γνωρίζουμε ότι αν h_reverb η κρουστική απόκριση του φίλτρου αντήχησης και h_dereverb η κρουστική απόκριση του φίλτρου απαλοιφής της αντήχησης τότε θα ισχύει:

$$h_{reverb} * h_{dereverb} = \delta[n]$$

Κάνοντας μετασχηματισμό Fourier στην παραπάνω εξίσωση παίρνουμε:

$$H_{reverb}H_{dereverb} = 1 \Rightarrow H_{dereverb} = \frac{1}{H_{reverb}}$$

Σύμφωνα με τα παραπάνω:

- Με χρήση της συνάρτησης **fft()** βρίσκουμε την H_reverb
- Στη συνέχεια με βάση την παραπάνω σχέση υπολογίζουμε την H_dereverb
- Έπειτα, με χρήση της **ifft()** βρίσκουμε την h_dereverb
- Τέλος, με χρήση των τύπων:

$$ai'_{dereverb} = T^{-1}_{h_{dereverb}} \cdot h'_{dereverb}$$
$$bi'_{dereverb} = h'_{dereverb} + T_{ho} \cdot ai'_{dereverb},$$

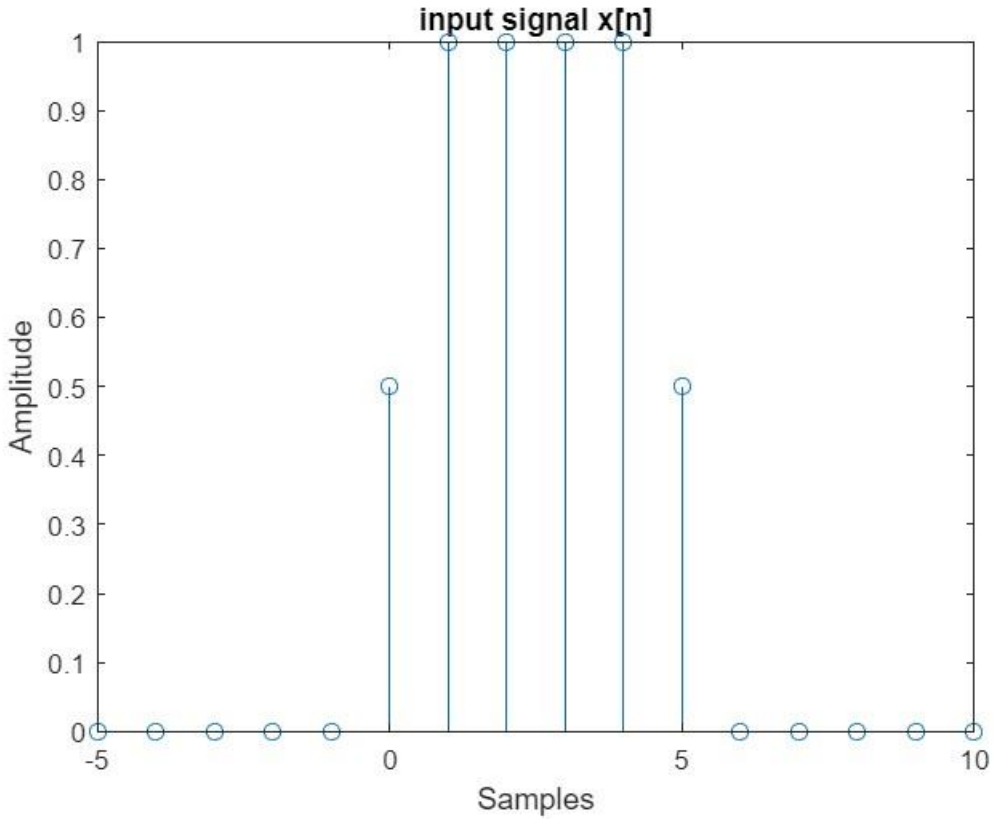
βρίσκουμε τα διανύσματα ai_dereverb & bi_dereverb για το φίλτρο απαλοιφής της αντήχησης.

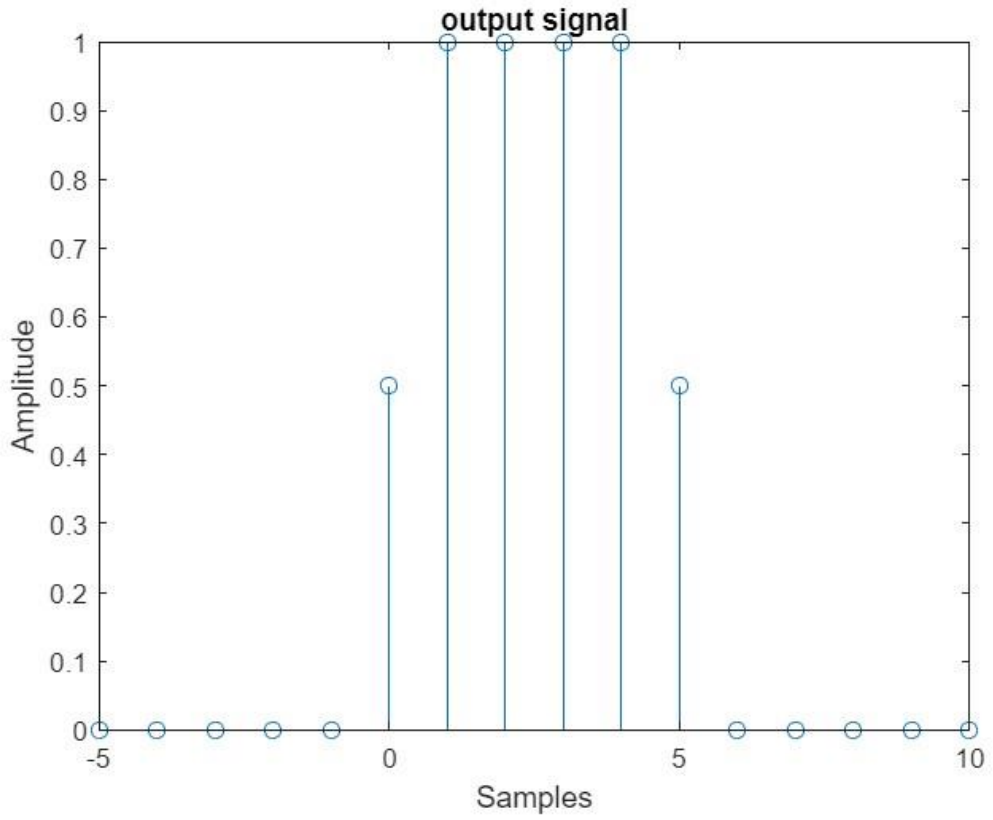
<u>Σημείωση</u>: Για μεγαλύτερη ακρίβεια και ορθότητα, στις συναρτήσεις **fft()** και **ifft()** χρησιμοποιήθηκε αριθμός δειγμάτων = 512.

Για επαλήθευση των αποτελεσμάτων χρησιμοποιούμε το σήμα:

$x[n] = u[n] - u[n-5]$ το οποίο υλοποιήθηκε με χρήση της συνάρτησης **heaviside()**.

Συγκεκριμένα, το σήμα x[n] πέρασε πρώτα μέσα από το φίλτρο αντήχησης και έπειτα από το φίλτρο απαλοιφής της αντήχησης και τα διαγράμματα που προέκυψαν είναι τα ακόλουθα:


input signal x[n]
Amplitude
Samples
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-5
0
5
10


output signal
Amplitude
Samples
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-5
0
5
10

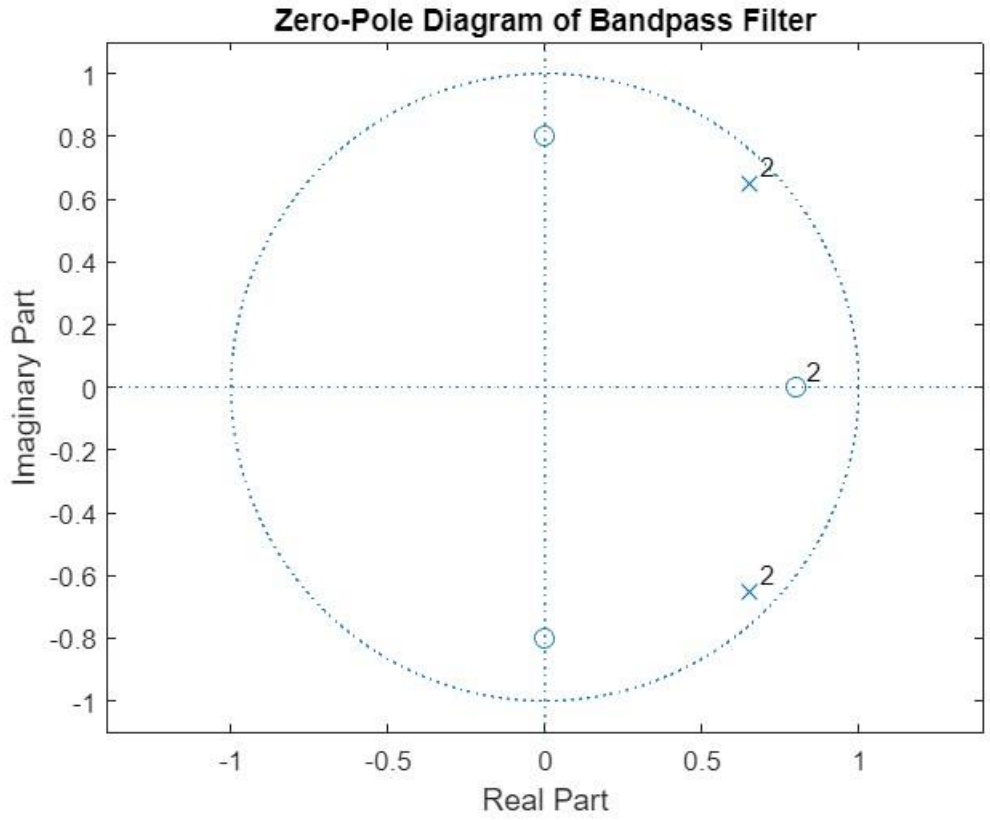
Παρατηρούμε πως η αρχική είσοδος x[n] ταυτίζεται με την εξόδο της εν σειρά σύνδεσης των δύο φίλτρων, όπως ήταν αναμενόμενο. Συνεπώς, το φίλτρο απαλοιφής της αντήχησης που κατασκευάσαμε είναι σωστό.

1.2 Σχεδίαση Ζωνοπερατών Φίλτρων:

α) Σύμφωνα με την εκφώνηση οι πόλοι και τα μηδενικά θα οριστούν ως εξής:

```
p1 = [(0.65 + 0.65i) (0.65 - 0.65i) (0.65 + 0.65i) (0.65 - 0.65i)]';
z1 = [0.8 0.8 0.8i -0.8i]';
```

Με χρήση της **zplane()** παίρνουμε το διάγραμμα πόλων – μηδενικών:

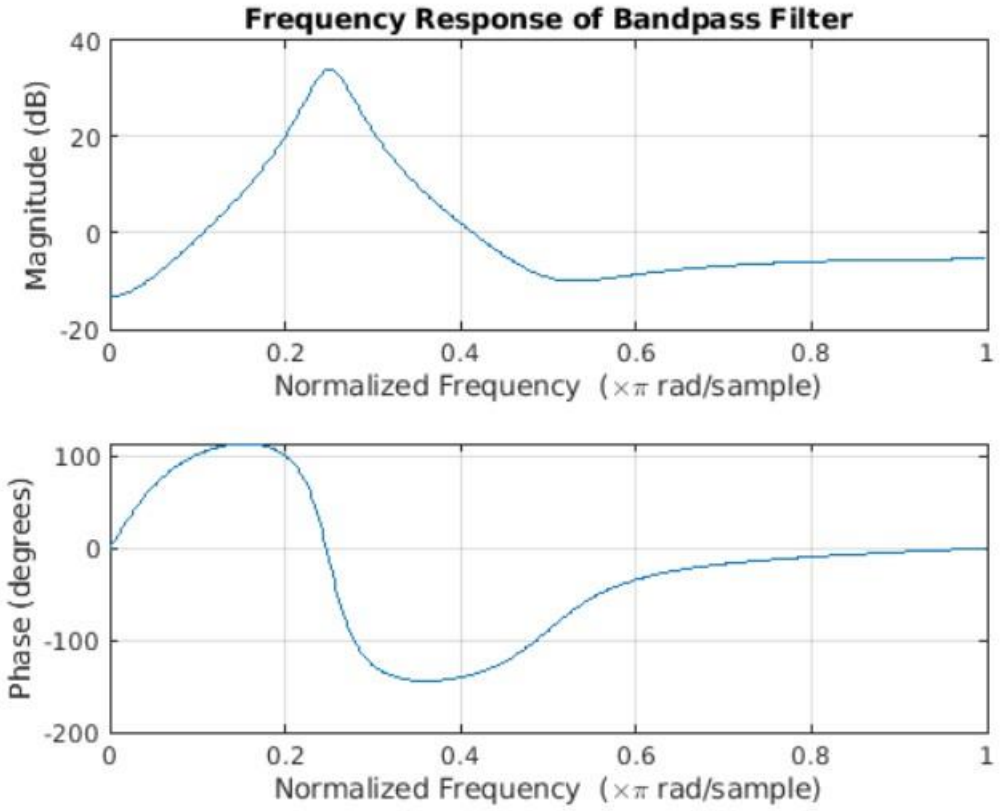


Για την εύρεση των διανυσμάτων συντελεστών a1, b χρησιμοποιήθηκε η συνάρτηση **zp2tf()** (στην οποία θέτουμε k = 1) και πήραμε:

b = [1 -1.6 1.28 -1.024 0.4096]

a1 = [1 -2.6 3.38 -2.197 0.714]

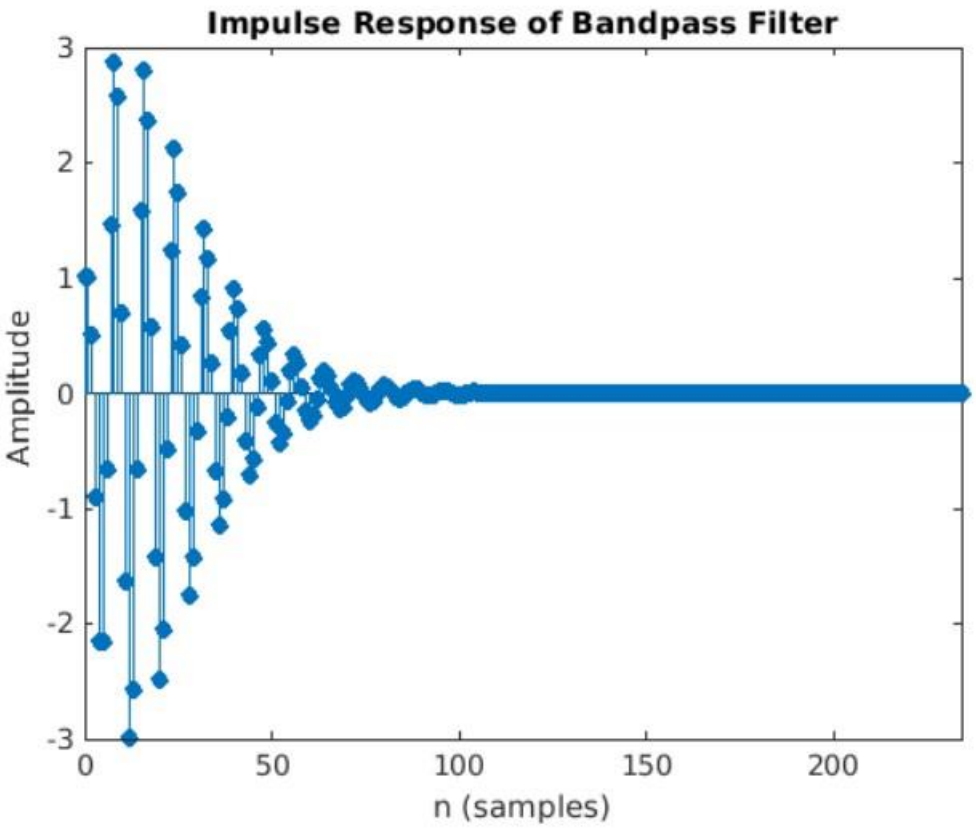
β) Όπως και στο ερώτημα 1.1 β) χρησιμοποιώντας τη συνάρτηση **freqz()** παίρνουμε την απόκριση πλάτους και φάσης του φίλτρου:

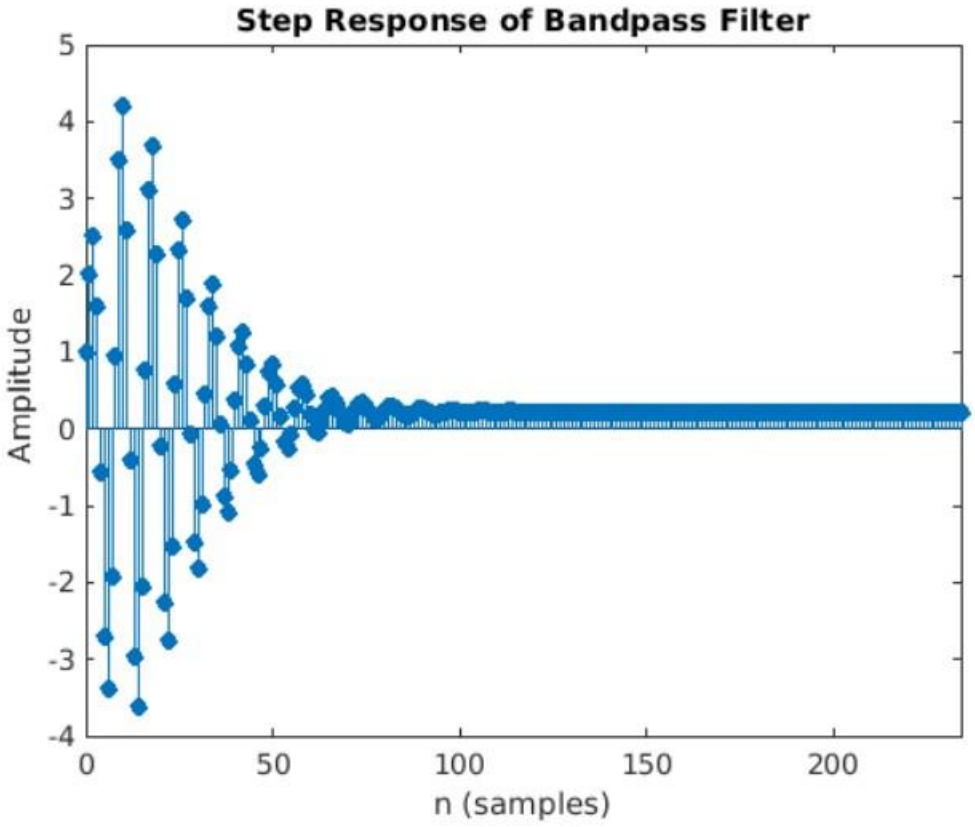


Το διάγραμμα αντιστοιχεί σε ένα bandpass filter (ζωνοπερατό φίλτρο) , όπως αναμενόταν.

Επειδή το σύστημα είναι αιτιατό και όλοι πόλοι βρίσκονται εντός του μοναδιαίου κύκλου , το σύστημα είναι ευσταθές και όσο αυξάνεται το κέρδος k (που θέσαμε = 1), αυξάνεται και το πλάτος της απόκρισης συχνότητας.

γ) Σχεδιάζουμε την κρουστική απόκριση του συστήματος με χρήση της **impz()** καθώς και τη βηματική απόκριση (με χρήση της **stepz()**) , χρησιμοποιώντας τα a1, b που υπολογίσαμε νωρίτερα:

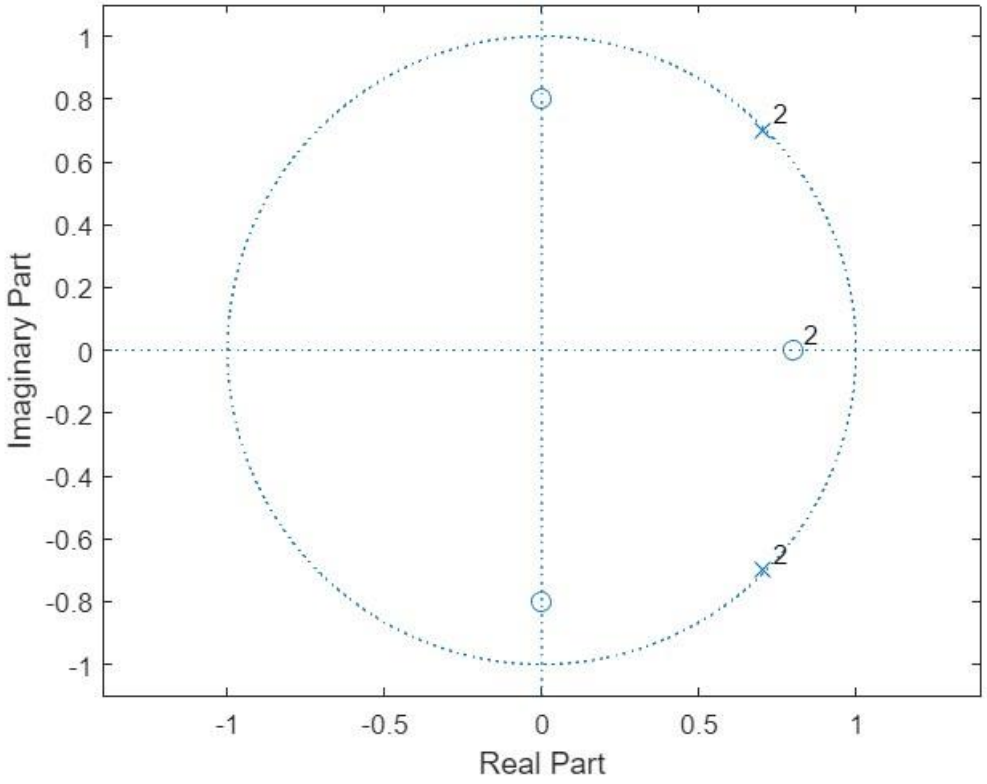

Impulse Response of Bandpass Filter
Amplitude
n (samples)
3
2
1
0
-1
-2
-3
0
50
100
150
200

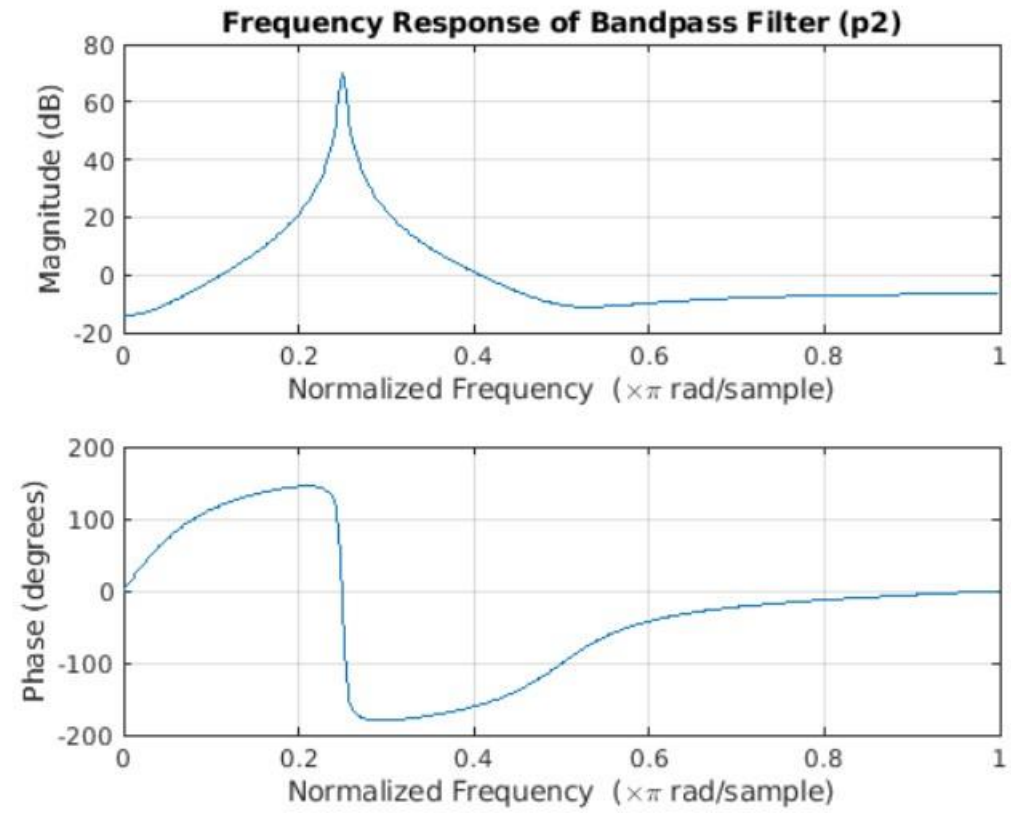

Step Response of Bandpass Filter
Amplitude
n (samples)
5
4
3
2
1
0
-1
-2
-3
-4
0
50
100
150
200

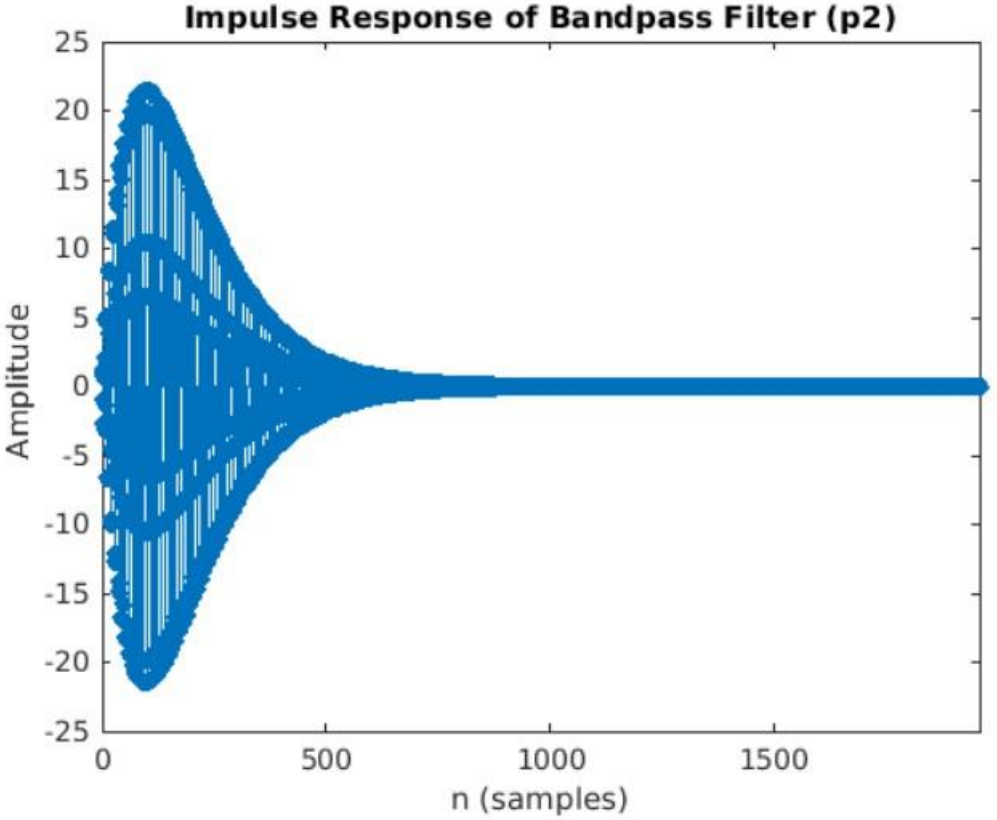
δ) Σύμφωνα με την εκφώνηση, οι νέοι πόλοι θα οριστούν ως εξής:

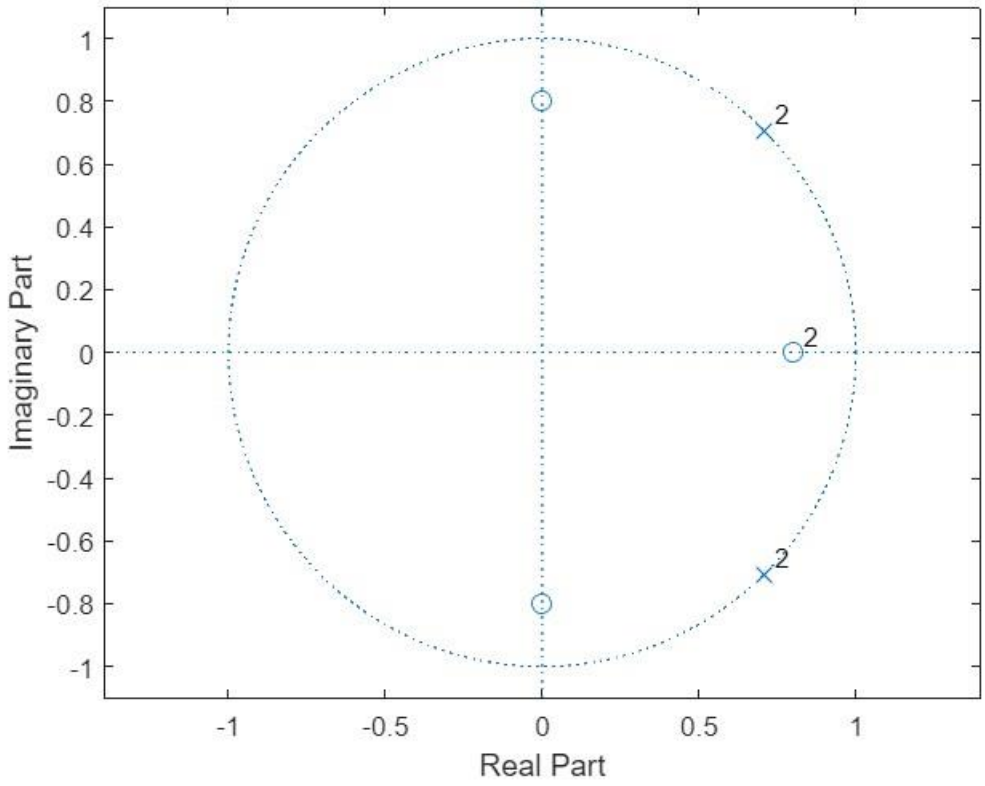
- p2 = [(0.7 + 0.7i) (0.7 - 0.7i) (0.7 + 0.7i) (0.7 - 0.7i)]';
- p3 = [(0.707 + 0.707i) (0.707 - 0.707i) (0.707 + 0.707i) (0.707 - 0.707i)]';
- p4 = [(0.75 + 0.75i) (0.75 - 0.75i) (0.75 + 0.75i) (0.75 - 0.75i)]';

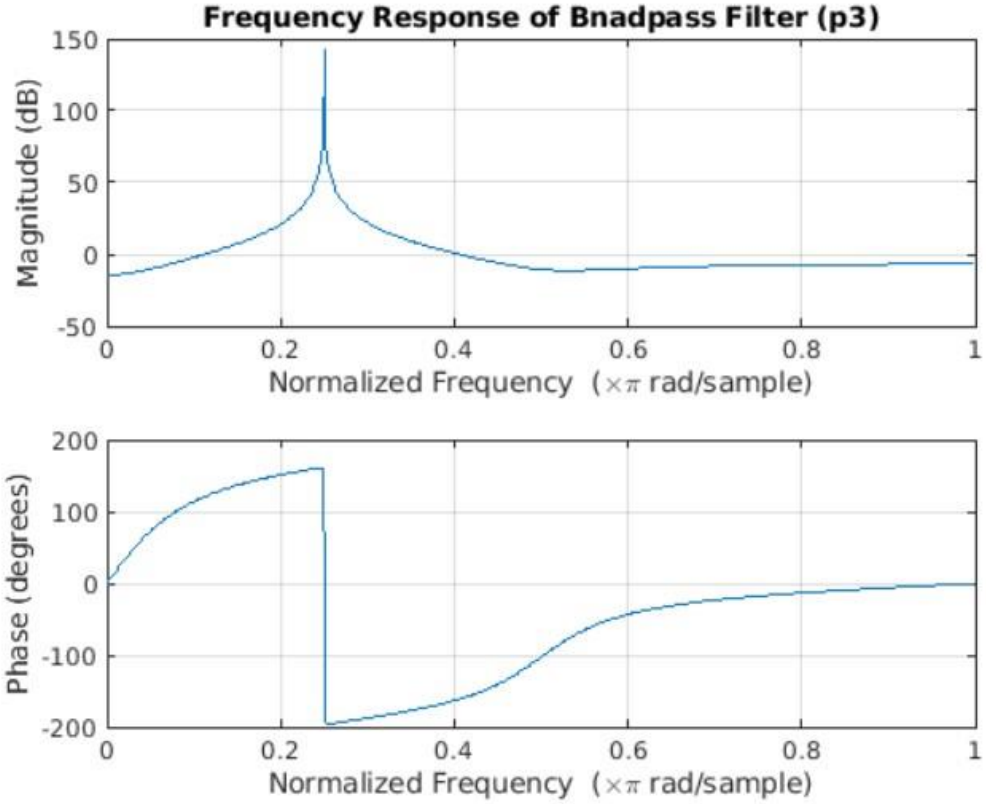
Χρησιμοποιώντας τα μηδενικά από το προηγούμενο ερώτημα, τους νέους πόλους και τη συνάρτηση **zp2tf()**, βρίσκουμε τα διανύσματα a και b, ενώ με τις συναρτήσεις **zplane()**, **freqz()**, **impz()** παίρνουμε το διάγραμμα πόλων-μηδενικών, την κρουστική απόκριση και την απόκριση πλάτους για κάθε περίπτωση:

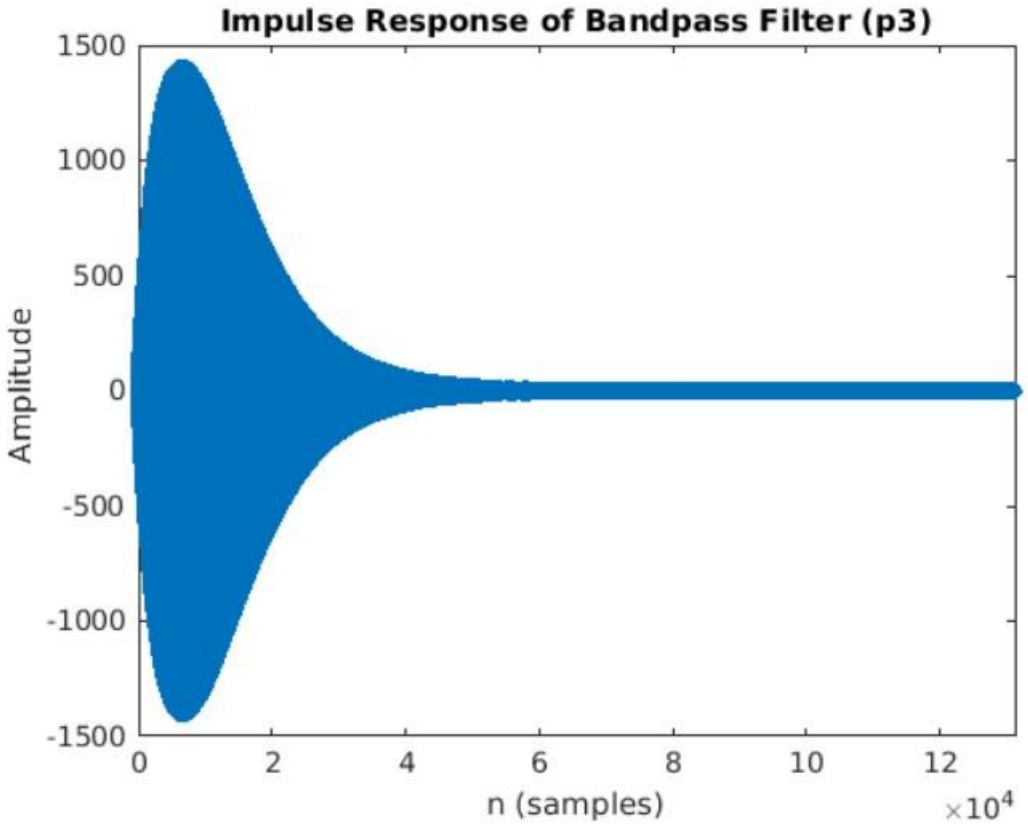


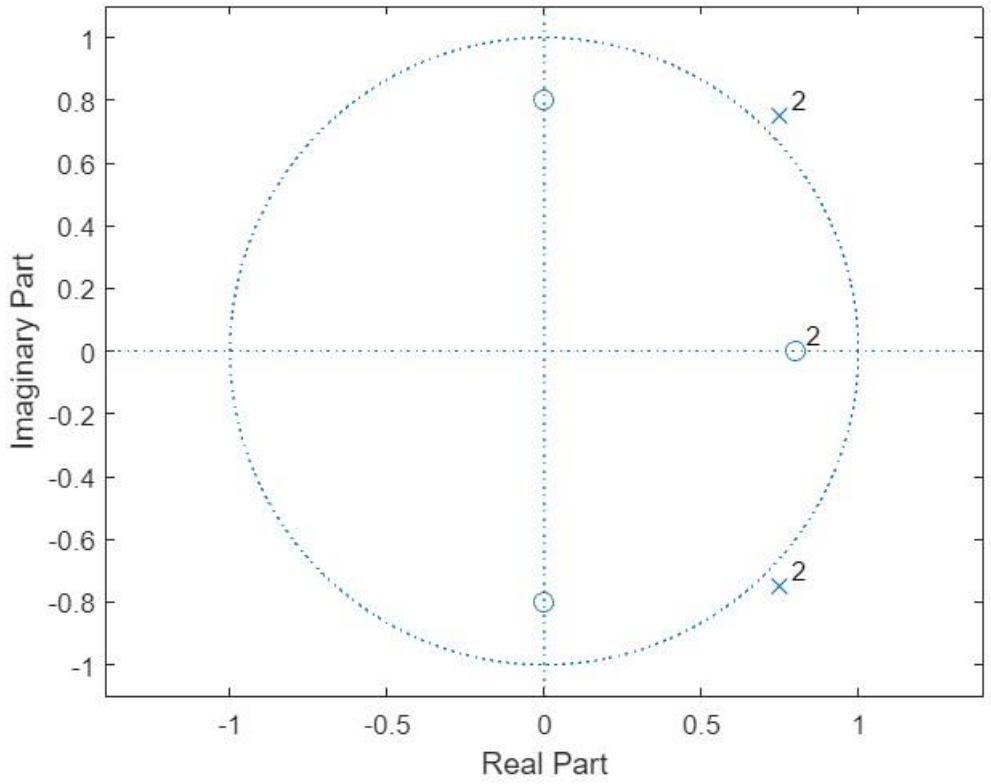

Frequency Response of Bandpass Filter (p2)
Magnitude (dB)
80
60
40
20
0
-20
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)
Phase (degrees)
200
100
0
-100
-200
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)

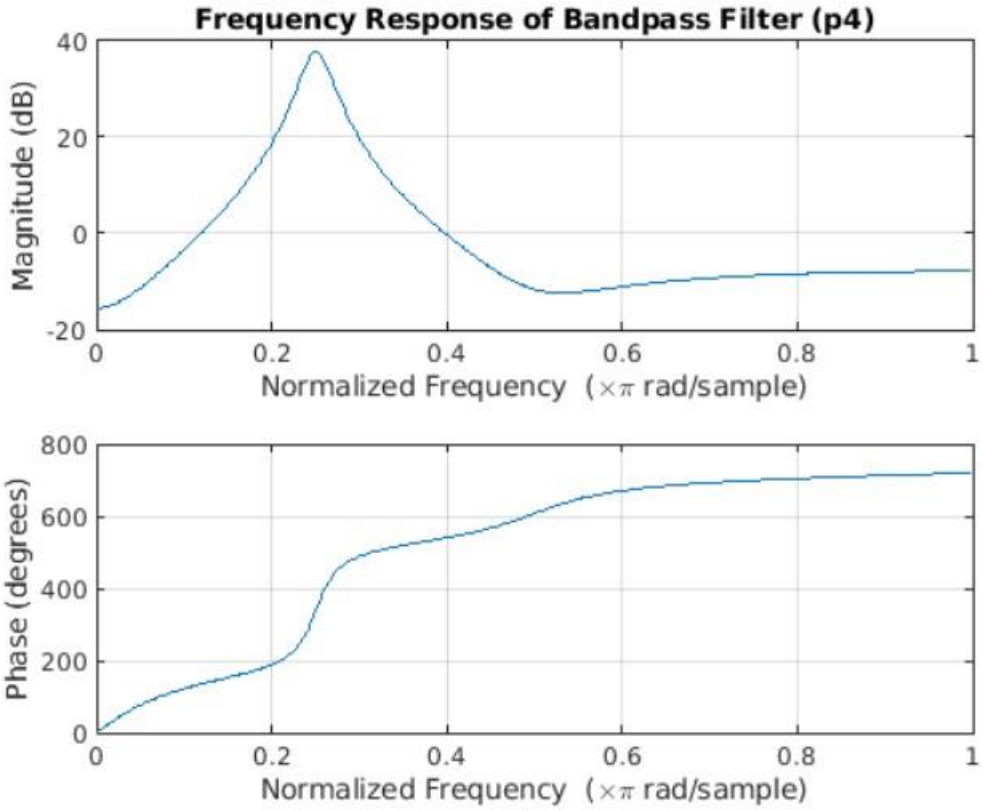

Impulse Response of Bandpass Filter (p2)
Amplitude
25
20
15
10
5
0
-5
-10
-15
-20
-25
0
500
1000
1500
n (samples)

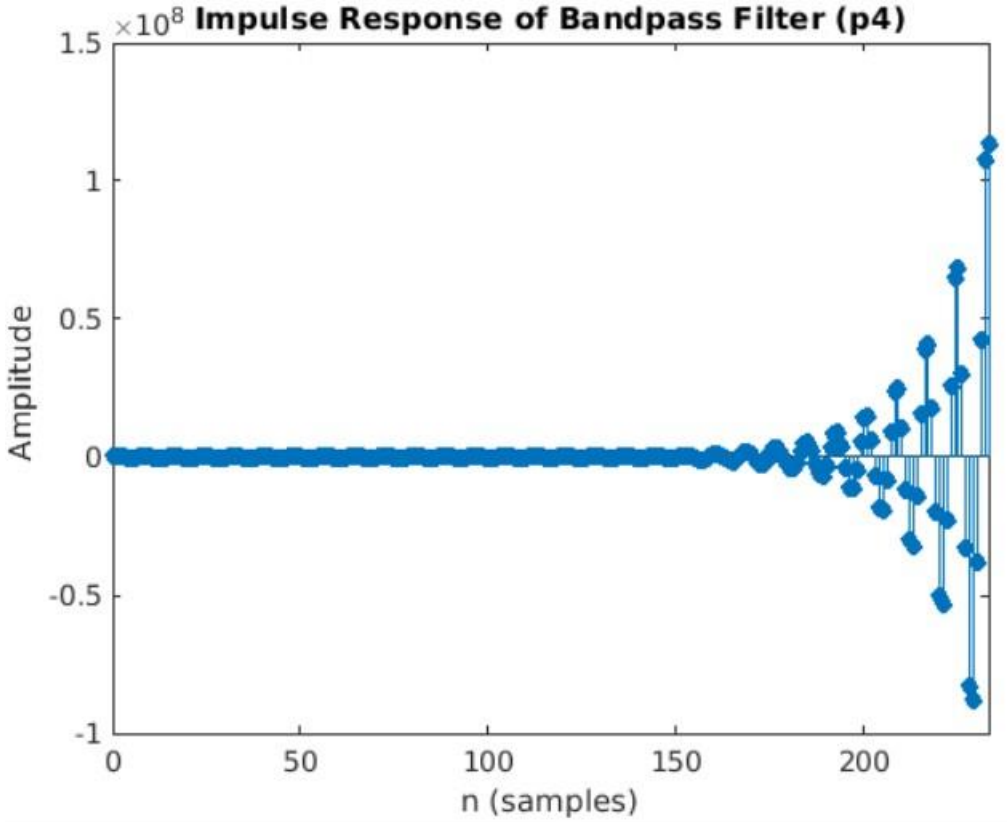

Imaginary Part
Real Part
2
2
2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.5
0
0.5
1


Frequency Response of Bnadpass Filter (p3)
Magnitude (dB)
Normalized Frequency (×π rad/sample)
Phase (degrees)
Normalized Frequency (×π rad/sample)
150
100
50
0
-50
200
100
0
-100
-200
0
0.2
0.4
0.6
0.8
1


Impulse Response of Bandpass Filter (p3)
Amplitude
n (samples)
×10^4
1500
1000
500
0
-500
-1000
-1500
0
2
4
6
8
10
12


Imaginary Part
Real Part
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.5
0
0.5
1
2
2
2


Frequency Response of Bandpass Filter (p4)
Magnitude (dB)
40
20
0
-20
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)
Phase (degrees)
800
600
400
200
0
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)

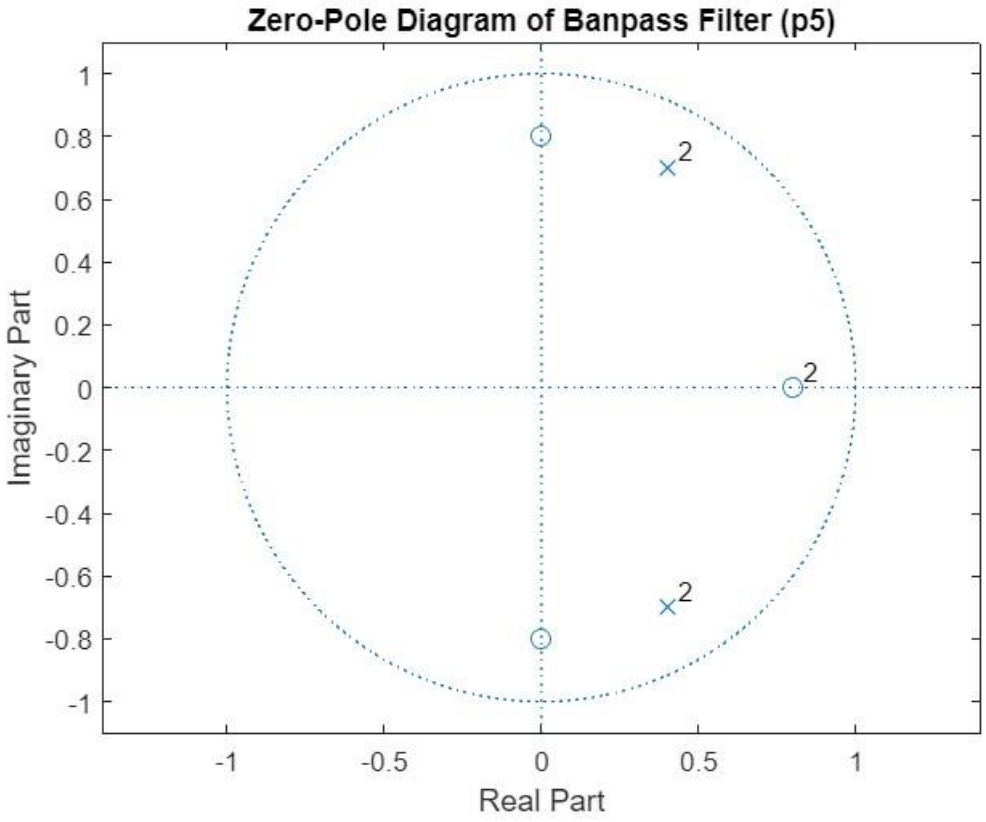

×10^8
Impulse Response of Bandpass Filter (p4)
Amplitude
1.5
1
0.5
0
-0.5
-1
0
50
100
150
200
n (samples)

Παρατηρώντας τα παραπάνω διαγράμματα βλέπουμε πως, καθώς οι πόλοι πλησιάζουν (τείνουν) στον μοναδιαίο κύκλο, τόσο αυξάνεται το πλάτος της κρουστικής απόκρισης και της απόκρισης συχνότητας, ενώ η απόκριση φάσης γίνεται όλο και πιο «απότομη». Όταν οι πόλοι βγουν έξω από τον μοναδιαίο κύκλο (0.75+-0.75i), τότε το πλάτος μείωνεται και τα διαγράμματα είναι τελείως διαφορετικά. Αυτό οφείλεται στο γεγονός ότι πλέον το σύστημα είναι ασταθές, κάτι που φαίνεται καλύτερα από το διάγραμμα απόκρισης φάσης.

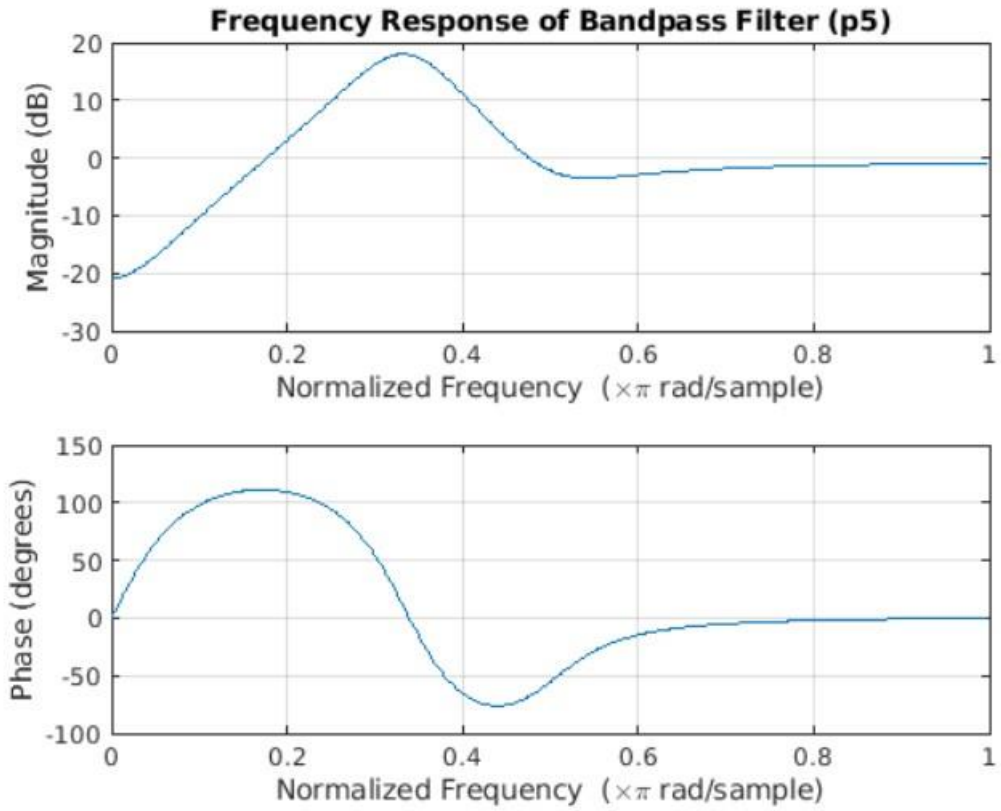
ε) Όμοια με πριν, ορίζουμε τους πόλους (σύμφωνα με την εκφώνηση) ως εξής:

p5 = [(0.4 + 0.7i) (0.4 - 0.7i) (0.4 + 0.7i) (0.4 - 0.7i)]';

Με χρήση της συνάρτησης **zplane()** παίρνουμε το ακόλουθο διάγραμμα:



Για την εύρεση των διανυσμάτων συντελεστών a,b χρησιμοποιούμε πάλι τη συνάρτηση **zp2tf()** και εν συνεχεία με τη συνάρτηση **freqz()** παίρνουμε την απόκριση φάσης και πλάτους:



Τα διανύσματα a, b που υπολογίστηκαν ήταν τα εξής:

a5 = [1 -1.6 1.94 -1.04 0.4225]

b5 = [1 -1.6 1.28 -1.024 0.4096]

Από την απόκριση συχνότητας, παρατηρούμε πως μετά τη μέγιστη τιμή της, αρκετές συχνότητες παιρνάνε από το φίλτρο, άρα το φίλτρο έχει μεγάλο εύρος ζώνης.

Επίσης, όσο οι πόλοι πλησιάζουν στο 0, αυτό το φαινόμενο γίνεται όλο και πιο έντονο και συνεπώς, το εύρος ζώνης διέλευσης του φίλτρου γίνεται όλο και μεγαλύτερο.

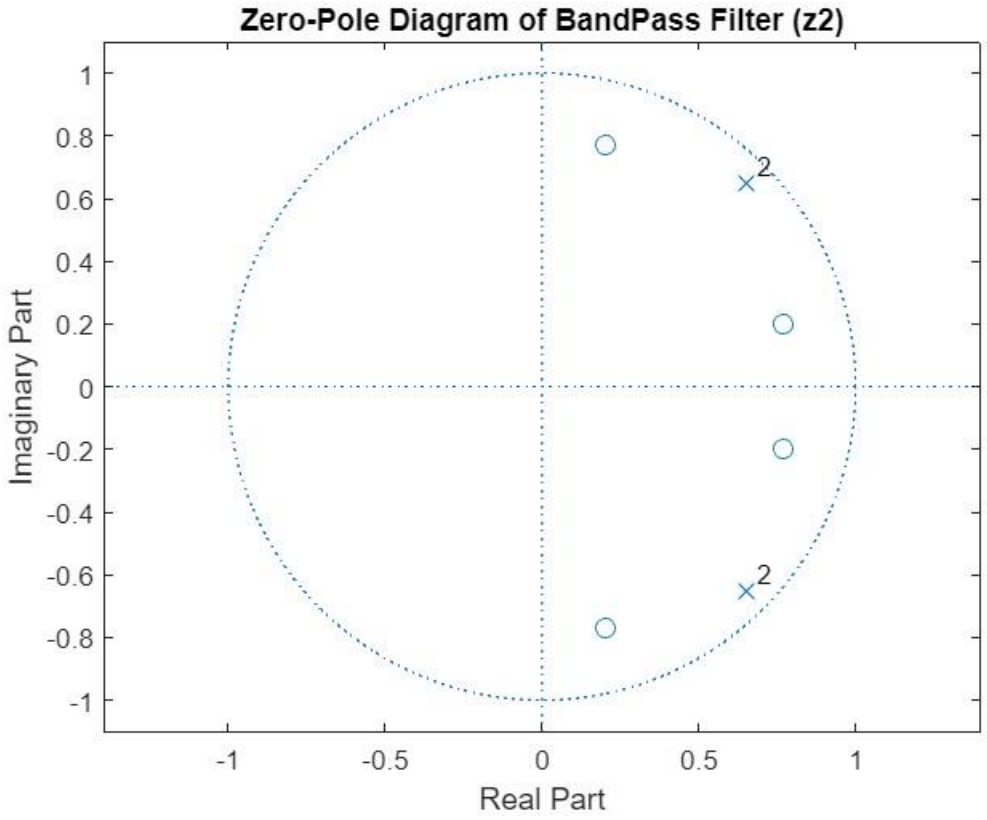
Η ασυμμετρία του σχήματος οφείλεται στην «ασύμμετρη» απόσταση των πόλων από τα ζεύγη των μηδενικών.

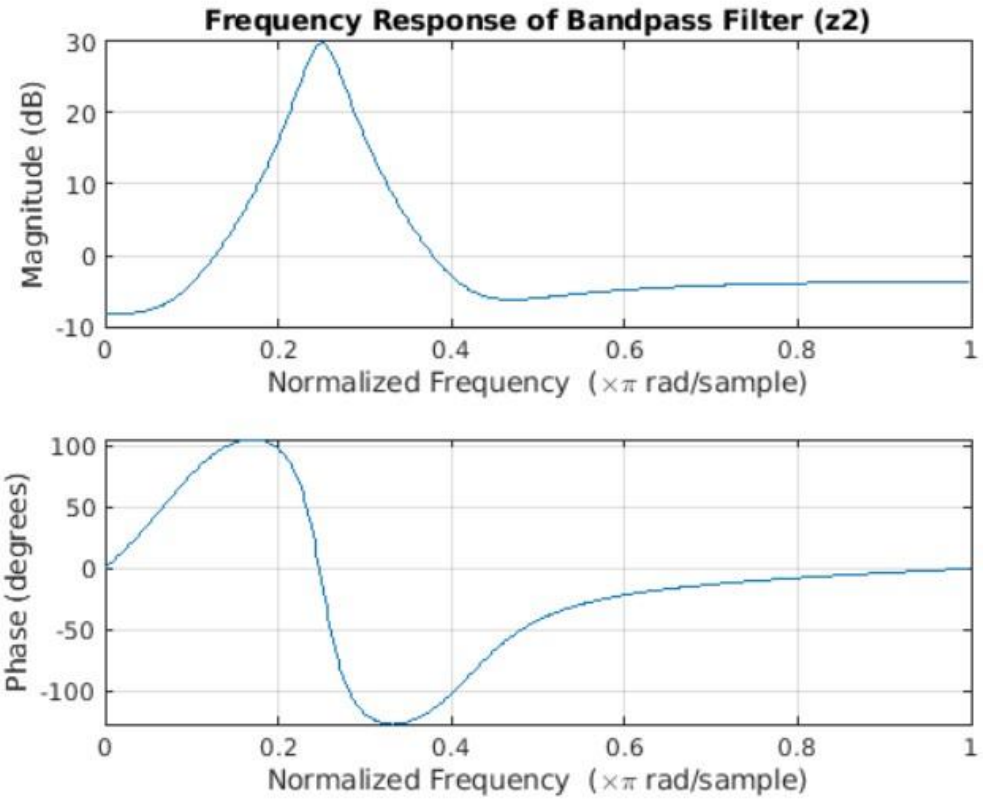
στ) Ακολουθώντας την ίδια διαδικασία με τα προηγούμενα ερωτήματα και χρησιμοποιώντας τους πόλους του α) ερωτήματος και τα νέα μηδενικά:

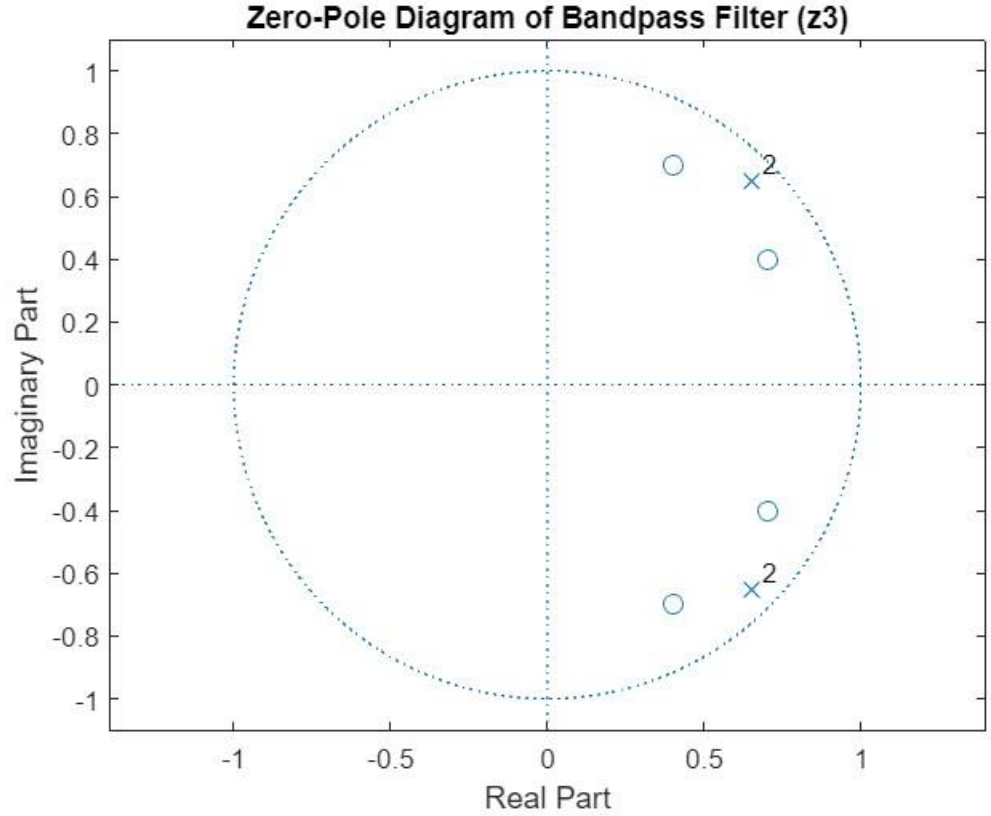
z2 = [(0.77 + 0.2i) (0.77 - 0.2i) (0.2 + 0.77i) (0.2 - 0.77i)]';

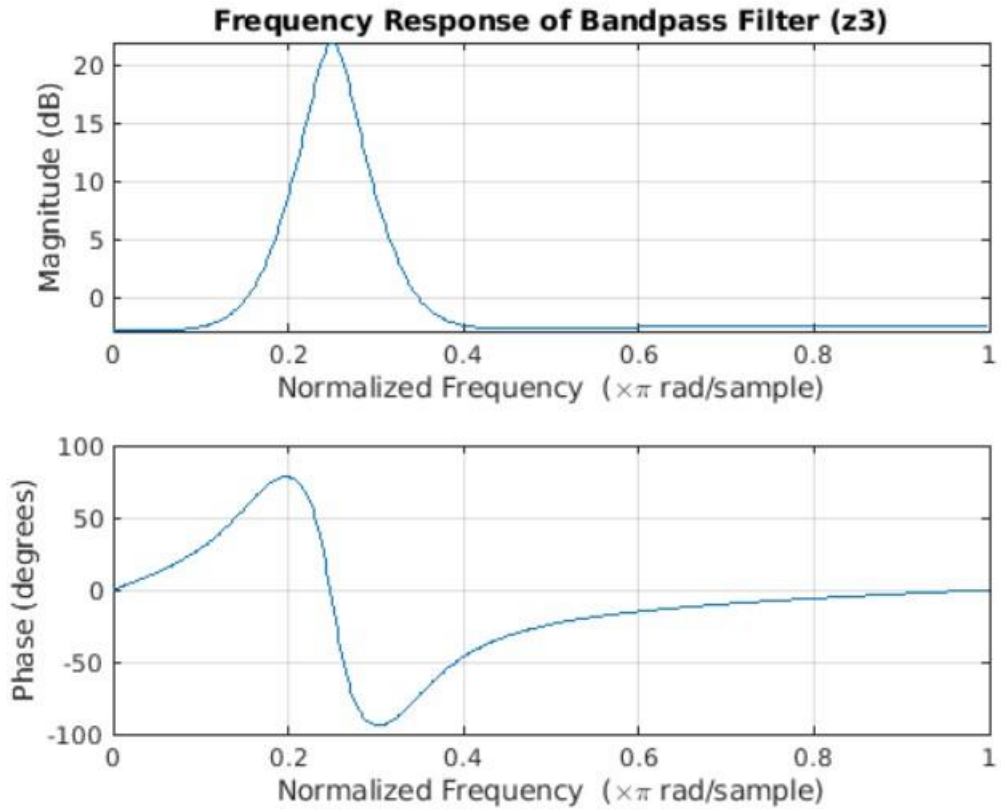
z3 = [(0.4 + 0.7i) (0.4 - 0.7i) (0.7 + 0.4i) (0.7 - 0.4i)]';

παίρνουμε τα παρακάτω διαγράμματα:




Frequency Response of Bandpass Filter (z2)
Magnitude (dB)
30
20
10
0
-10
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)
Phase (degrees)
100
50
0
-50
-100
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)


Zero-Pole Diagram of Bandpass Filter (z3)
Imaginary Part
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
2
2
-1
-0.5
0
0.5
1
Real Part

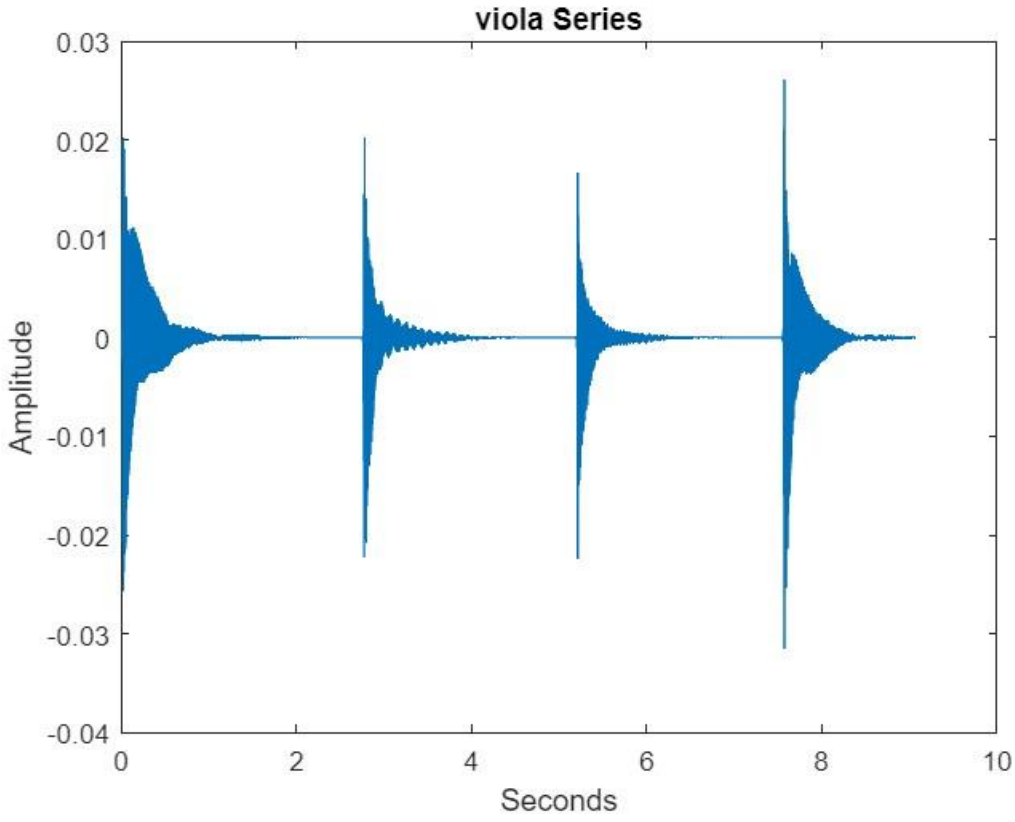


Από τα διαγράμματα παραπάνω, βλέπουμε ότι όσο τα μηδενικά πλησιάζουν τους πόλους , τόσο μικρότερο γίνεται το εύρος ζώνης διέλευσης του φίλτρου, δηλαδή το φίλτρο γίνεται καλύτερο, ενώ το κέρδος μειώνεται (μικρότερο πλάτος).

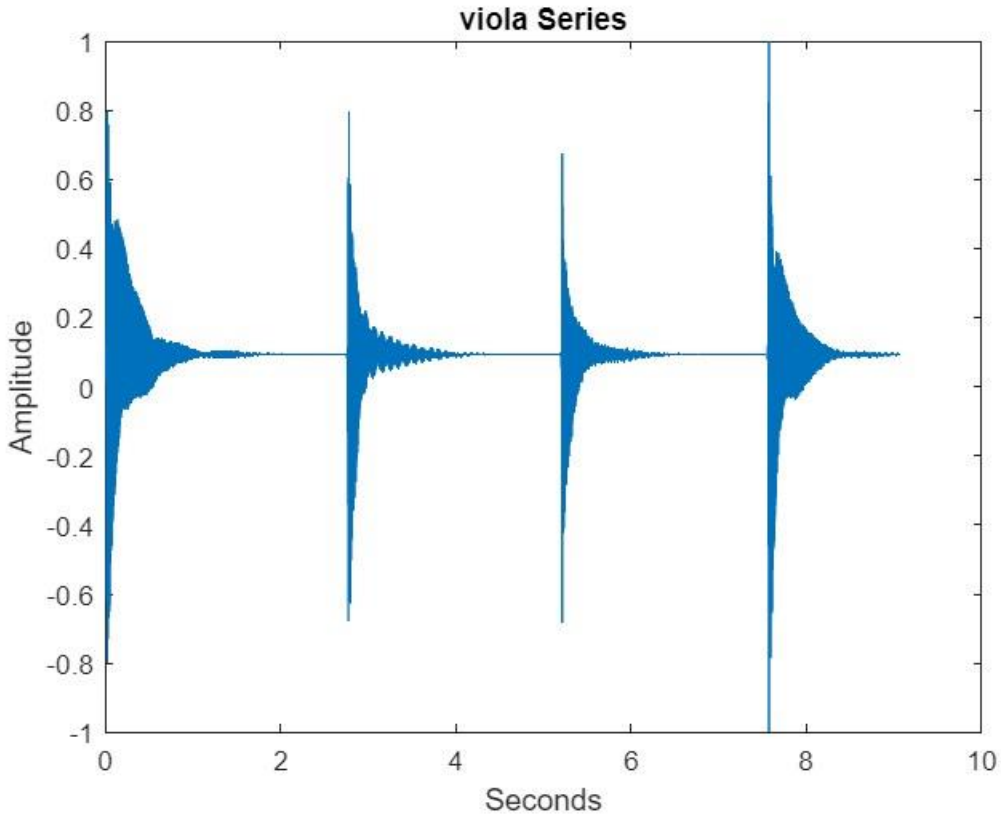
Άσκηση 2 – Ανάλυση Μουσικών Σημάτων και Εφαρμογή Φίλτρων:

2.1 Ανάλυση Μουσικών Σημάτων:

α) Με την εντολή **audioread()** φορτώνουμε το αρχείο το αρχείο “viola_series.wav” και κατόπιν, με τις εντολές **sound()** & **plot()** ακούμε και σχεδιάζουμε το σήμα:



β) Πρώτα, με χρήση της εντολής **normalize()** κανονικοποιούμε το σήμα στο διάστημα [-1, 1]:



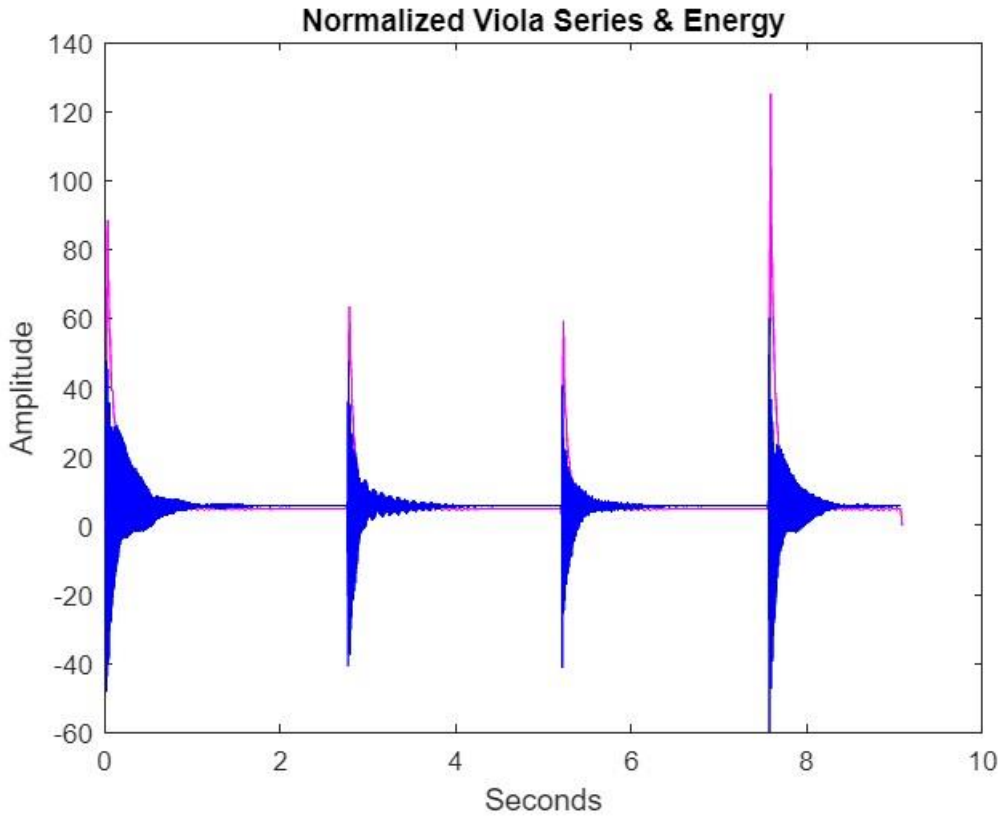
Για τον υπολογισμό της ενέργειας, γνωρίζουμε ότι:

- $E[n] = \sum_{m=0}^{M} x^2[m]w[n-m]$, όπου w[n] το παράθυρο Hamming το οποίο δίνεται από την εξίσωση: $w[n] = 0.54 + 0.46\cos(\frac{2\pi n}{N})$, το οποίο ισοδυναμεί με συνέλιξη των δύο σημάτων, δηλαδή:

$$E[n] = x^2[n] * w[n]$$

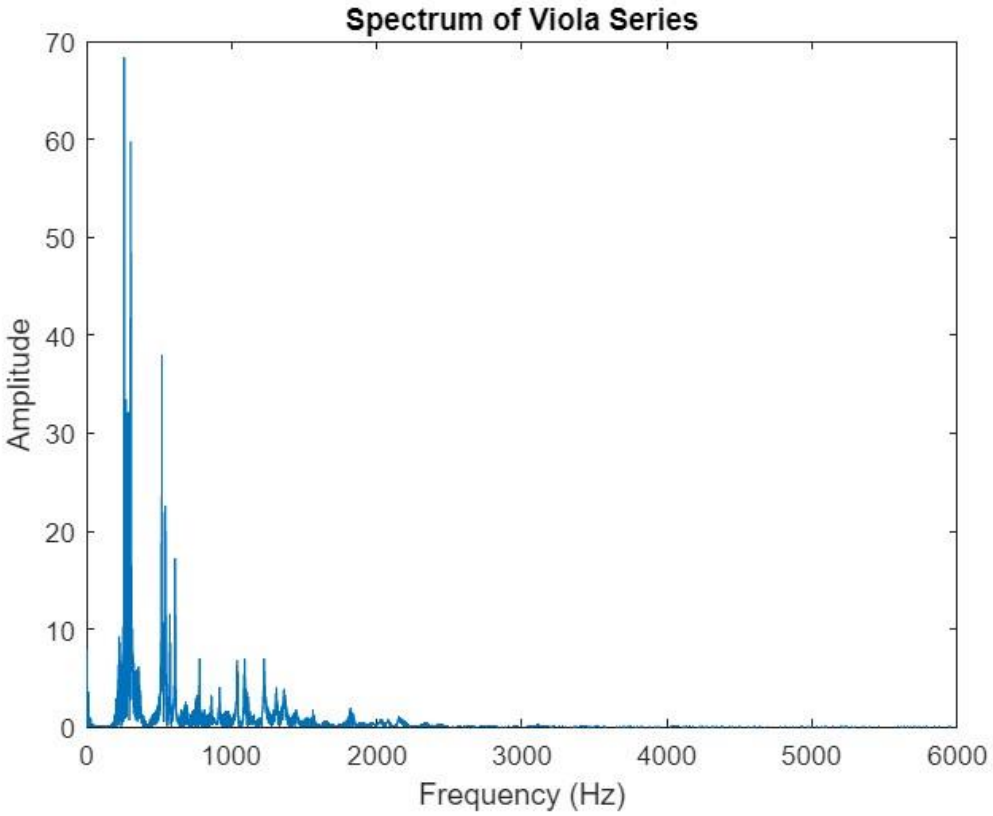
Για τον ορισμό του w[n] χρησιμοποιούμε την εντολή **hamming()** με παράμετρο

N = 1000 δείγματα, όπως μας λέει η εκφώνηση, ενώ για τον υπολογισμό της ενέργειας χρησιμοποιούμε την εντολή **conv()** για συνέλιξη. Μετά, κανονικοποιούμε το σήμα και με χρήση της **normalize()** στο διάστημα [-60, 60] και σχεδιάζουμε την ενέργεια του σήματος (magenta) μαζί με το σήμα (blue) στο ίδιο διάγραμμα:

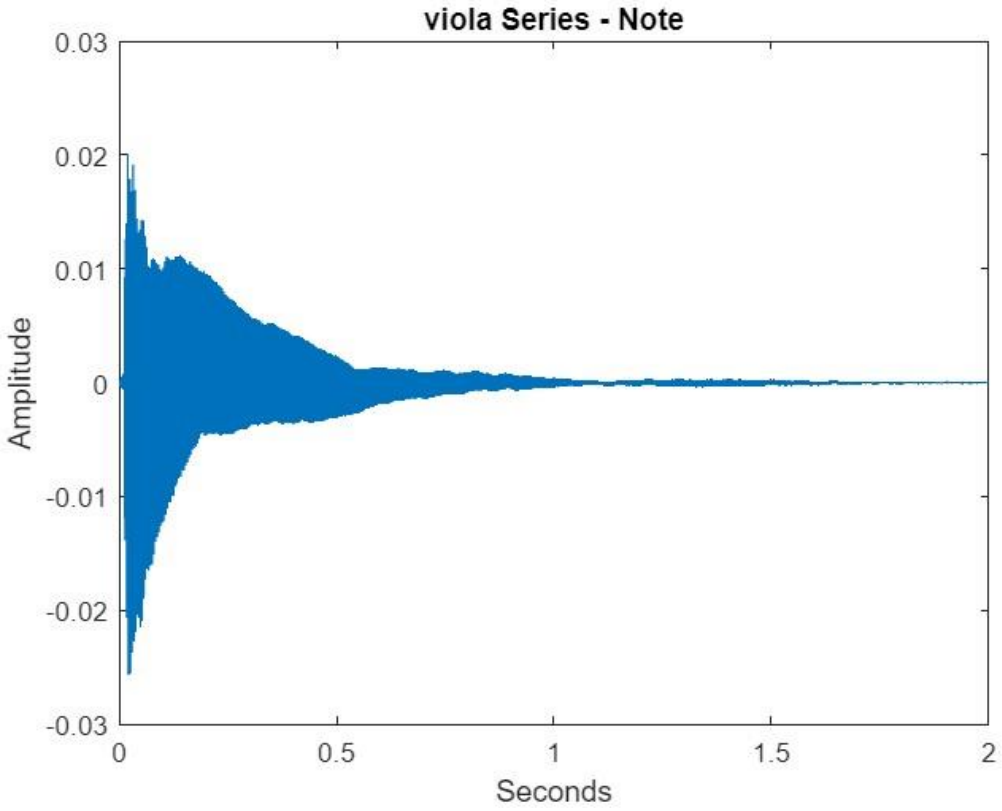


Από το διάγραμμα, παρατηρούμε πως η ενέργεια είναι συμφασική με το σήμα και έχει διπλάσιο πλάτος.

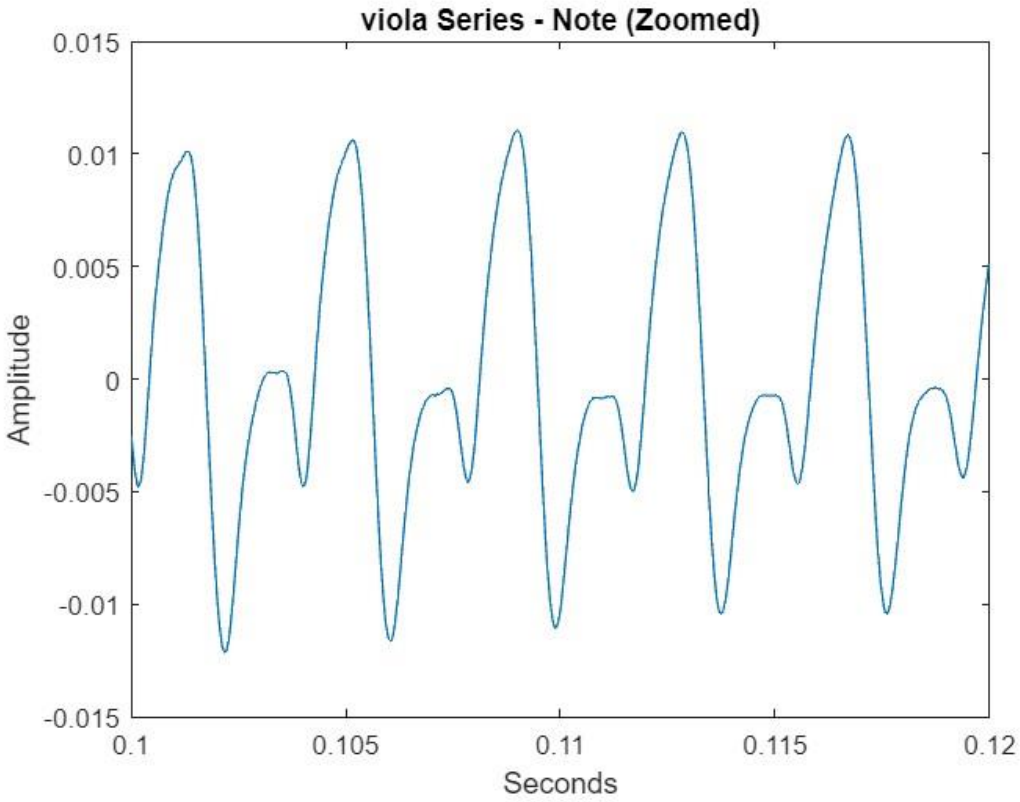
γ) Για τον υπολογισμό του Διακριτού Μετασχηματισμού Fourier (φάσμα) χρησιμοποιούμε τη συνάρτηση **fft()** και σχεδιάζουμε το μέτρο του φάσματος (abs(Y_viola)):



δ) Απομονώνουμε την πρώτη νότα από το σήμα με χρήση των **plot() & xlim():**



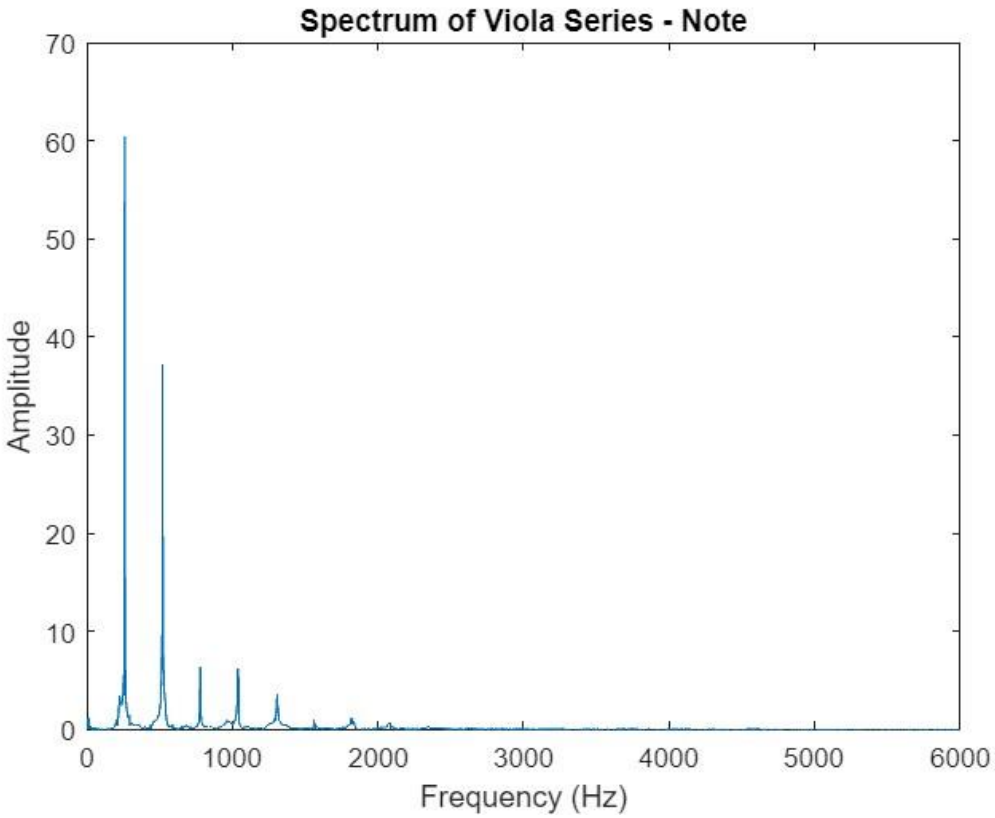
Ή σε μεγενθυμένη μορφή:



Από το διάγραμμα συμπεραίνουμε ότι η νότα είναι περιδοική με περίοδο T = 3.88 ms. Για τον υπολογισμό των δειγμάτων, κατά τα οποία είναι περιοδικό το σήμα, ισχύει:

$$T = N\frac{1}{Fs_{viola}} \Rightarrow N = T \cdot Fs_{viola} = 171.1 \Rightarrow N \cong 170\ δείγματα.$$

ε) Το μέτρο του φάσματος που προκύπτει μέσω της εντολής **fft()** είναι το ακόλουθο:



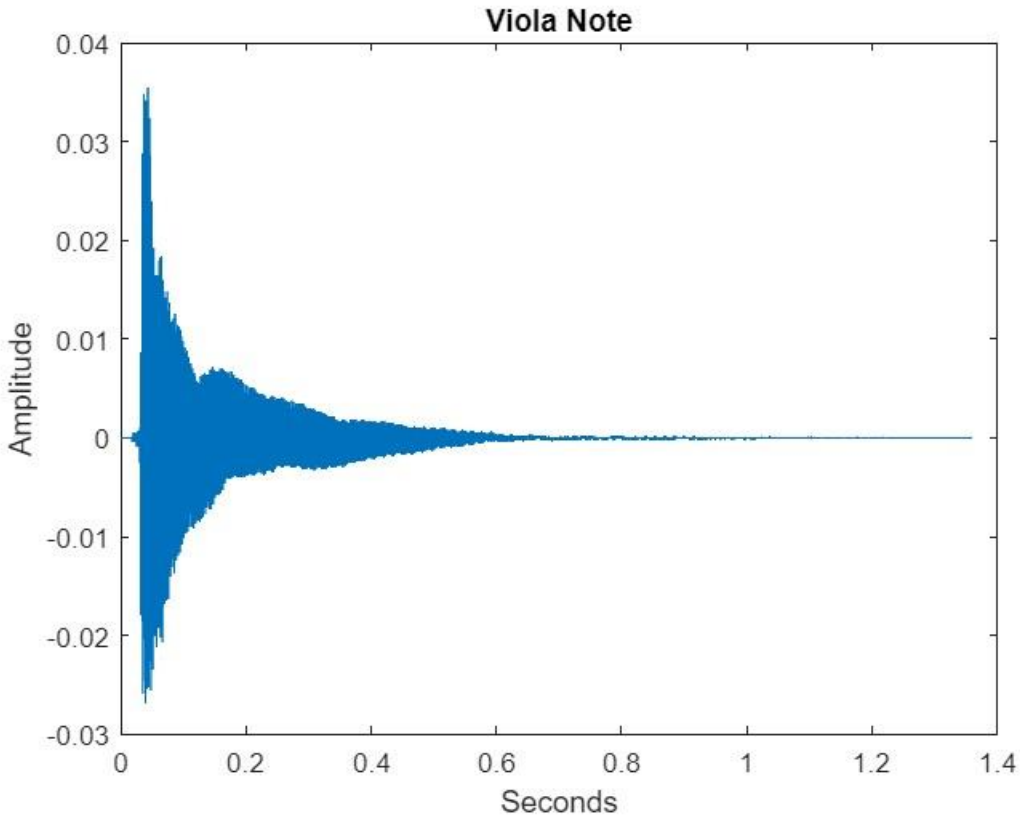
Η μέγιστη τιμή του φάσματος εντοπίζεται στα 259 Hz, δηλαδή Fo = 259 Hz.

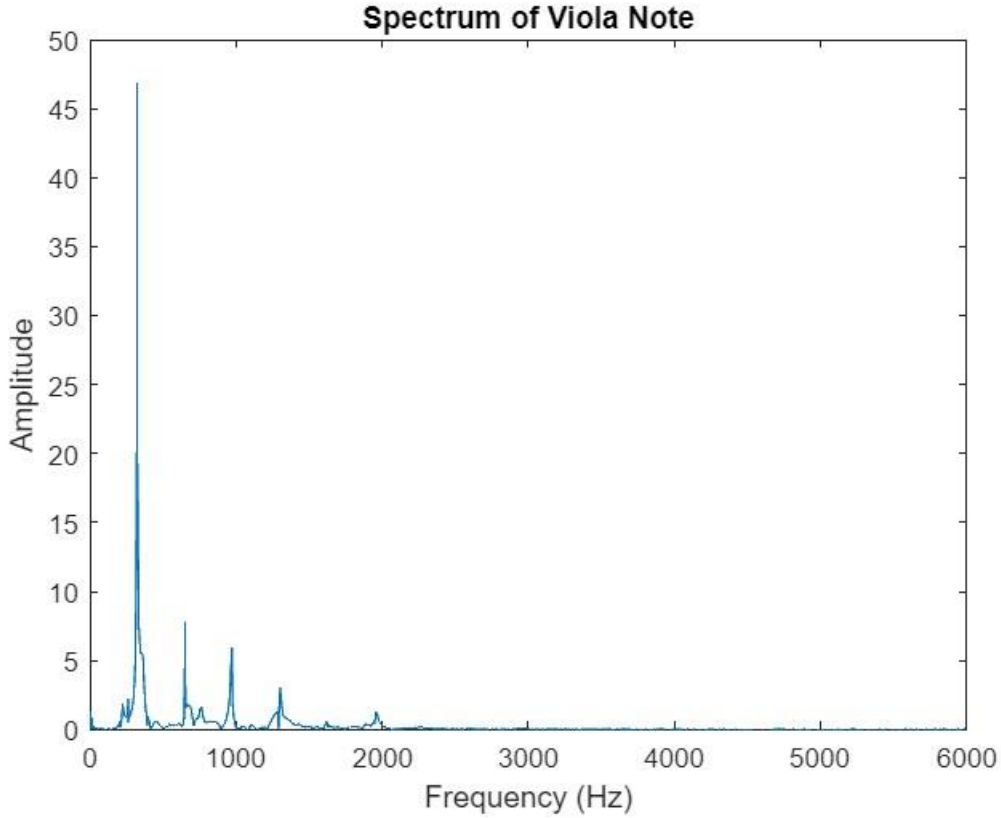
Άρα $To = \frac{1}{Fo} = \frac{1}{259} = 3.86\ ms\ (\cong 3.88\ ms)$, δηλαδή πολύ κοντά στην αρχική τιμή που βρήκαμε νωρίτερα.

Συγκρίνοντας το διάγραμμα με αυτό του ερωτήματος γ), βλέπουμε πως τώρα, εφόσον έχει απομονωθεί μία νότα (η πρώτη), απουσιάζουν οι αρμονικές των υπόλοιπων νοτών και εμφανίζονται μόνο οι αρμονικές της νότας που απομονώσαμε.

Όσον, αφορά την εμφάνιση αρμονικών υψηλότερης τάξης, βλέπουμε πως το πλάτος τους μειώνεται εκθετικά. Μάλιστα, από ένα σημείο και μετά, το πλάτος τείνει στο 0.

στ) Όπως και στο ερώτημα α), έτσι και εδώ , φορτώνουμε το αρχείο “viola_note.wav” με χρήση της εντολής **audioread()** και σχεδιάζουμε τη νότα καθώς και το φάσμα της:


Viola Note
Amplitude
Seconds
0.04
0.03
0.02
0.01
0
-0.01
-0.02
-0.03
0
0.2
0.4
0.6
0.8
1
1.2
1.4


Spectrum of Viola Note
Amplitude
Frequency (Hz)
50
45
40
35
30
25
20
15
10
5
0
0
1000
2000
3000
4000
5000
6000

Από το φάσμα παραπάνω, παρατηρούμε πως:

- ✓ Η 1η αρμονική εντοπίζεται στα 321.93 Hz
- ✓ Η 2η αρμονική εντοπίζεται στα 648.27 Hz
- ✓ Η 4η αρμονική εντοπίζεται στα 1300.21 Hz

Άρα, αν F_o = 321.93 Hz, τότε η 2η και η 4η αρμονική εντοπίζονται κοντά στις συχνότητες $2 \cdot Fo$ και $4 \cdot Fo$ αντίστοιχα.

Συνεπώς, για την υλοποίηση του φίλτρου, προσπαθούμε να βρούμε μηδενικά και πόλους, ώστε να πετύχουμε κεντρική συχνότητα στα $2 \cdot Fo \cong 650\ Hz$ για τη 2η αρμονική και $4 \cdot Fo \cong 1300\ Hz$ για την 4η.

Ύστερα, ακολουθούμε την παρακάτω διαδικασία, δηλαδή:

- ❖ με χρήση της **zp2tf()** βρίσκουμε τα διανύσματα a και b
- ❖ με την **freqz()** σχεδιάζουμε την απόκριση συχνότητας
- ❖ με την **impz()** υπολογίζουμε την κρουστική απόκριση
- ❖ με χρήση της **conv()** κάνουμε συνέλιξη της νότας (y_viola_note) με την κρουστική απόκριση του φίλτρου (h_viola \ h_viola_2) , ώστε να υπολογίσουμε την έξοδο (y_filter_1 \ y_filter_2).
- ❖ με χρήση της **plot()** σχεδιάζουμε την έξοδο
- ❖ για το πέρασμα στο πεδίο της συχνότητας χρησιμοποιούμε την συνάρτηση **fft()** και
- ❖ με τη χρήση της **plot()** σχεδιάζουμε το φάσμα της εξόδου.

Στο μέρος 1 είδαμε τα εξής:

1. Όσο οι πόλοι πλησιάζουν τον μοναδιαίο κύκλο, τόσο μεγαλύτερο γίνεται το πλάτος της απόκρισης συχνότητας και τόσο πιο απότομη γίνεται η απόκριση φάσης.
2. Όταν οι πόλοι βγουν έξω από τον μοναδιαίο κύκλο, τότε το σύστημα γίνεται ασταθές και αυτό δεν το θέλουμε.
3. Όσο οι πόλοι πλησιάζουν στο 0, τόσο μεγαλύτερο γίνεται το εύρος ζώνης διέλευσης του φίλτρου, δηλαδή παιρνάνε όλο και περισσότερες συχνότητες, το οποίο δε θέλουμε.
4. Όσο τα μηδενικά πλησιάζουν τους πόλους, τόσο μικρότερο γίνεται το εύρος ζώνης διέλευσης του φίλτρου.

Όμως, για την εύρεση των πόλων και μηδενικών, αξιοποιούμε τα συμπεράσματα από το μέρος 1, που αναφέρθηκαν παραπάνω. Συγκεκριμένα, θέλουμε:

- ➢ Οι πόλοι να βρίσκονται πολύ κοντά στον μοναδιαίο κύκλο, αλλά εντός αυτού, ώστε το σύστημα να είναι ευσταθές και
- ➢ Η απόσταση πόλων και μηδενικών να είναι πολλή μικρή.

Σύμφωνα με όλα τα παραπάνω, και κατόπιν δοκιμών, βρέθηκαν οι εξής τιμές για τους πόλους και τα μηδενικά:

Για τη 2η αρμονική:

p_viola = [(0.993 + 0.09i) (0.993 - 0.09i) (0.993 + 0.09i) (0.993 - 0.09i)]';

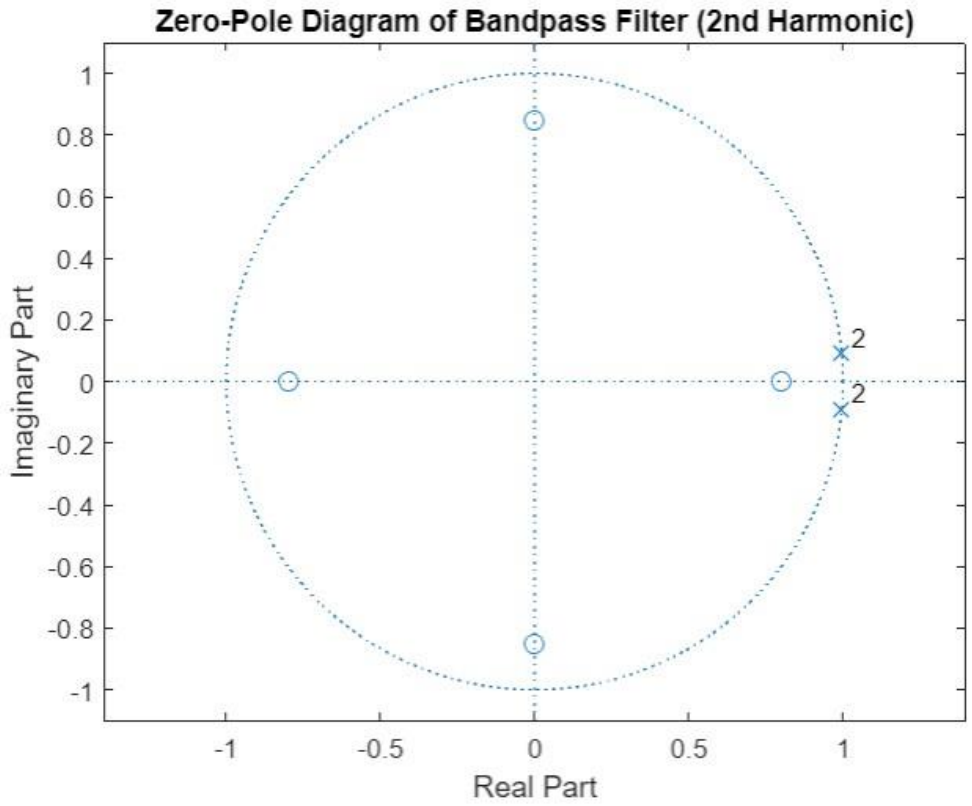
z_viola = [0.8 -0.8 0.85i -0.85i]';

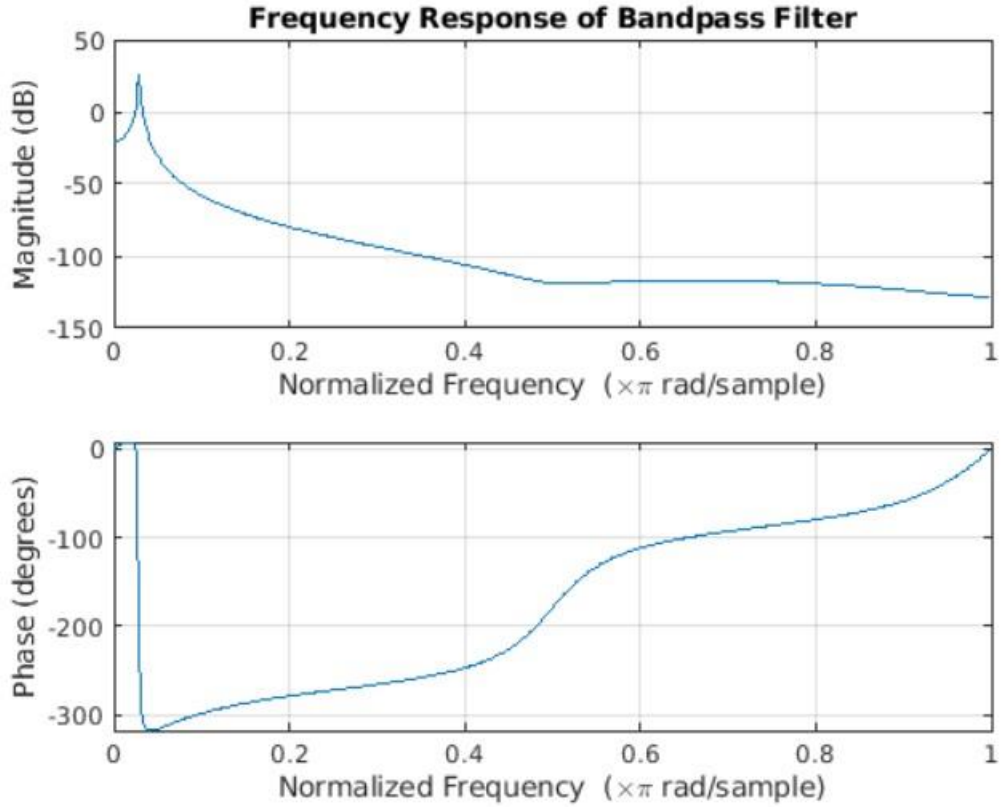
Για τη 4η αρμονική:

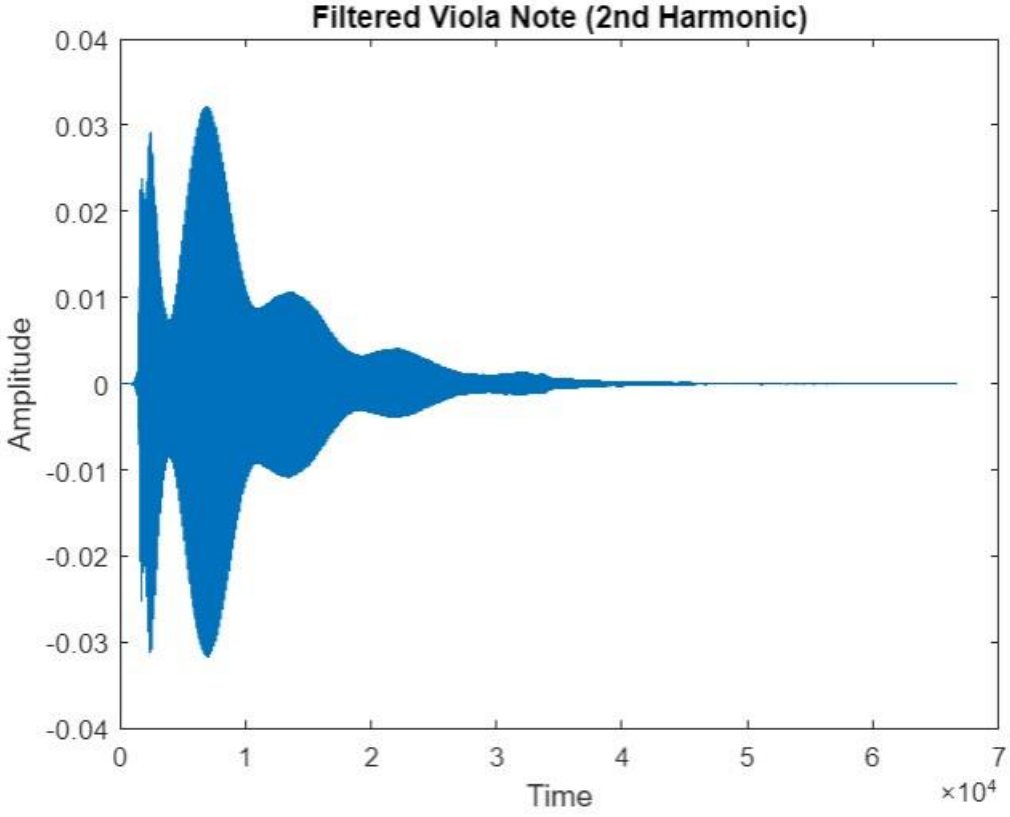
p_viola_2 = [(0.98 + 0.185i) (0.98 – 0.185i) (0.98 + 0.185i) (0.98 – 0.185i)]';

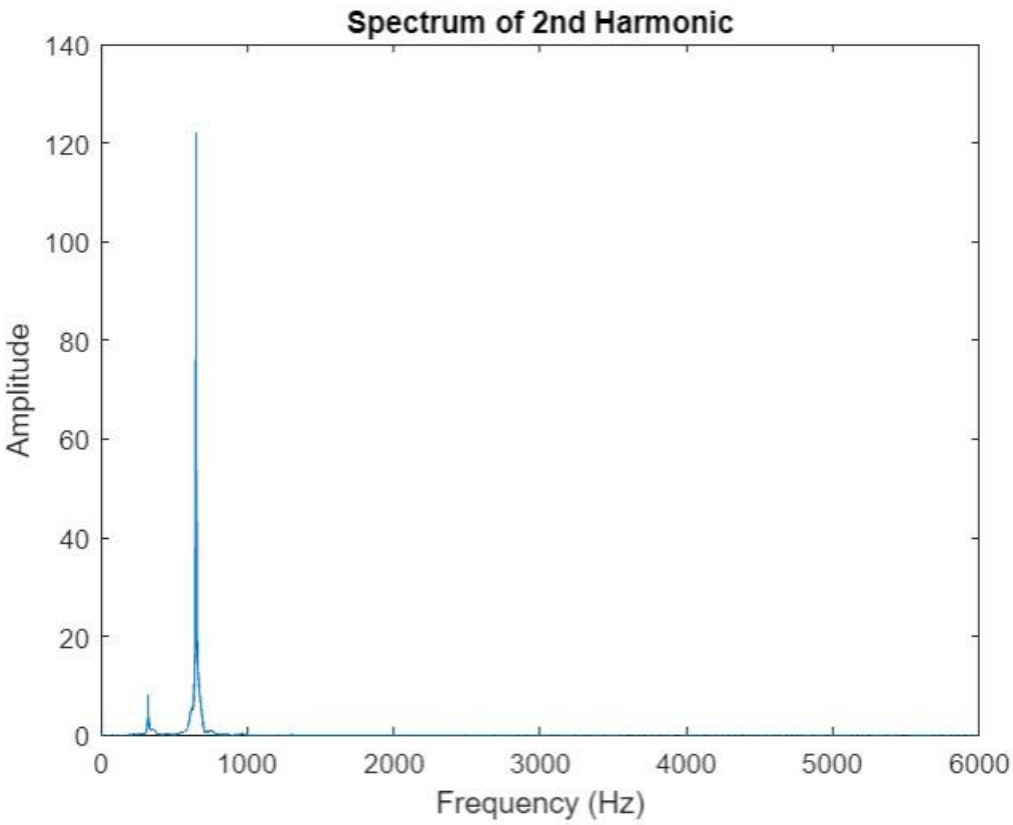
Z_viola_2 = [0.8 -0.8 0.85i -0.85i]';

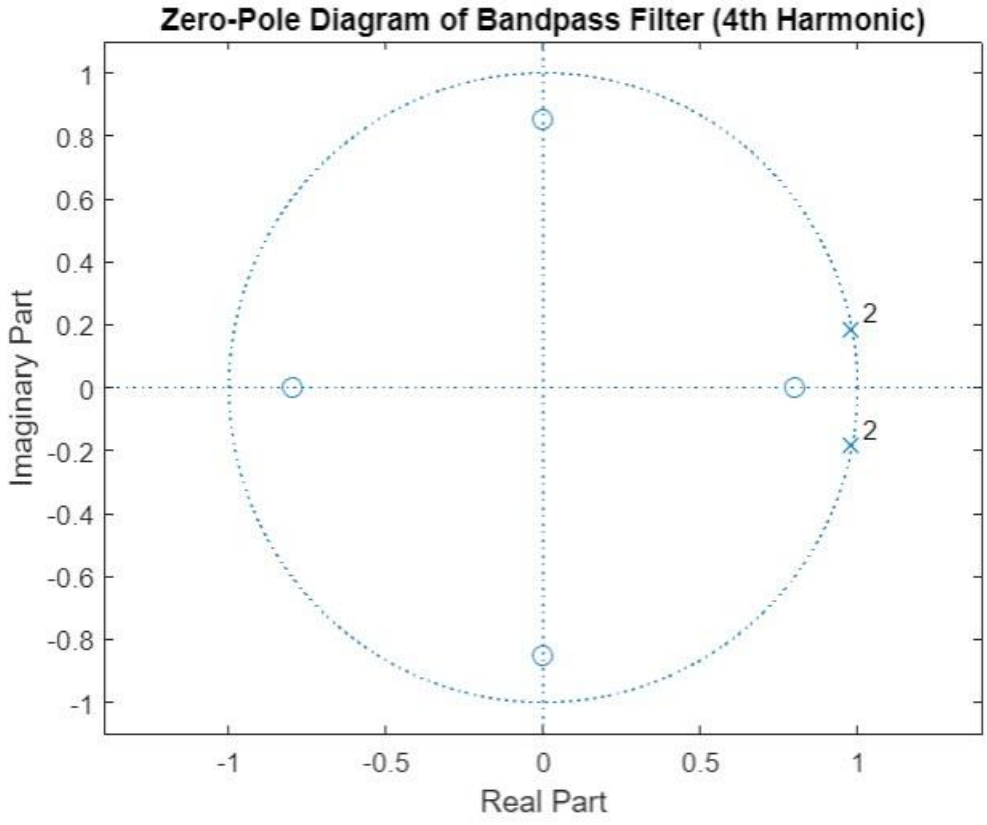
Τα διαγράμματα που προέκυψαν παρατίθενται παρακάτω:

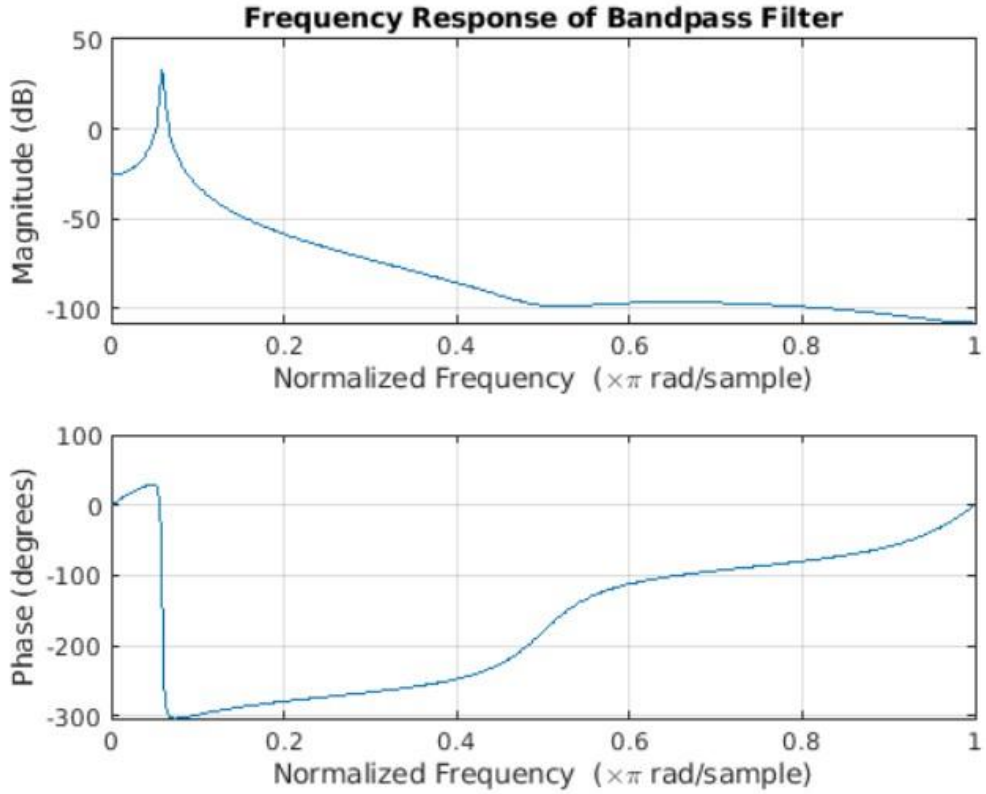


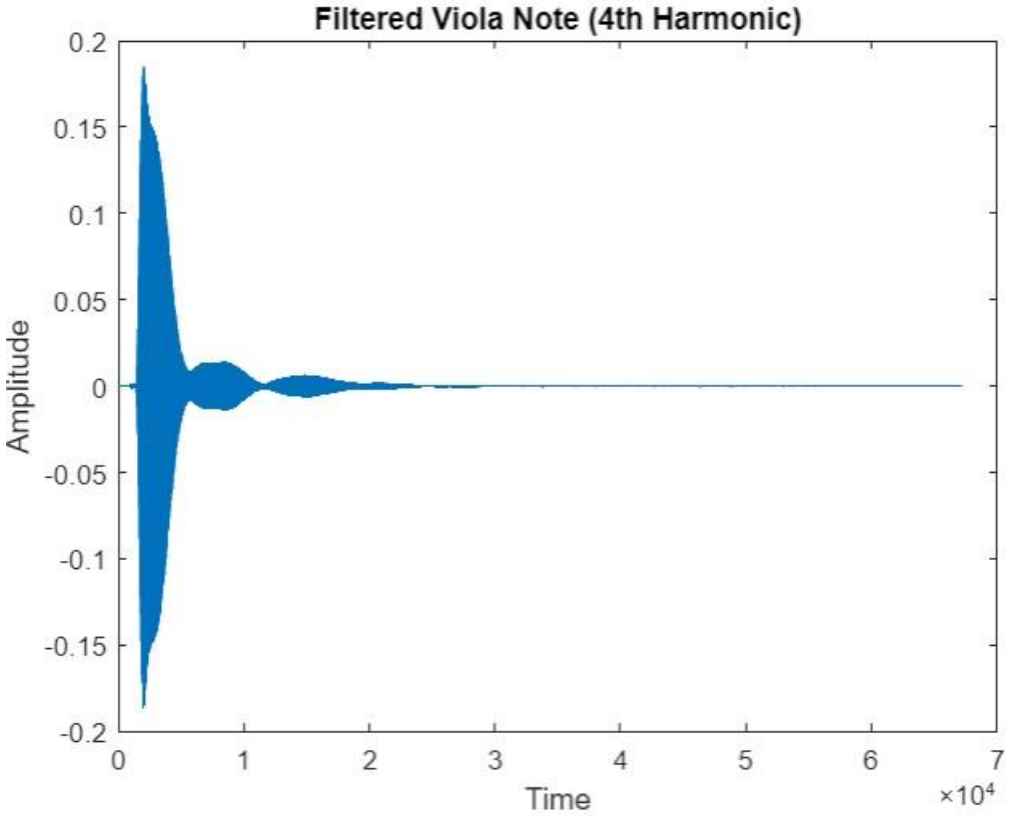

Frequency Response of Bandpass Filter
Magnitude (dB)
50
0
-50
-100
-150
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)
Phase (degrees)
0
-100
-200
-300
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)

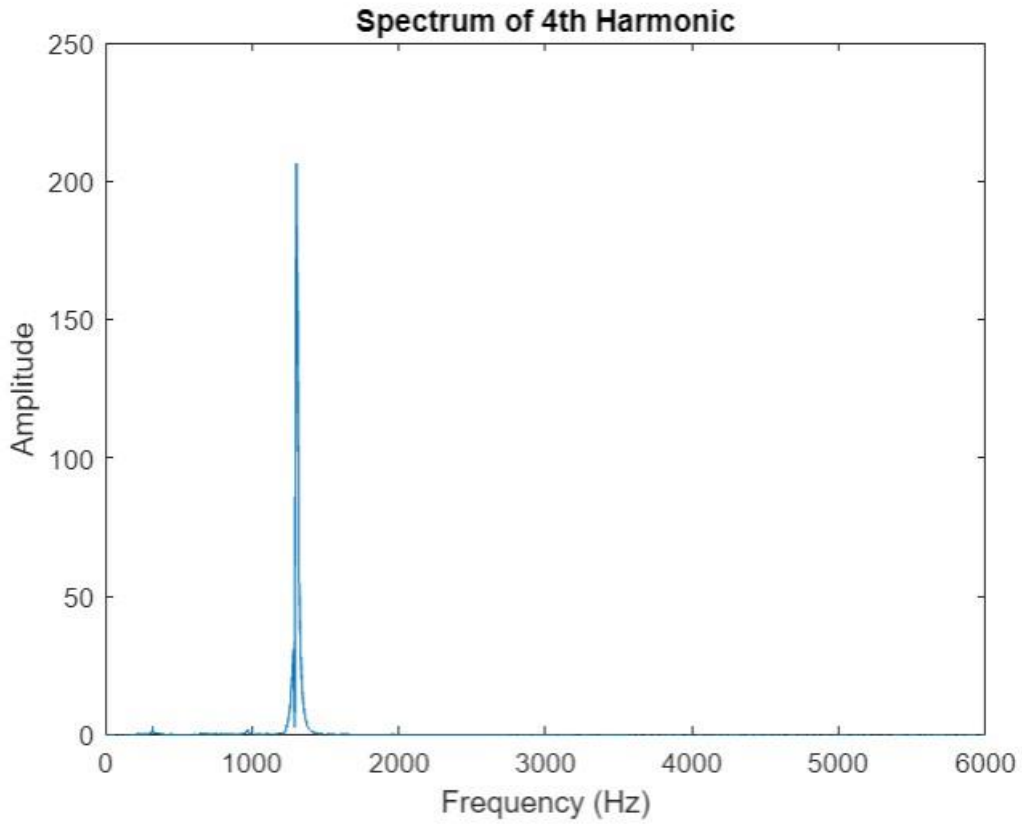

Filtered Viola Note (2nd Harmonic)
Amplitude
0.04
0.03
0.02
0.01
0
-0.01
-0.02
-0.03
-0.04
0
1
2
3
4
5
6
7
Time
×10^4


Spectrum of 2nd Harmonic
Amplitude
Frequency (Hz)
0
20
40
60
80
100
120
140
0
1000
2000
3000
4000
5000
6000


Zero-Pole Diagram of Bandpass Filter (4th Harmonic)
Imaginary Part
Real Part
2
2
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.5
0
0.5
1


Frequency Response of Bandpass Filter
Magnitude (dB)
50
0
-50
-100
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)
Phase (degrees)
100
0
-100
-200
-300
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)


Filtered Viola Note (4th Harmonic)
Amplitude
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
0
1
2
3
4
5
6
7
Time
×10^4

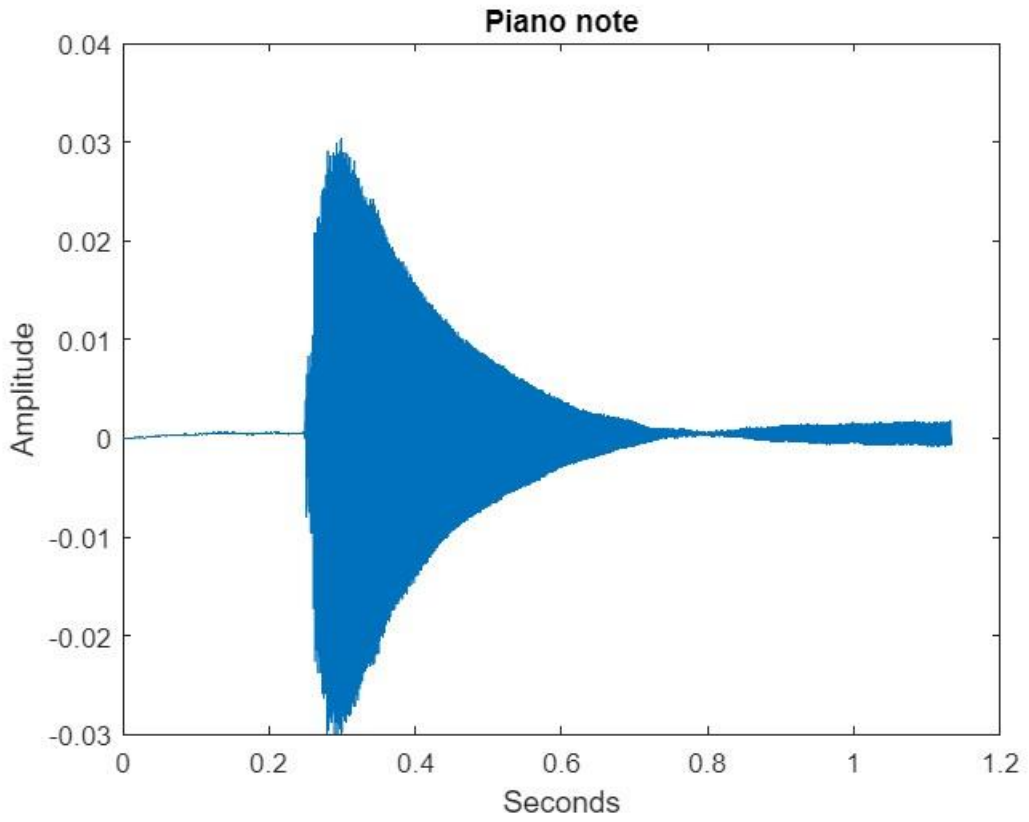


Όπως βλέπουμε, στο φίλτρο απομόνωσης της 2ης αρμονικής, πετύχαμε κεντρική συχνότητα 648.121 Hz, ενώ για την απομόνωση της 4ης αρμονικής, 1301.72 Hz, δηλαδή και στις 2 περιπτώσεις, το αποτέλεσμα είναι αρκετά ικανοποιητικό και κοντά στο στόχο μας. Επίσης, από την απόκριση συχνότητας των φίλτρων, φαίνεται ότι αυτά αντιστοιχούν σε ζωνοπερατά φίλτρα (κάτι προφανώς αναμενόμενο), όπως επίσης και ότι το εύρος της ζώνης διέλευσης του φίλτρου, είναι περιορισμένο.

Για τον περιορισμό τους πλάτους , στη συνάρτηση **zp2tf()** το κέρδος k ήταν αρκετά μικρό ($10^{-5} - 10^{-4}$).

2.2 Εφαρμογή Φίλτρων για τη Δημιουργία Ηχούς και Αντήχησης εφέ σε Μουσικά Σήματα:

α) Όμοια με την προηγούμενη άσκηση, χρησιμοποιούμε την εντολή **audioread()** για τη φόρτωση του αρχείου “piano_note.wav” και τις εντολές **plot()** και **sound()** για τη σχεδίαση και το άκουσμα της νότας:



β) Για την υλοποίηση του φίλτρου ηχούς θα χρησιμοποιήσουμε την εξίσωση της άσκησης 1.1 (η οποία χρησιμεύει στην προσομοίωση του echo_effect) με παράμετρο c = 0.6 , όπως μας λέει η εκφώνηση:

$$y[n] = 0.6x[n] + 0.4x[n - P]$$

Γνωρίζουμε ότι η συχνότητα δειγματοληψίας είναι: Fs = 44.1 kHz. Οπότε, για την εύρεση της παραμέτρου P (ώστε να έχουμε καθυστέρηση 0.15s) χρησιμοποιούμε την εξής σχέση:

$$\frac{1}{Fs} \cdot P = t_d \Rightarrow P = Fs \cdot t_d = 44.1k \cdot 0.15 \Rightarrow \boldsymbol{P = 6615}$$

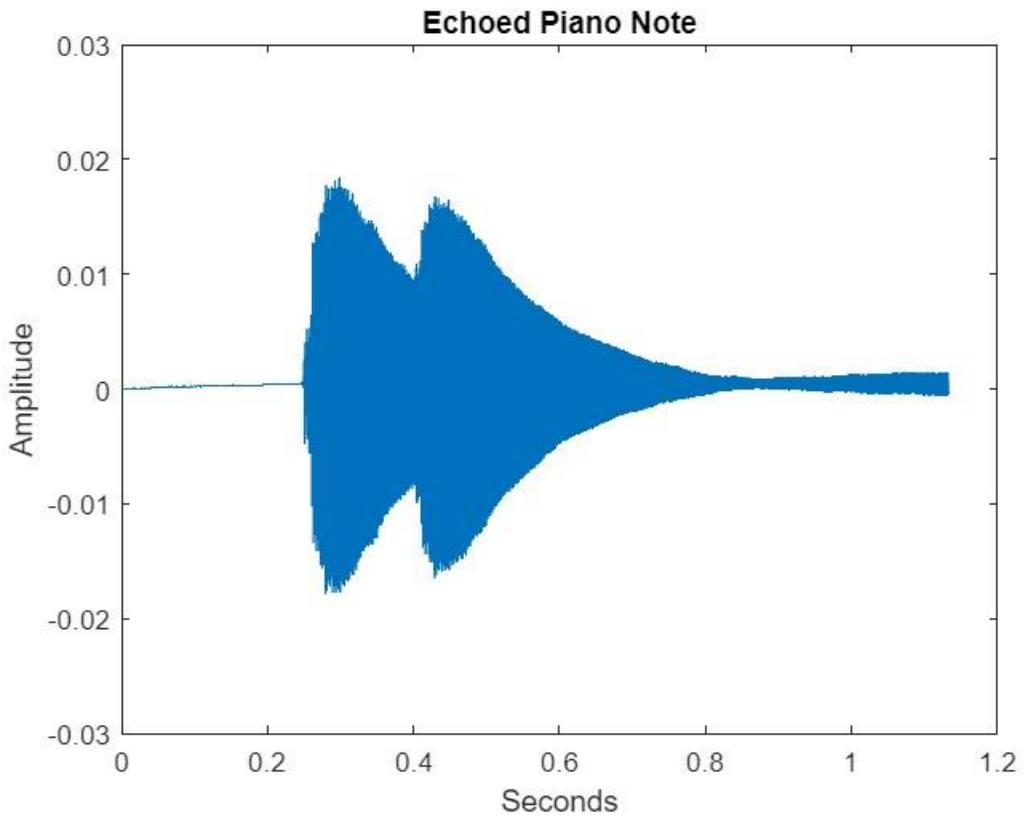
Όπως και στην άσκηση 1.1, έτσι και εδώ, με απλή παρατήρηση βρίσκουμε τα διανύσματα a_echo & b_echo:

a_echo = [1 0 0 … 0] (μέγεθος 1x6615)

b_echo = [0.6 0 0 … 0 0.4] (μέγεθος 1x6615)

Έχοντας υπολογίσει τα διανύσματα a_echo & b_echo, βρίσκουμε τώρα την κρουστική αποκριση h_echo του φίλτρου ηχούς, με τη βοήθεια της εντολής **impz()** .

Εν συνεχεία, φιλτράρουμε τη νότα μέσω της εντολής **filter()** (η οποία δέχεται ως παραμέτρους τα a_echo, b_echo και τη νότα y_piano) και με τις εντολές plot() και sound() σχεδιάζουμε και ακούμε τη νότα:



Για την κατασκευή του φίλτρου αντήχησης , χρησιμοποιούμε 3 διαδοχικά (σε σειρά) φίλτρα δημιουργίας ηχούς με τη βοήθεια της **conv()**, δηλαδή:

$$reverb_h = h_{echo} * h_{echo} * h_{echo}$$

Για τον υπολογισμό των διανυσμάτων ai_piano_reverb * bi_piano_reverb, χρησιμοποιούμε τους τύπους:

$$ai'_{piano_{reverb}} = T^{-1}_{reverb_h} \cdot reverb'_h$$

$$bi'_{piano_{reverb}} = reverb'_h + T_{ho} \cdot ai'_{piano_{reverb}}$$

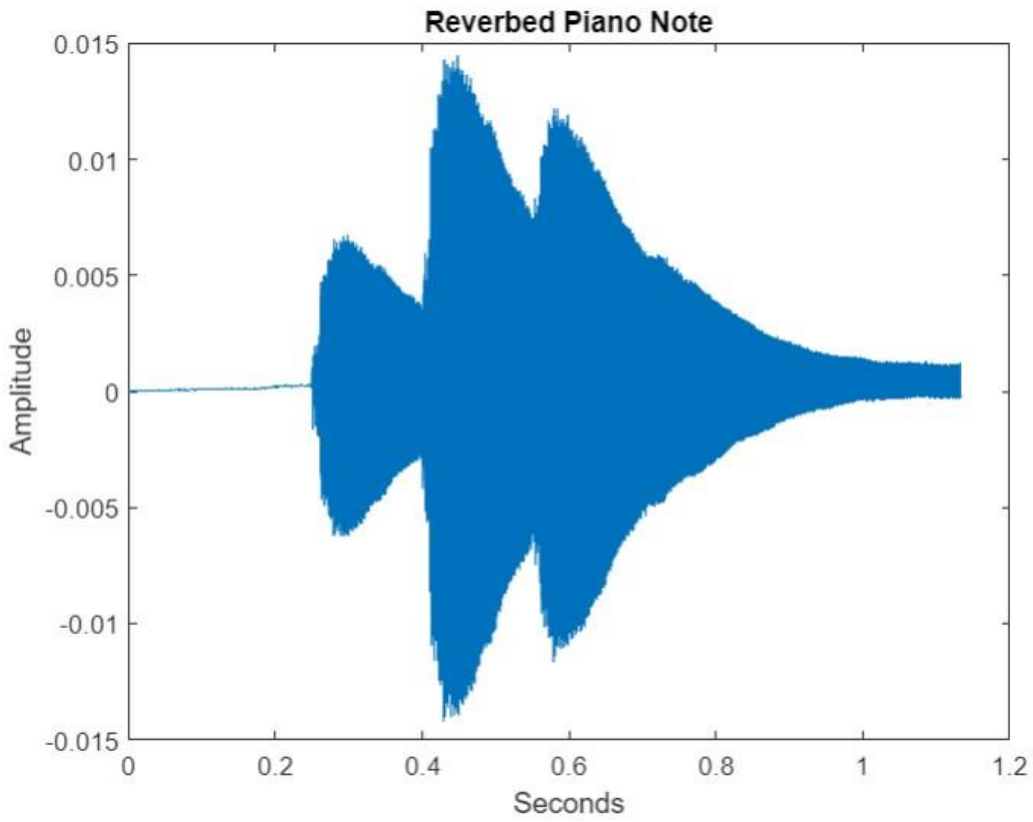
Έτσι, βρίσκουμε:

$$ai_{piano_{reverb}} = [1 \;\; 0 \;\; \dots \;\; 0]' \Rightarrow ai_{piano_{reverb}} = 1 \; (\mu\acute{\varepsilon}\gamma\varepsilon\theta o\varsigma \; 1x1)$$

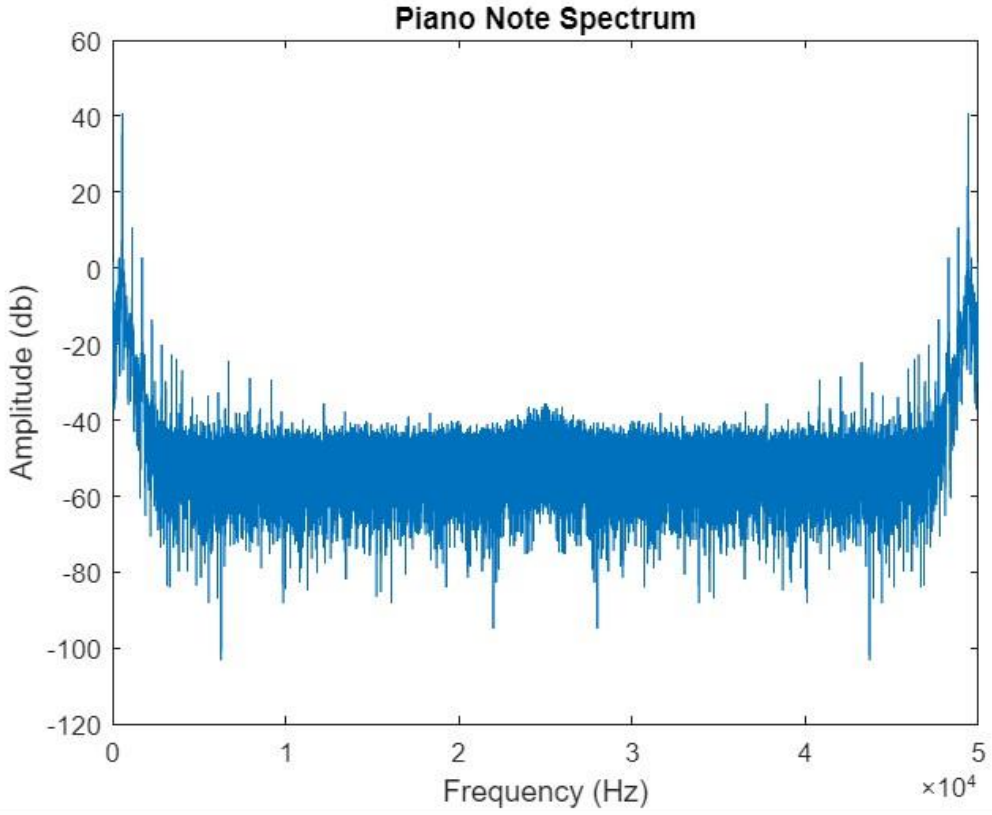
Αφού ai_piano_reverb = 1 ⇒ bi_piano_reverb = reverb_h

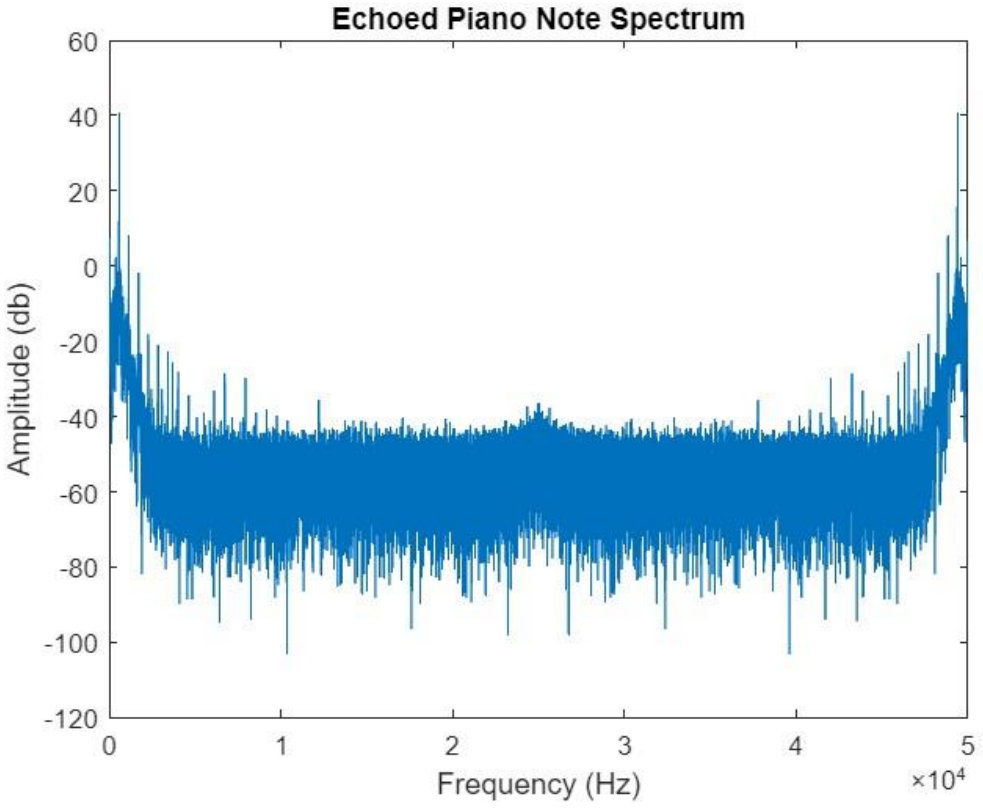
<u>Σημείωση</u>: (το διάνυσμα ai_piano_reverb υπολογίστικε αρχικά μέσω της εντολής **toeplitz()** και βρέθηκε ίσο με [1 0 … 0]' . Ωστόσο, επειδή ο υπολογισμός του καθυστερούσε το πρόγραμμα, πήρα απευθείας το αποτέλεσμα και όρισα ai_piano_reverb = 1.**)**

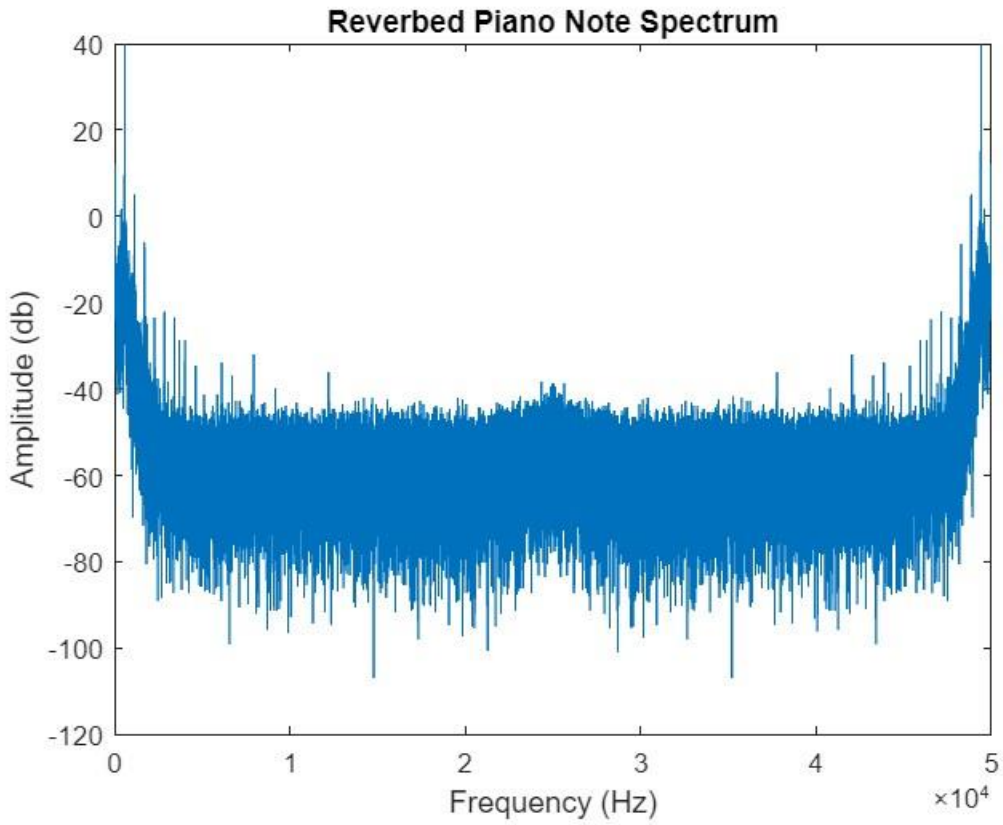
Αφού βρούμε και τα διανύσματα a και b, φιλτράρουμε το σήμα με την εντολή **filter()** και ακούμε και σχεδιάζουμε την έξοδο με τις εντολές **sound()** & **plot()** αντίστοιχα:



γ) Με χρήση των συναρτήσεων **fft()** και **plot()** σχεδιάζουμε το μέτρο του φάσματος (σε λογαριθμική κλίμακα) του αρχικού σήματος καθώς και των echoed και reverbed:







Συγκρίνοντας τα 3 διαγράμματα, παρατηρούμε πως το φάσμα του echoed σήματος είναι πιο «πυκνό» από το φάσμα του αρχικού και το φάσμα του reverbed σήματος είναι ακόμα πιο «πυκνό».

Για την ερμηνεία των παρατηρήσεων, χρειαζόμαστε το εξής:

$$y_{echoed}[n] = 0.6x[n] + 0.4x[n - 6615\,]\text{, } \textit{όπου x[n] το αρχικό σήμα}$$

Μετασχηματίζοντας την παραπάνω σχέση κατά Fourier:

$$Y_{echoed}(\omega) = \left(0.6 + 0.4 \cdot e^{-6615j\omega}\right)X(\omega)$$

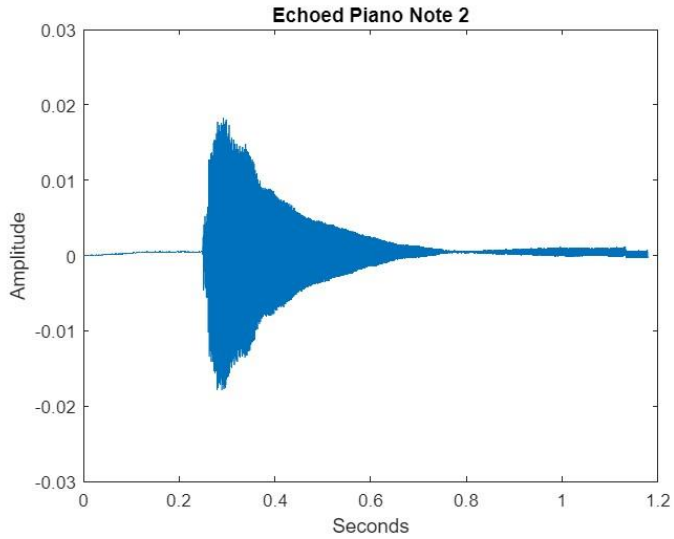
Από την τελευταία σχέση συμπεραίνουμε πως είναι λογικό , το φάσμα του echoed σήματος να είναι πιο πυκνό , αφού ουσιαστικά εμφανίζεται το σήμα 2 φορές με μερική επικάλυψη λόγω του γραμμικού συνδυασμού που προκύπτει από την εξίσωση.

Όσον αφορά το φάσμα του reverbed σήματος , αυτό προφανώς θα είναι ακόμη πιο πυκνό αφού προκύπτει από τριπλή συνέλιξη της h_echo.

δ) Για το ερώτημα αυτό, ορίζουμε τα διανύσματα a και b με παρόμοιο τρόπο και ουσιαστικά μεταβάλλουμε το μέγεθός τους (δηλαδή το P), μέχρι να πετύχουμε το επιθυμητό αποτέλεσμα.

Για την εύρεση του P, χρησιμοποιούμε τη συνάρτηση **sound()** για να ακούσουμε το νέο echoed σήμα και δοκιμάζουμε διάφορες τιμές, μέχρι να ακούσουμε έναν μόνο συνεχή ήχο, δηλαδή μέχρι να μη μπορούμε να αντιληφθούμε με το αυτί το echo effect.

Έπειτα από αρκετές δοκιμές, βρέθηκε πως για τιμές κοντά στο 2000, είχαμε ικανοποιητικά αποτελέσματα. Τελική τιμή επιλέχθηκε η P = 2021 και παρατίθεται το echoed σήμα που προκύπτει:



ε) Για την αποθήκευση των φιλτραρισμένων (echoed & dereverbed) σημάτων, χρησιμοποιήθηκε η εντολή **audiowrite()** .

στ) Στο ερώτημα 1.1 στ) χρησιμοποιήθηκαν οι τύποι με πίνακα Toeplitz. Ωστόσο, επειδή τα μεγέθη τώρα είναι πολύ μεγαλύτερα, η χρήση (υπολογισμός) του πίνακα Toeplitz δεν είναι εφικτή. Γι' αυτό, θα ακολουθήσουμε μία λίγο διαφορετική προσέγγιση.

Γνωρίζουμε ότι αν reverb_h η κρουστική απόκριση του φίλτρου αντίχησης και dereverb_h η κρουστική απόκριση του φίλτρου απαλοιφής της αντήχησης τότε θα ισχύει:

$$rever\,b_h * derever\,b_h = \delta[n]$$

Μετασχηματίζοντας την παραπάνω σχέση κατά Z παίρνουμε:

$$rever\,b_H \cdot derever\,b_H = 1 \Rightarrow derever\,b_H = \frac{1}{rever\,b_H}$$

Γνωρίζουμε όμως ότι για τη συνάρτηση μεταφορά ισχύει:

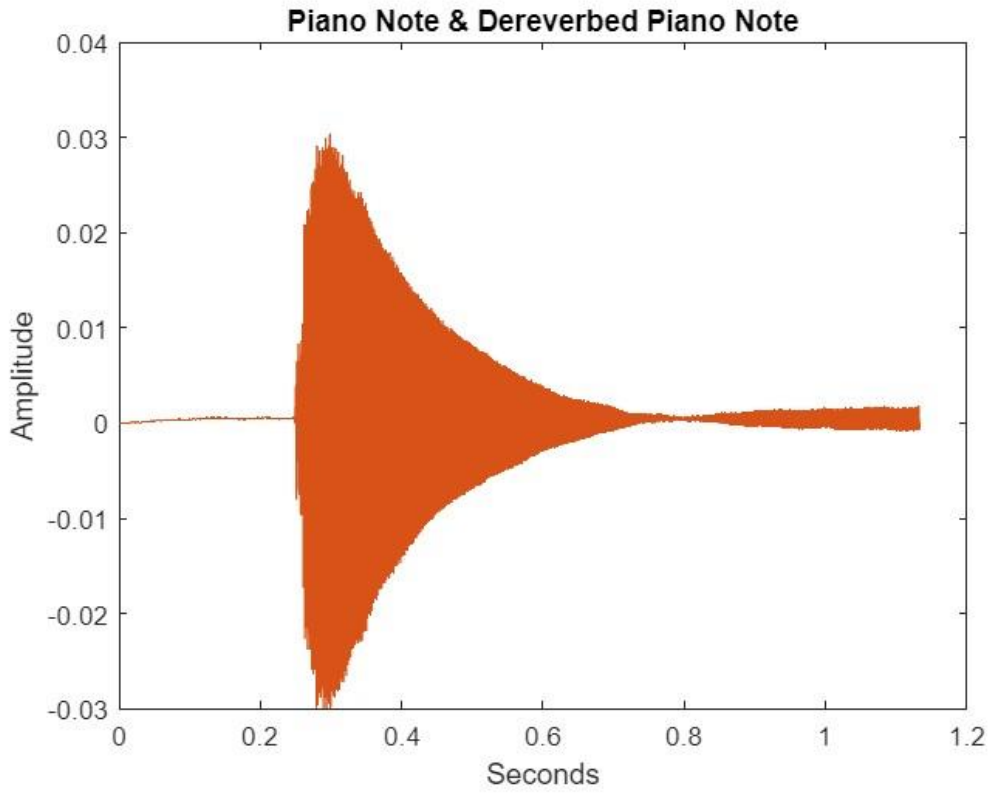
$$H(z) = \frac{\sum_{i=0}^{q} bi \cdot z^{-i}}{\sum_{i=0}^{p} ai \cdot z^{-i}}$$

Από τις τελευταίες δύο εξισώσεις, συμπεραίνουμε ότι:

$$ai_{piano_{dereverb}} = bi_{piano_{reverb}}$$
$$bi_{piano_{dereverb}} = ai_{piano_{reverb}}$$

Έχοντας βρει πλέον τα διανύσματα ai_piano_dereverb & bi_piano_dereverb μπορούμε να ακολουθήσουμε τη γνωστή διαδικασία:

- Με χρήση της εντολής **filter()** φιλτράρουμε το reverbed σήμα με το φίλτρο που κατασκευάσαμε και
- Με τις εντολές **plot()** και **hold on – hold off** σχεδιάζουμε το αρχικό και το dereverbed σήμα μαζί:



Όπως βλέπουμε, το ένα σήμα επικαλύπτει πλήρως το άλλο (ουσιαστικά τα δύο σήματα είναι ίδια πλέον) και αυτό ήταν αναμενόμενο, αφού το φίλτρο που φτιάξαμε απαλοίφει («ακυρώνει») πλήρως το φίλτρο αντήχησης. Άρα το φίλτρο είναι σωστό.

ζ) Η παράμετρος P του φίλτρου που σχεδιάσαμε στο ερώτημα 2.2 στ) ήταν ίση με 6615. Εφόσον θέλουμε να φτιάξουμε ένα φίλτρο απαλοιφής της αντήχησης με καθυστέρηση μεγαλύτερη κατά 5, 10 και 50 δείγματα, θα αυξήσουμε την παράμετρο P κατά 5, 10 και 50 αντίστοιχα. Δηλαδή, η παράμετρος P θα είναι ίση με 6620, 6625 και 6665 αντίστοιχα. Για να πετύχουμε το επιθυμητό αποτέλεσμα, θα συνδέσουμε σε σειρά το reverbation φίλτρο βαθμού 3 και καθυστέρησης 0.15sec που υλοποιήσαμε νωρίτερα με το dereverbation φίλτρο, επίσης βαθμού 3 , που θα σχεδιαστεί για απαλοιφή reverbation με καθυστέρηση μεγαλύτερη κατά 5/10/50 δείγματα από αυτήν του αρχικού φίλτρου.

Για την παραγωγή των νέων φίλτρων, θα ακολουθήσουμε παρόμοια διαδικασία με πριν. Πιο συγκεκριμένα, ορίζουμε αρχικά τους συντελεστές a_echo & b_echo του φίλτρου ηχούς ως εξής:

- Για το διάνυσμα a_echo, η πρώτη τιμή του θα είναι ίση με 1 και όλες οι υπόλοιπες θα είναι ίσες με 0.
- Για το διάνυσμα b_echo, η πρώτη τιμή θα είναι ίση με 0.6, η τελευταία ίση με 0.4 και όλες οι υπόλοιπες ίσες με 0.

Προφανώς, το μέγεθος κάθε διανύσματος θα ταυτίζεται με την παράμετρο P του κάθε φίλτρου. Στη συνέχεια, υπολογίζουμε τους συντελεστές ai_piano_reverb & bi_piano_reverb του φίλτρου αντήχησης και έπειτα, αντιστρέφοντας τους συντελεστές παίρνουμε τα διανύσματα ai_piano_dereverb & bi_piano_dereverb, δηλαδή θα ισχύει:

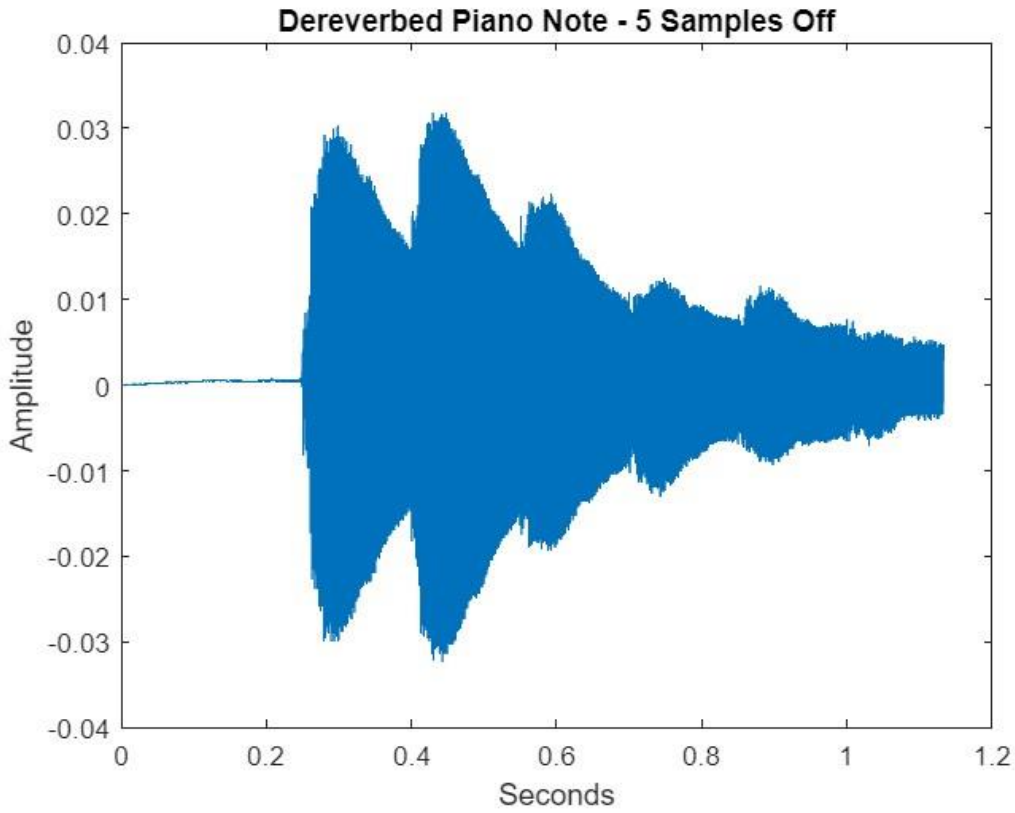
$$ai_{piano_{dereverb}} = bi_{piano_{reverb}}$$

$$bi_{piano_{dereverb}} = ai_{piano_{reverb}}$$

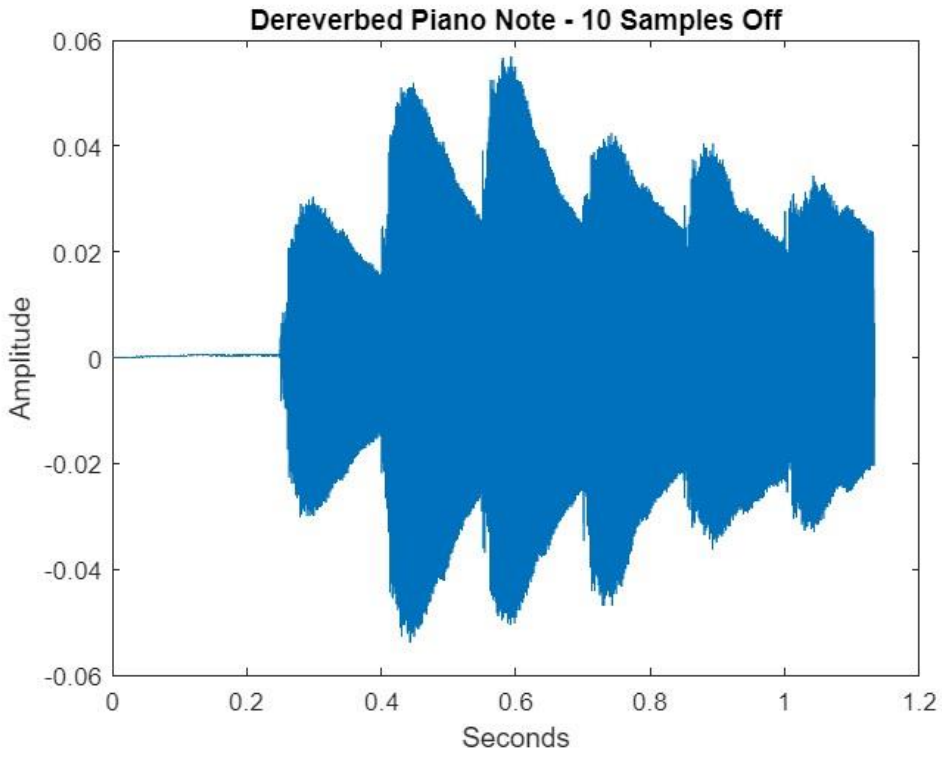
όπως εξηγήσαμε νωρίτερα.

Εφαρμόζοντας την παραπάνω μεθοδολογία για 5,10 και 50 δείγματα παραπάνω, παίρνουμε τα ακόλουθα διαγράμματα για τα σήματα:

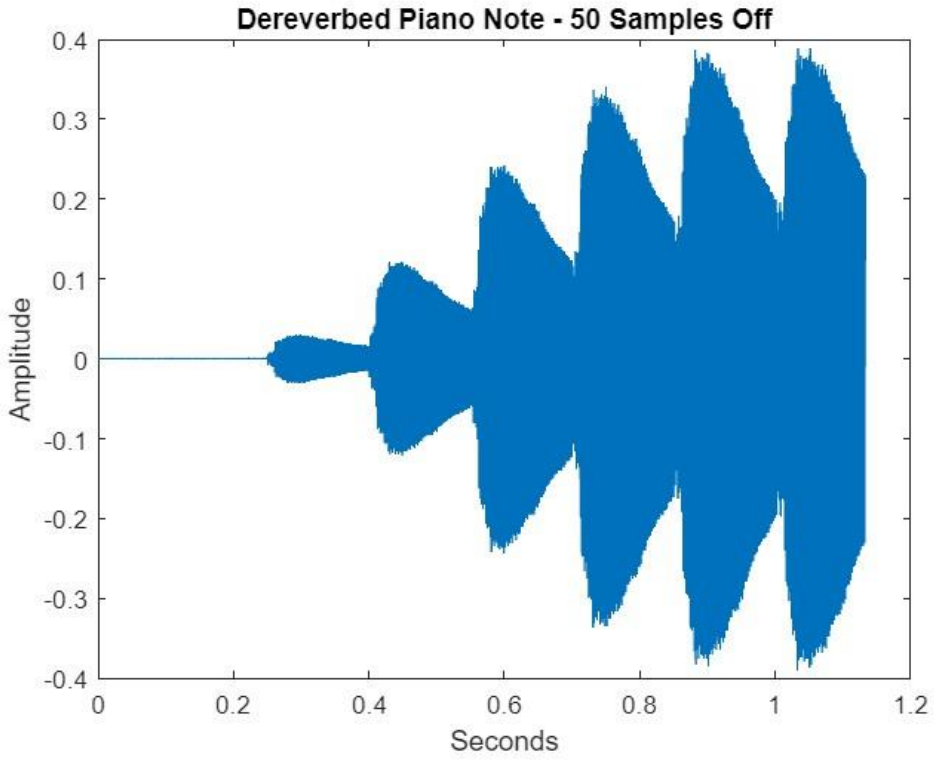
✓ Για 5 παραπάνω δείγματα (P = 6620):



✓ Για 10 παραπάνω δείγματα (P = 6625):



✓ Για 50 παραπάνω δείγματα (P = 6665):



Για τον υπολογισμό των συντελεστών ai_final & bi_final , δηλαδή των συντελεστών της συνολικής κρουστικής απόκρισης τους συστήματος (που είναι η συνέλιξη του h_piano_reverb και h_piano_dereverb), αξιοποιούμε τον τύπο της συνάρτησης μεταφοράς:

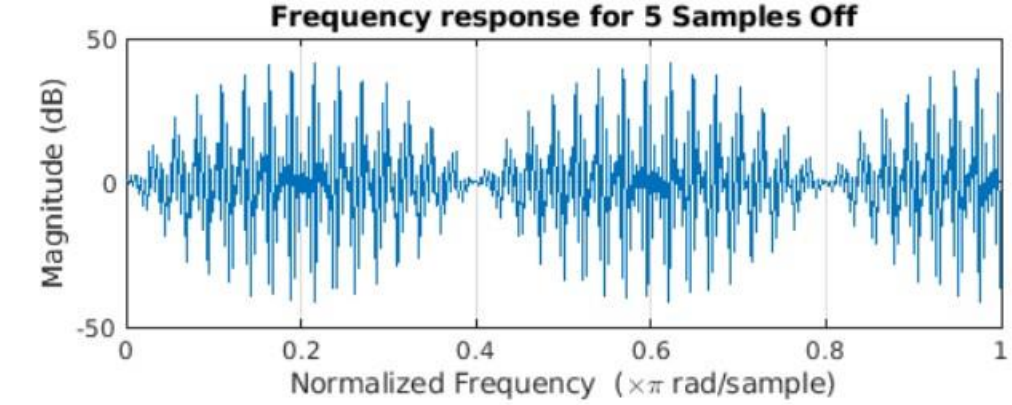
$$H(z) = \frac{\sum_{i=0}^{q} bi \cdot z^{-i}}{\sum_{i=0}^{p} ai \cdot z^{-i}}$$

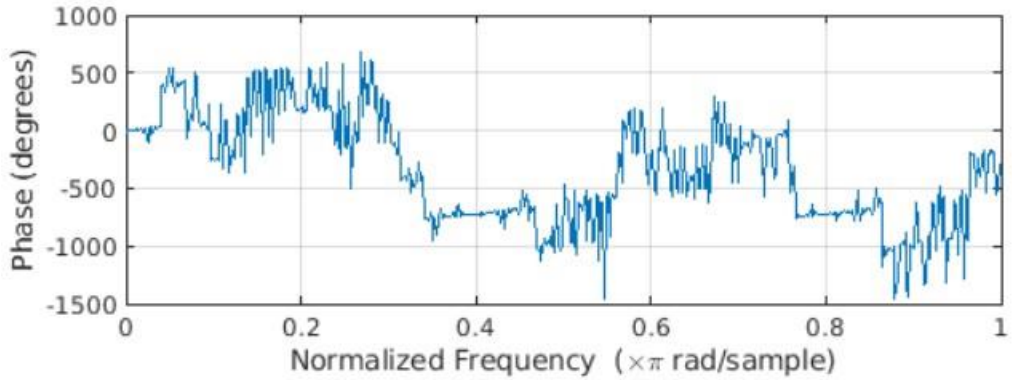
Όπως γνωρίζουμε, η συνέλιξη στο πεδίο του χρόνου ισοδυναμεί με πολλαπλασιασμό στο πεδίο Z (αναφέρθηκε και νωρίτερα) , όποτε συνδυάζοντας και την παραπάνω σχέση, καταλήγουμε στις εξής σχέσεις για τα διανύσματα ai_final & bi_final:

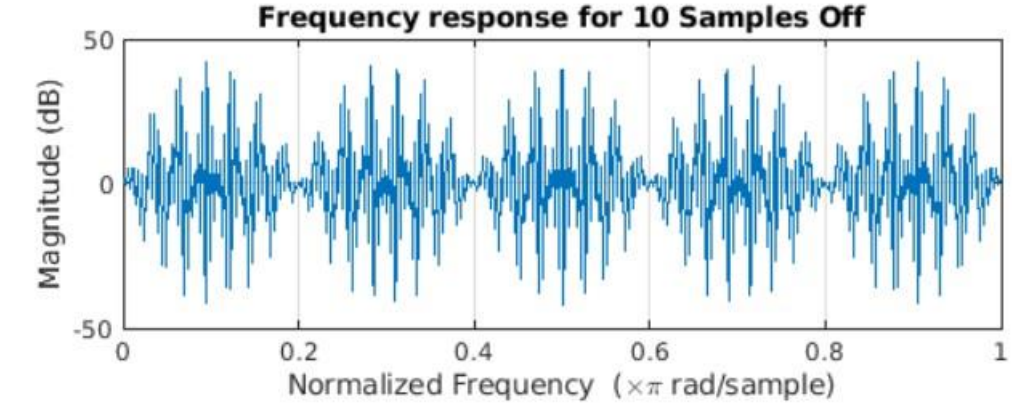
$$ai_{final} = ai_{piano_{reverb}} \cdot ai_{piano_{dereverb}}$$

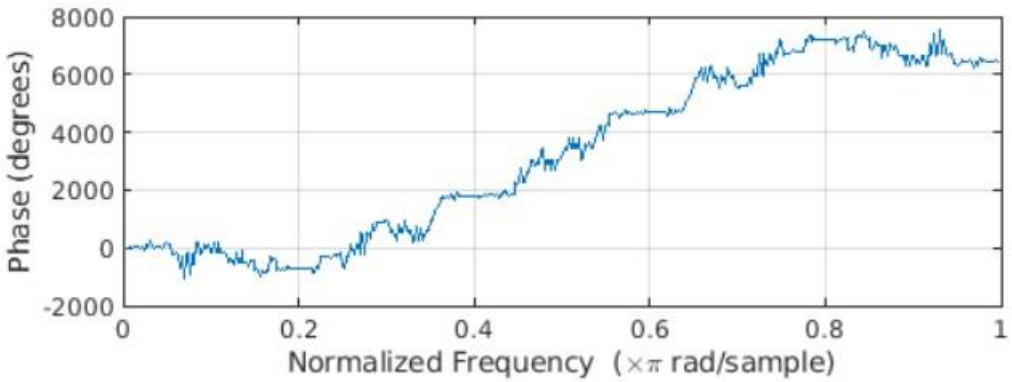
$$bi_{final} = bi_{piano_{reverb}} \cdot bi_{piano_{dereverb}}$$

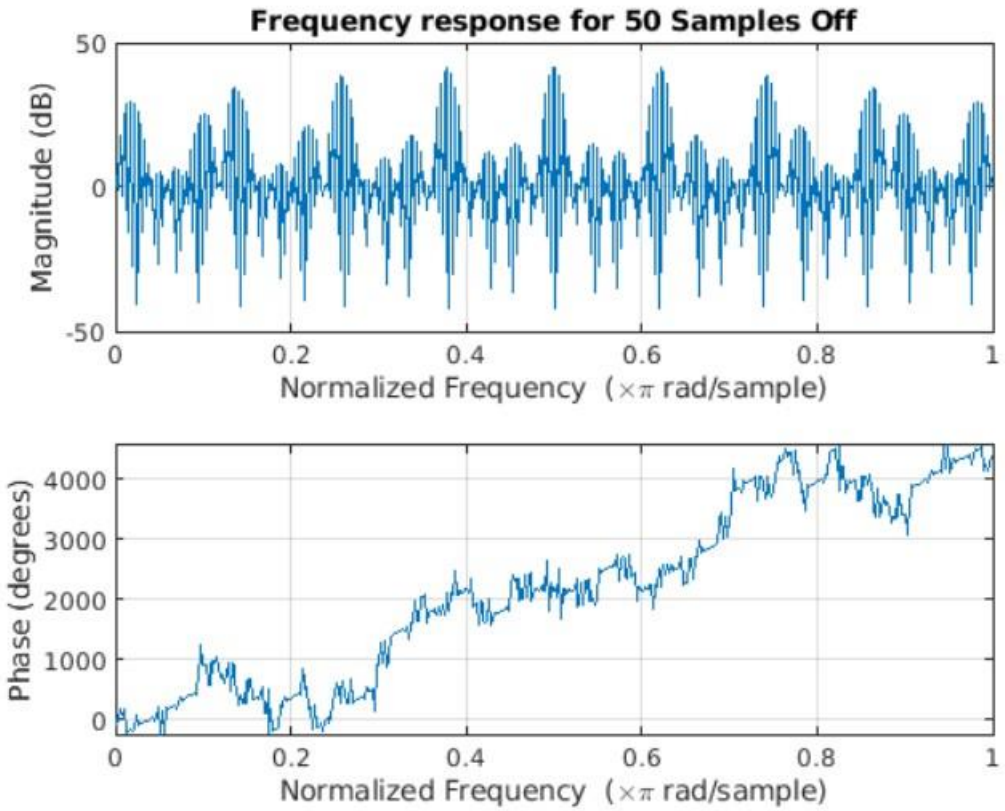
Αφού βρούμε τους συντελεστές/διανύσματα ai_final & bi_final , παιρνώντας τα στη συνάρτηση **freqz(),** παίρνουμε την απόκριση συχνότητας για κάθε φίλτρο:


Frequency response for 5 Samples Off
Magnitude (dB)
50
0
-50
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)

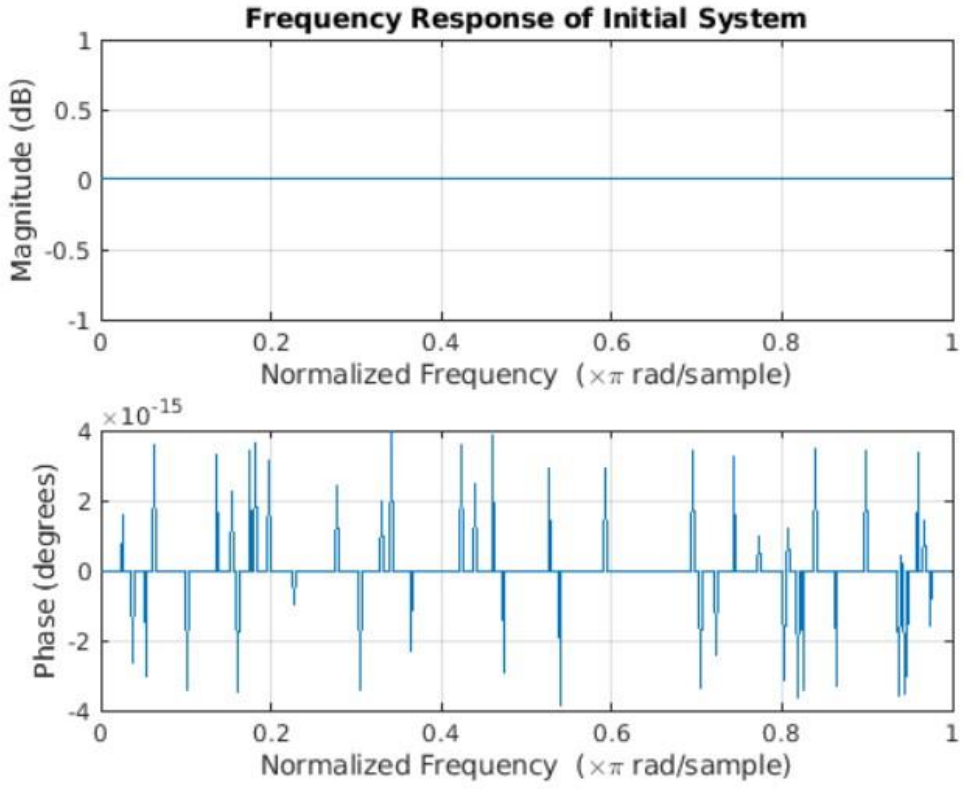

Phase (degrees)
1000
500
0
-500
-1000
-1500
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)


Frequency response for 10 Samples Off
Magnitude (dB)
50
0
-50
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)


Phase (degrees)
8000
6000
4000
2000
0
-2000
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)



Τέλος, παραθέτουμε και την απόκριση συχνότητας του αρχικού συστήματος:



Παρατηρούμε, πως η απόκριση πλάτους – φάσης του αρχικού συστήματος είναι μηδέν. Αυτό είναι αναμενόμενο, αφού το σύστημα αποτελείται από την εν σειρά σύνδεση του φίλτρου αντήχησης με το φίλτρο απαλοιφής της αντήχησης. Δηλαδή, το ένα φίλτρο «ακυρώνει» το άλλο και συνεπώς το τελικό σύστημα δεν προκαλεί καμία μεταβολή στο σήμα. Αντίθετα, στα υπόλοιπα διαγράμματα απόκρισης πλάτους – φάσης δεν ισχύει αυτό (δηλαδή είναι διάφορα του μηδενός) , καθώς το φίλτρο απαλοιφής της αντήχησης δεν αντιστρέφει την αντήχηση που προκαλεί το πρώτο φίλτρο. Τέλος, σημειώνεται ότι η συχνότητα της απόκρισης πλάτους αυξάνεται όσο αυξάνουμε τον αριθμό των παραπάνω δειγμάτων, κάτι που επιβεβαιώνεται και μέσα από τα διαγράμματα.