Σήματα και Συστήματα - Εργασία MATLAB (2019-2020)

Κυριακόπουλος Γεώργιος

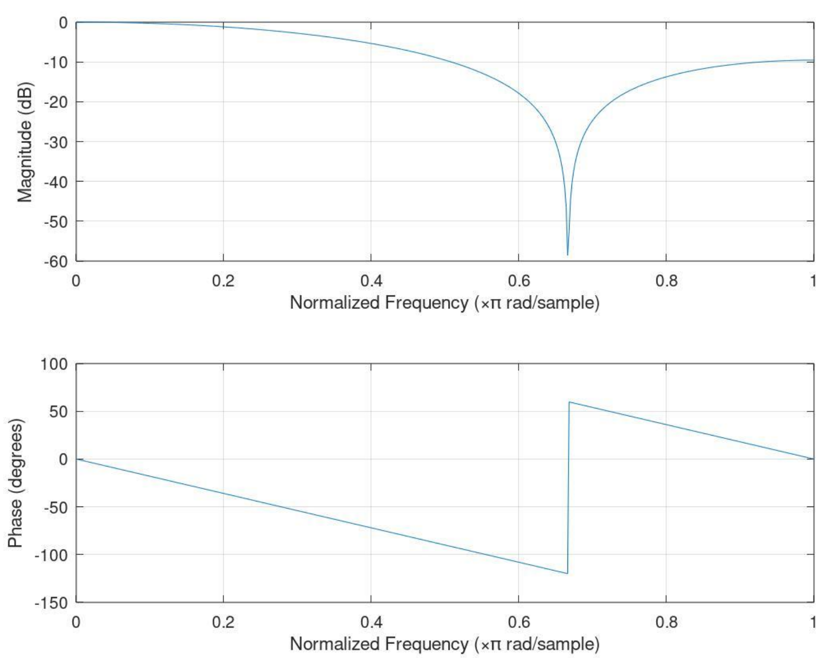
el18153

1.1 Σχεδίαση Βαθυπερατών Φίλτρων (Κινούμενου Μέσου)

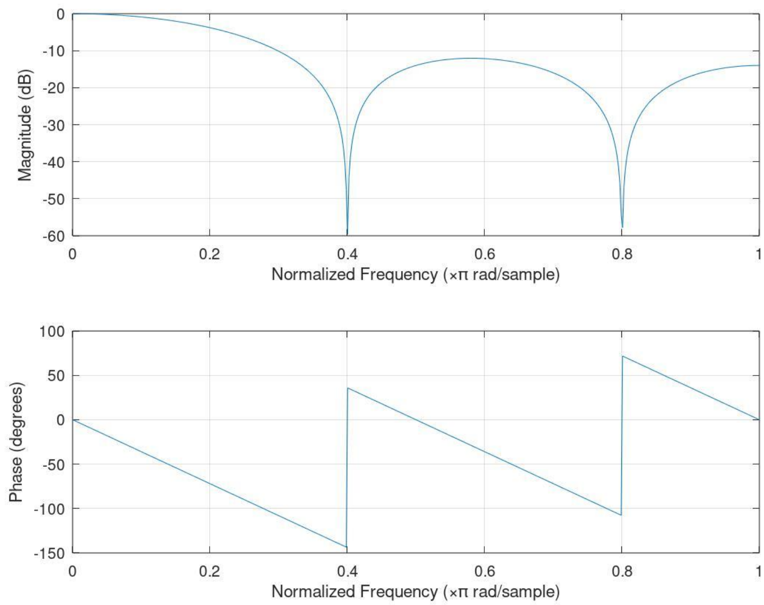
α) Ορίζουμε απλά τα διανύσματα ai,bi για τις 3 περιπτώσεις των φίλτρων μας σύμφωνα με το δοσμένο τύπο για τα φίλτρα κινούμενου μέσου.

β) Με χρήση της εντολής freqz(b,a) σχεδιάζουμε τις αποκρίσεις πλάτους και φάσης για τα 3 φίλτρα.

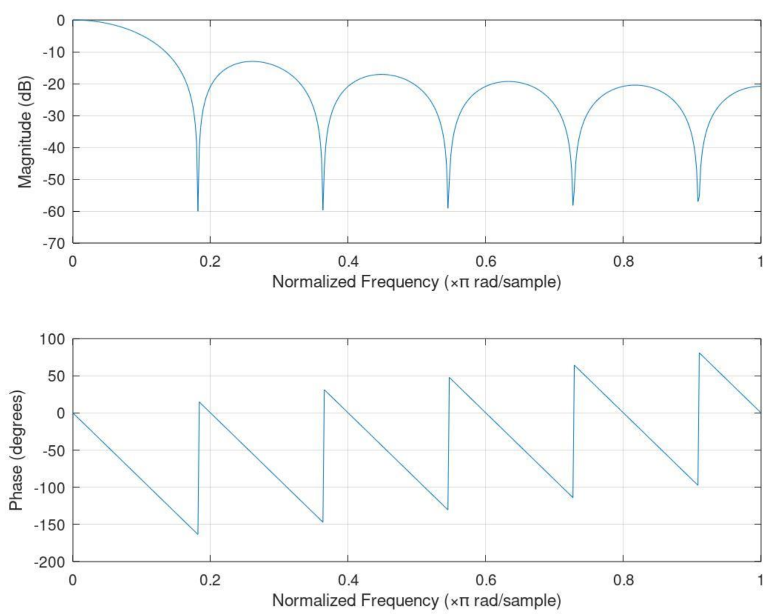
Για το πρώτο φίλτρο με N = 2 έχουμε:



Για το δεύτερο φίλτρο με N = 4 έχουμε:



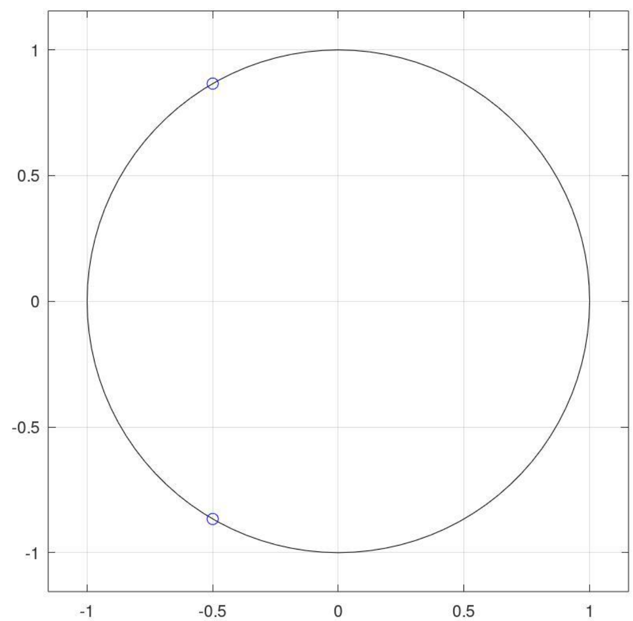
Για το τρίτο φίλτρο με N = 10 έχουμε:



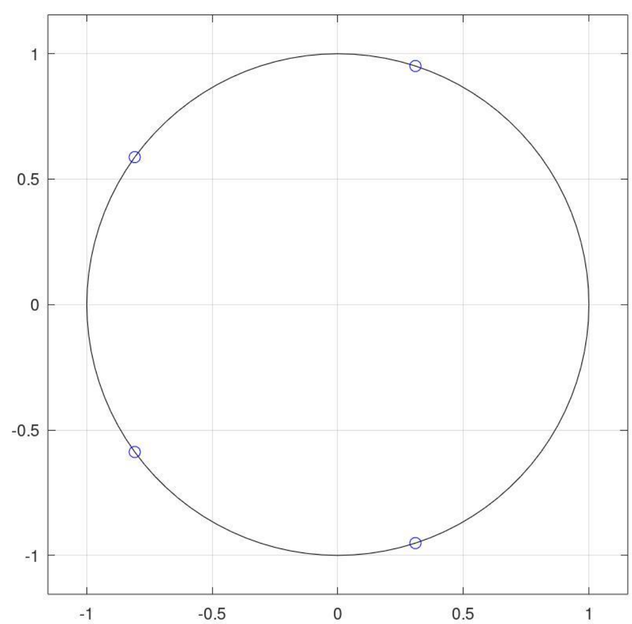
Καθώς αυξάνεται το N, παρατηρούμε ότι το πλάτος της απόκρισης συχνότητας μειώνεται όλο και περισσότερο και όσο προχωράμε στον άξονα της συχνότητας. Επίσης βλέπουμε περισσότερα άλματα προς τα κάτω (πλησιάζει το μηδενικό πλάτος), κάτι που οφείλεται στους όλο και περισσότερους πόλους που προστίθενται στο σύστημα με την αύξηση του N. Η φάση έχει μεγαλύτερες διακυμάνσεις και περισσότερα άλματα κατά π όσο αυξάνεται το N, επίσης.

γ) Με χρήση της συνάρτησης tf2zp(b,a) βρίσκουμε από τα διανύσματα b,a τα μηδενικά και τους πόλους των φίλτρων μας και στη συνέχεια με τη συνάρτηση zplane(z,p) σχεδιάζουμε το διάγραμμα πόλων – μηδενικών του φίλτρου.

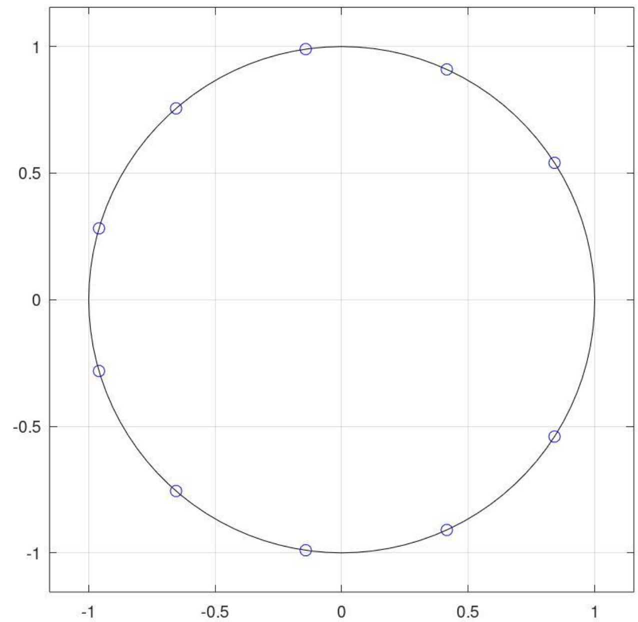
Για το πρώτο φίλτρο έχουμε:



Για το δεύτερο φίλτρο έχουμε:



Για το τρίτο φίλτρο έχουμε:

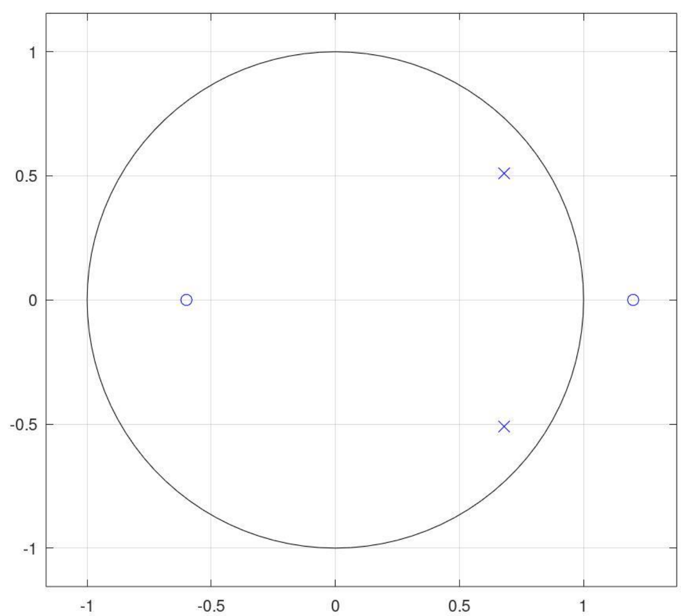


Όπως ήταν αναμενόμενο, με βάση τη συνάρτηση μεταφοράς (διάνυσμα a = 1 για όλα τα φίλτρα) δεν έχουμε κανένα πόλο, αλλά μόνο μηδενικά ισάριθμα με το N, συζυγή ανά 2 και όλα πάνω στο μοναδιαίο κύκλο.

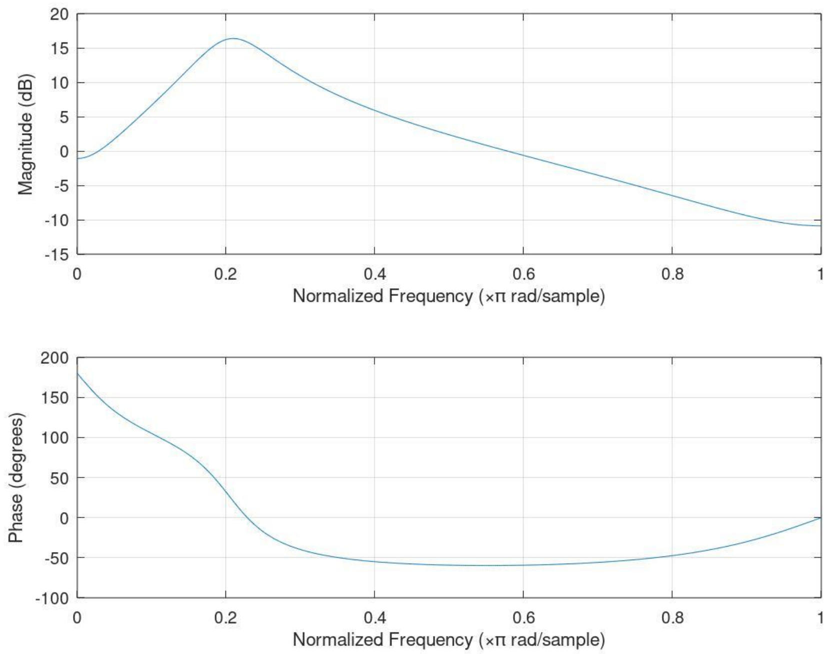
1.2 Σχεδίαση Ζωνοπερατών Φίλτρων

α) Ορίζουμε το διάνυσμα των πόλων και των μηδενικών και στη συνέχεια με χρήση της εντολής zplane(z,p) σχεδιάζουμε το διάγραμμα πόλων – μηδενικών του φίλτρου. Μετά με χρήση της συνάρτησης zp2tf(z,p,1) βρίσκουμε τα διανύσματα b,a που αντιστοιχούν στο φίλτρο αυτό.

Για το διάγραμμα πόλων – μηδενικών του φίλτρου έχουμε ακριβώς του πόλους και τα μηδενικά στις θέσεις που περιμέναμε με βάση και την εκφώνηση και την υλοποίηση του ερωτήματος:



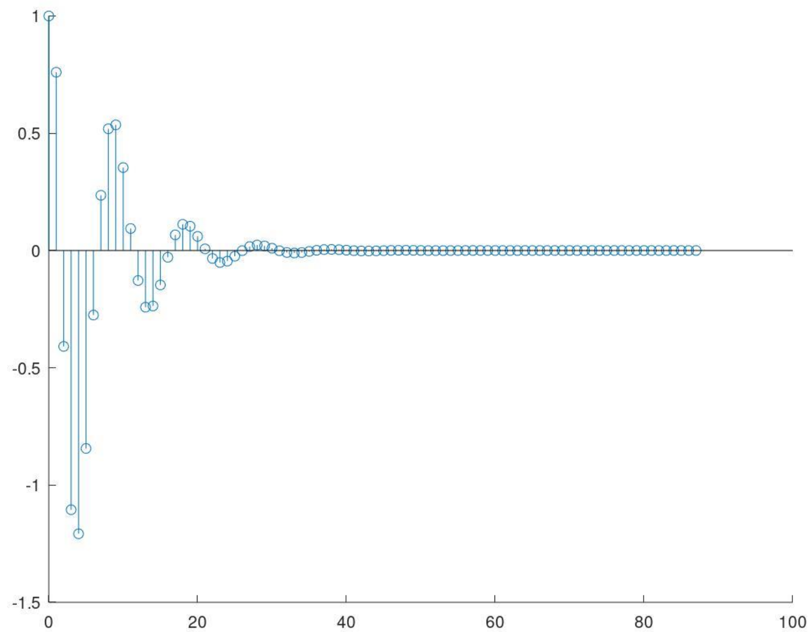
β) Με τη χρήση της συνάρτησης freqz(b,a) και με βάση τα b,a που προέκυψαν από το προηγούμενο ερώτημα με την zp2tf(z,p,1) σχεδιάζουμε την απόκριση πλάτους και φάσης του παραπάνω φίλτρου:



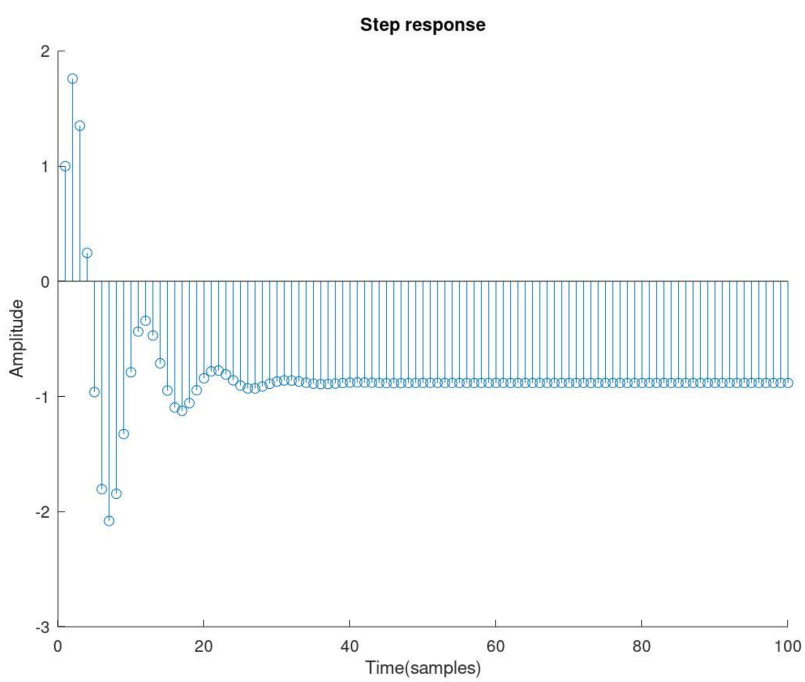
Από το διάγραμμα αυτό και κυρίως με βάση την απόκριση πλάτους καταλαβαίνουμε ότι έχουμε ένα ζωνοπερατό φίλτρο, αφού ενισχύεται μία συγκεκριμένη ζώνη συχνοτήτων και η απόκριση φάσης του φίλτρου μοιάζει στην κρίσιμη περιοχή με αυτήν ενός ζωνοπερατού.

γ) Με χρήση της impz(b,a) και της stepz(b,a,100) σχεδιάζουμε την κρουστική και βηματική απόκριση του φίλτρου για 100 δείγματα, με βάση τα b,a που προήλθαν από τη zp2tf(z,p,1).

Για την κρουστική απόκριση έχουμε:

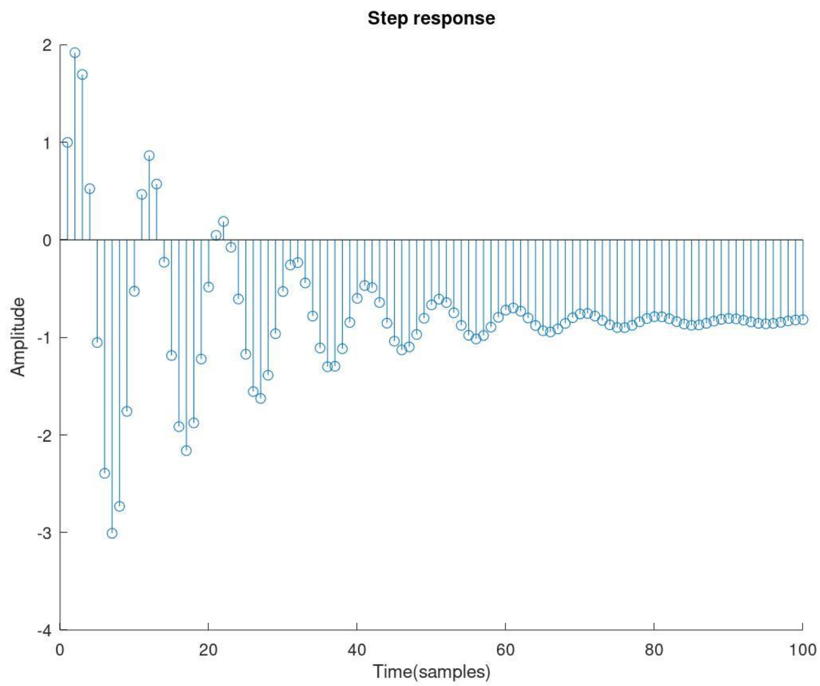


Για τη βηματική απόκριση έχουμε:

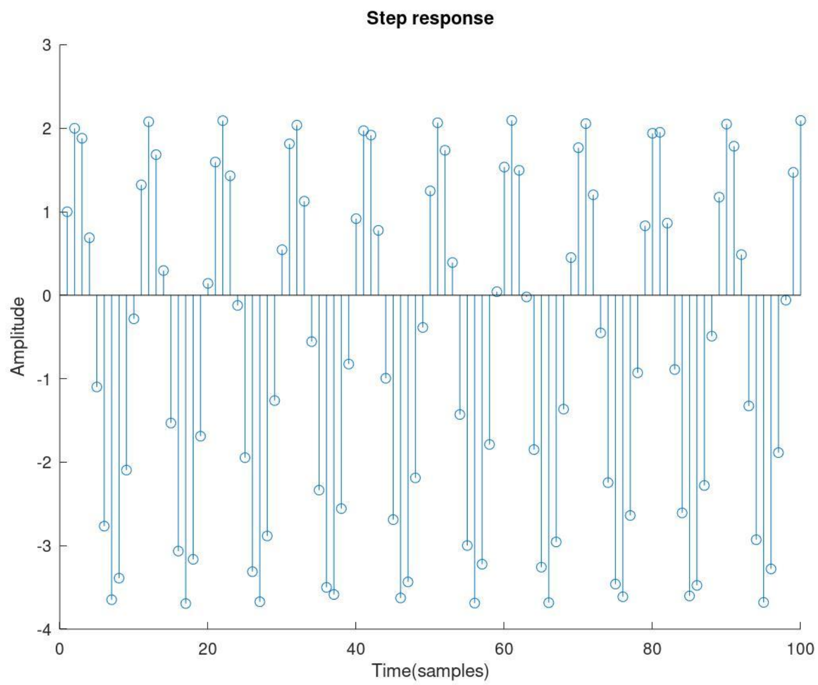


δ) Αρχικά, ορίζουμε τα νέα διανύσματα με τους πόλους ενώ στη συνέχεια βρίσκουμε τα αντίστοιχα διανύσματα b,a των 3 αυτών νεών φίλτρων με χρήση της zp2tf(z,p,1). Στη συνέχεια, σχεδιάζουμε τη βηματική απόκριση για κάθε ένα από αυτά τα καινούρια φίλτρα με τη χρήση της stepz(b,a,100) και τέλος σχεδιάζουμε την απόκριση πλάτους για το πρώτο από αυτά.

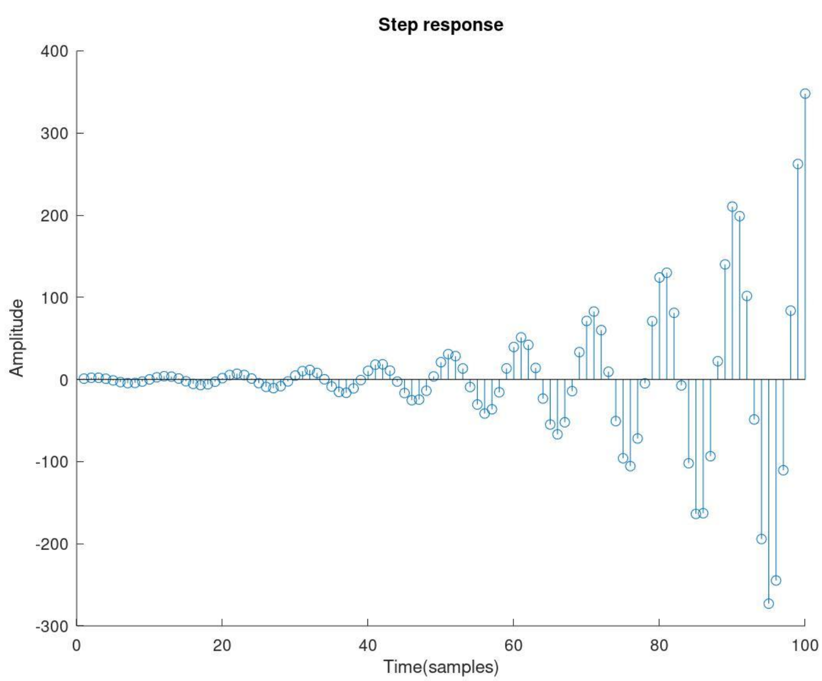
Για το πρώτο φίλτρο έχουμε:



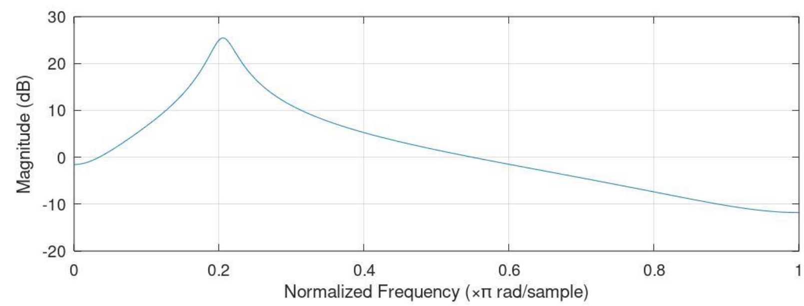
Για το δεύτερο φίλτρο έχουμε:

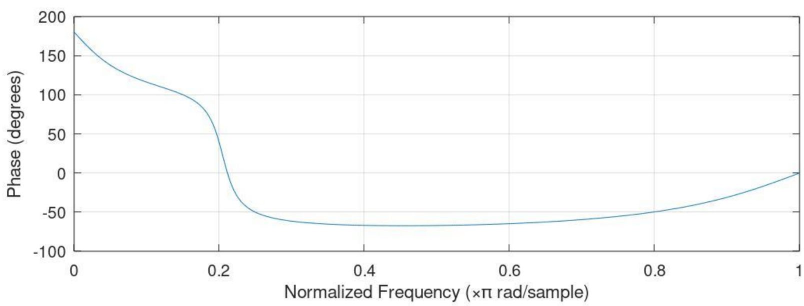


Για το τρίτο φίλτρο έχουμε:

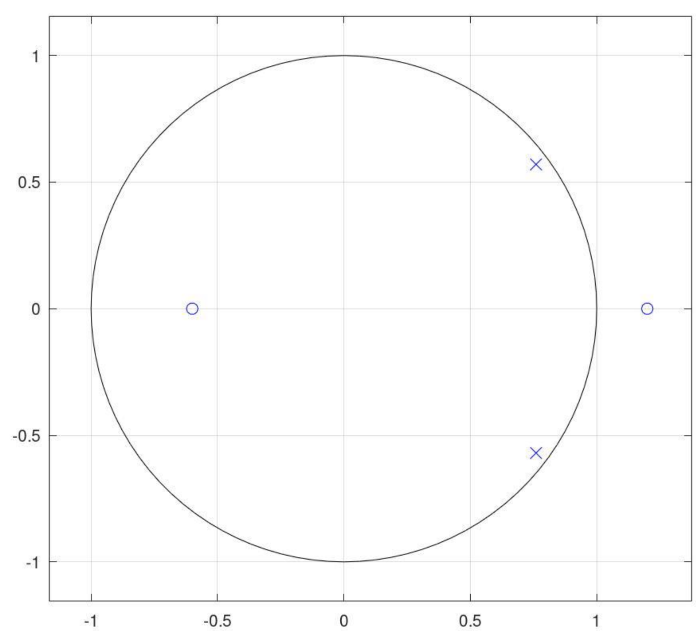


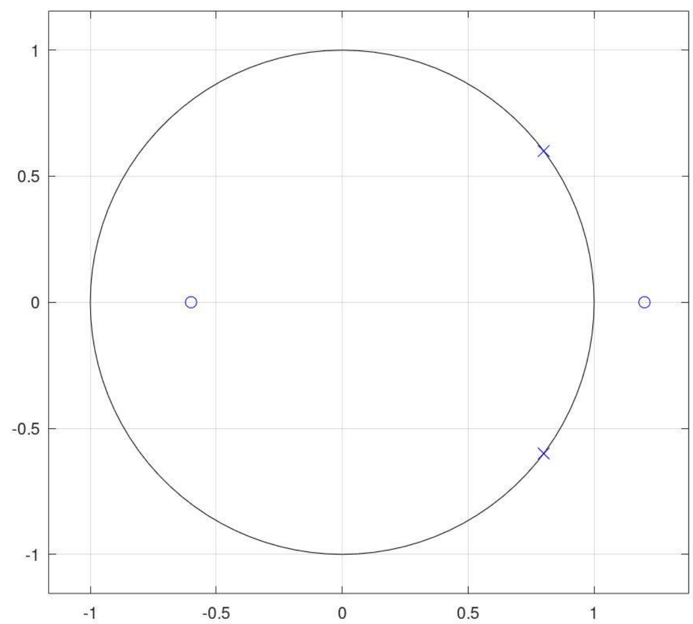
Επίσης, για την απόκριση πλάτους (και φάσης) του πρώτου φίλτρου έχουμε:

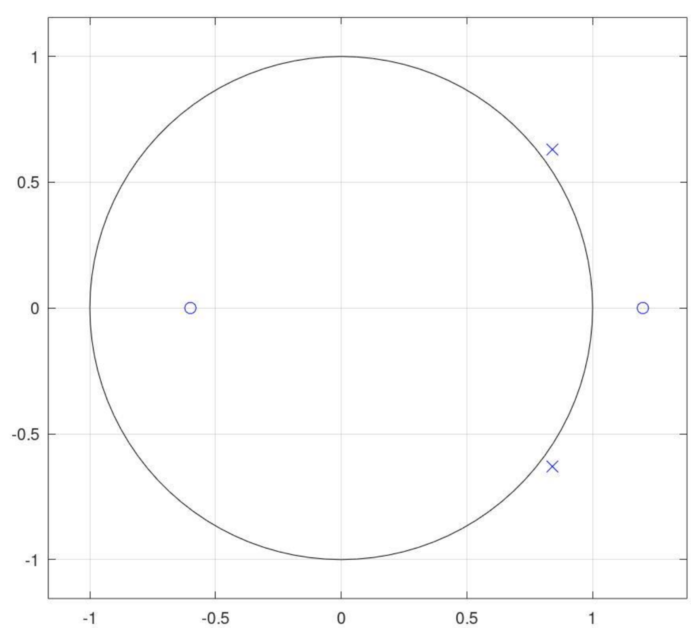




Ακολουθούν με τη σειρά τα διαγράμματα πόλων – μηδενικών για τα 3 νέα αυτά φίλτρα:







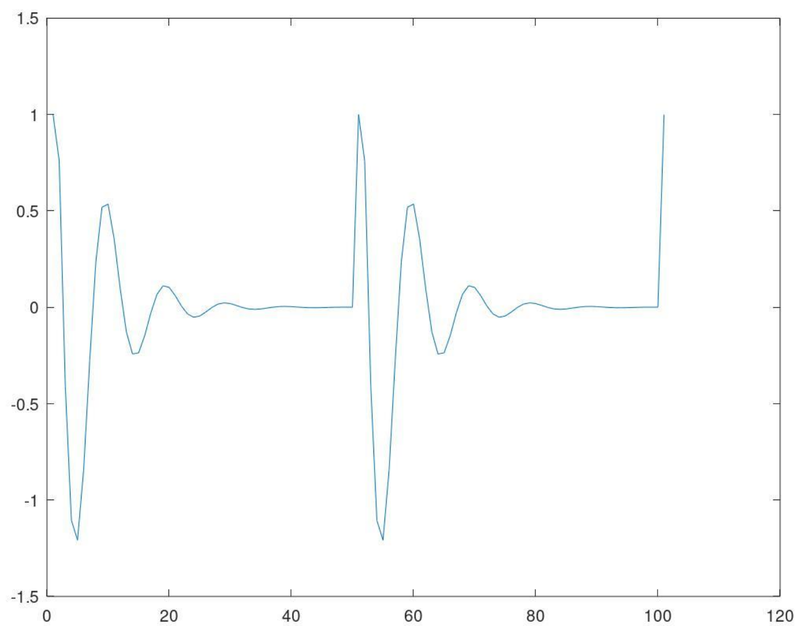
Στο πρώτο φίλτρο, όπως βλέπουμε έχουμε πόλους μέσα στο μοναδιαίο κύκλο και επομένως το σύστημα μας είναι ευσταθές, για αυτό παίρνουμε και την συγκεκριμένη βηματική απόκριση.

Στο δεύτερο φίλτρο, έχουμε πόλους μονής πολλαπλότητας πάνω στο μοναδιαίο κύκλο και άρα το σύστημα μας είναι οριακά ευσταθές, για αυτό έχουμε την αντίστοιχη βηματική απόκριση.

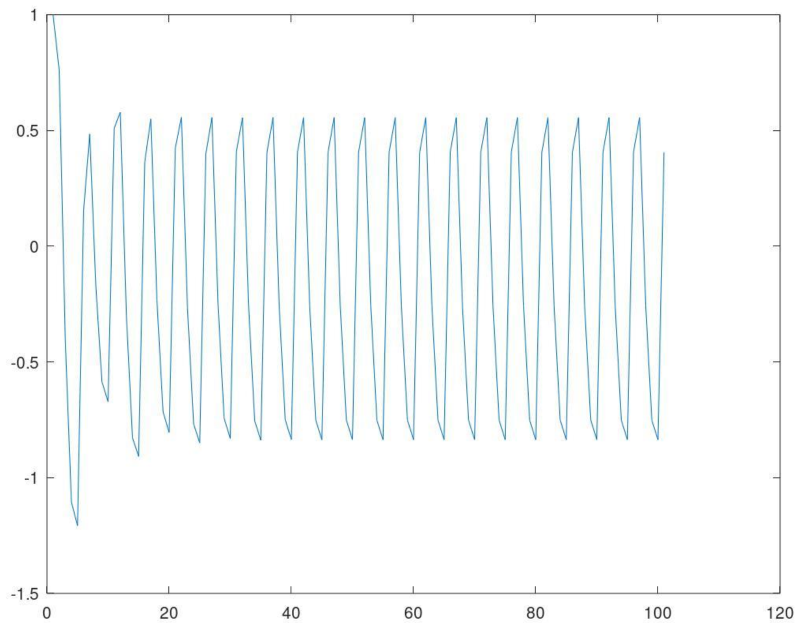
Τέλος, στο τρίτο φίλτρο, οι πόλοι βρίσκονται εκτός του μοναδιαίου κύκλου με αποτέλεσμα να έχουμε ένα ασταθές σύστημα, όπως φαίνεται και στην τελευταία βηματική απόκριση.

ε) Χρησιμοποιώντας την gensig(“pulse”,T,100,1), σύμφωνα και με την εκφώνηση για την παραγωγή της κρουστικής παλμοσειράς και στη συνέχεια διεγείροντας το σύστημα μέσω της εντολής filter(b,a,[pulse]) παίρνουμε την έξοδο και σχεδιάζουμε την πρώτη στήλη αυτής (δημιουργούνται 2 λόγω της δισδιάστατης μορφής της pulse, αλλά μας ενδιαφέρει μόνο η πρώτη).

Για περίοδο T = 50s έχουμε:

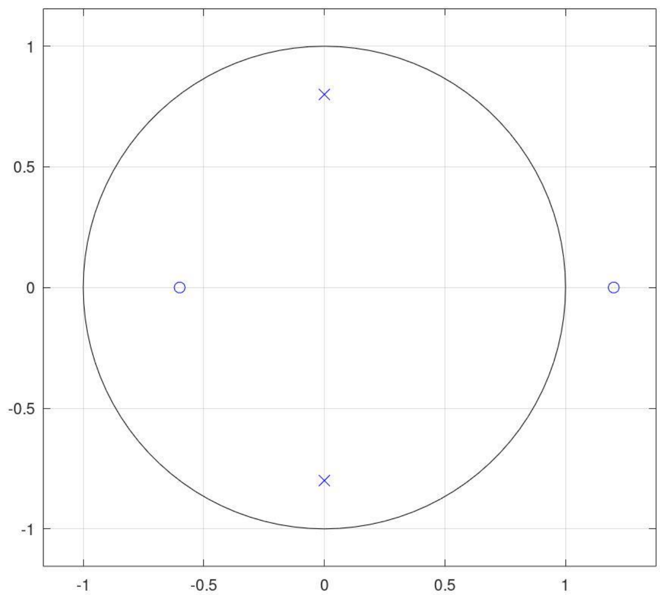


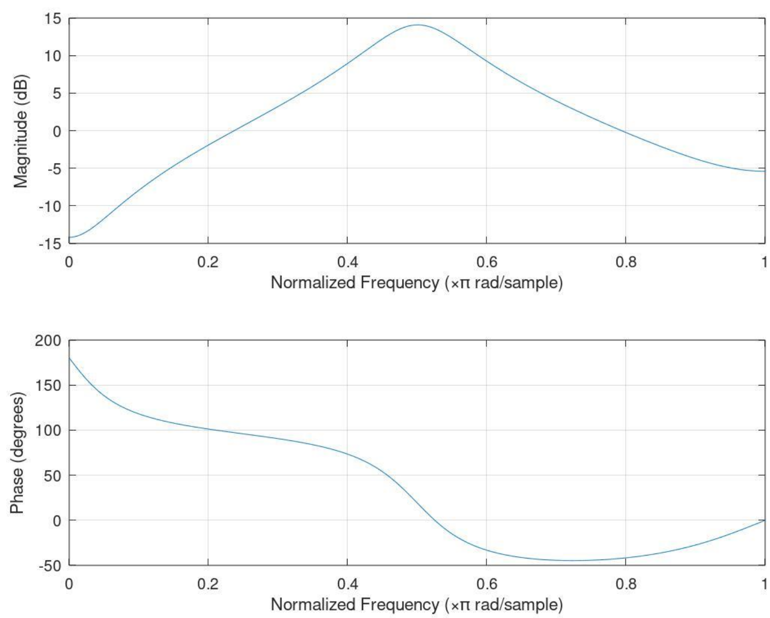
Για περίοδο T = 5s έχουμε:



Παρατηρούμε ότι σε διάρκεια σήματος 100s η κρουστική παλμοσειρά με περίοδο 50s παράγει 2 κρουστικές αποκρίσεις, ενώ η κρουστική παλμοσειρά με περίοδο 5s παράγει 20 κρουστικές αποκρίσεις, όπως είναι και αναμενόμενο.

στ) Ορίζουμε το νέο διάνυσμα των πόλων και στη συνέχεια με χρήση της zplane(z,p) σχεδιάζουμε το διάγραμμα πόλων – μηδενικών για αυτό το φίλτρο. Έπειτα με χρήση της zp2tf παίρνουμε τα διανύσματα b,a του νέου αυτού φίλτρου και σχεδιάζουμε την απόκριση πλάτους και φάσης.





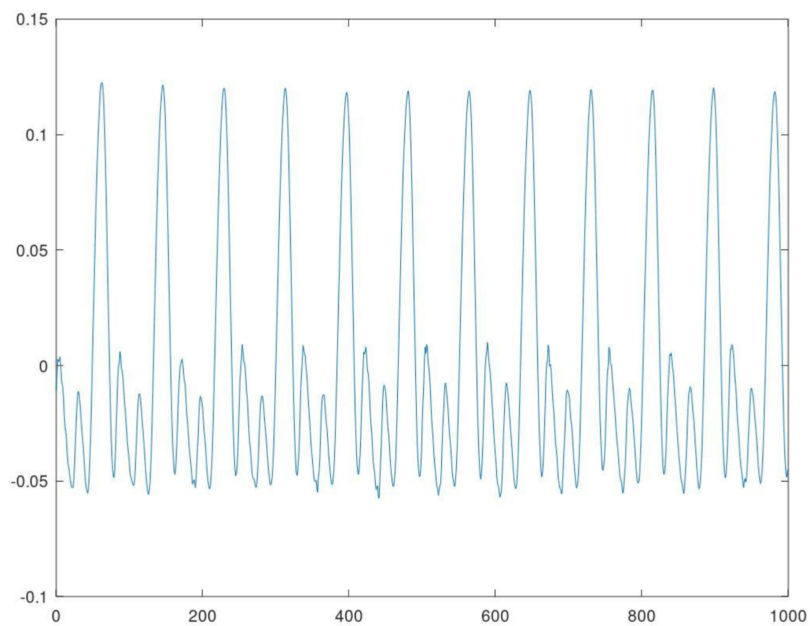
Παρατηρούμε μία μετατόπιση της ζώνης διέλευσης του νέου φίλτρου και αντίστοιχη μετατόπιση και στην απόκριση της φάσης προς τα δεξιά (σε μεγαλύτερη συχνότητα δηλαδή).

2.1. Ανάλυση Μουσικών Σημάτων

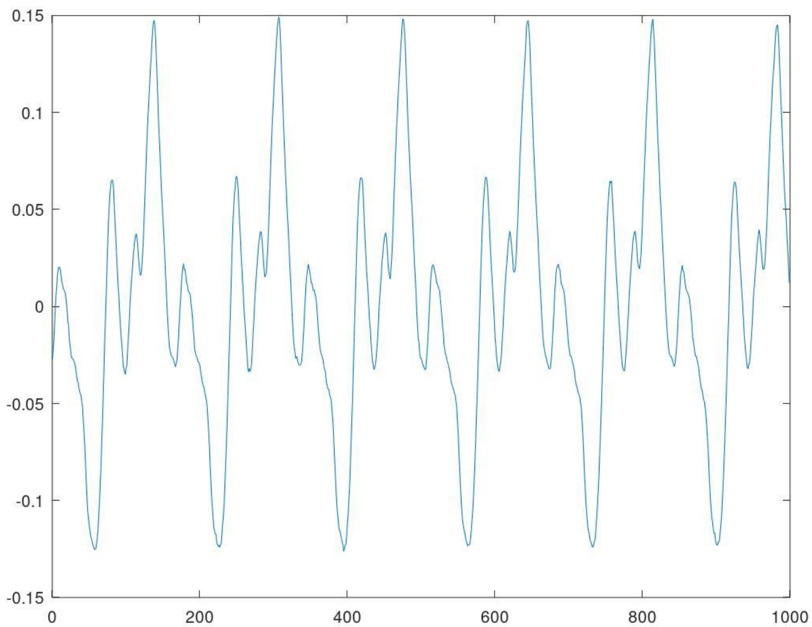
α) Φορτώνουμε με τη χρήση της εντολής audioread('filename.wav') τα 3 αρχεία μας. Έπειτα τρέχουμε την εντολή sound(file,fs) και ακούμε τα 3 σήματα.

β) Διαλέγουμε τυχαία για το κάθε αρχείο ένα απόσπασμα μήκους 1000 δειγμάτων και σε άξονα μήκους 1000 δειγμάτων σχηματίζουμε τις γραφικές για τα 3 αυτά σήματα.

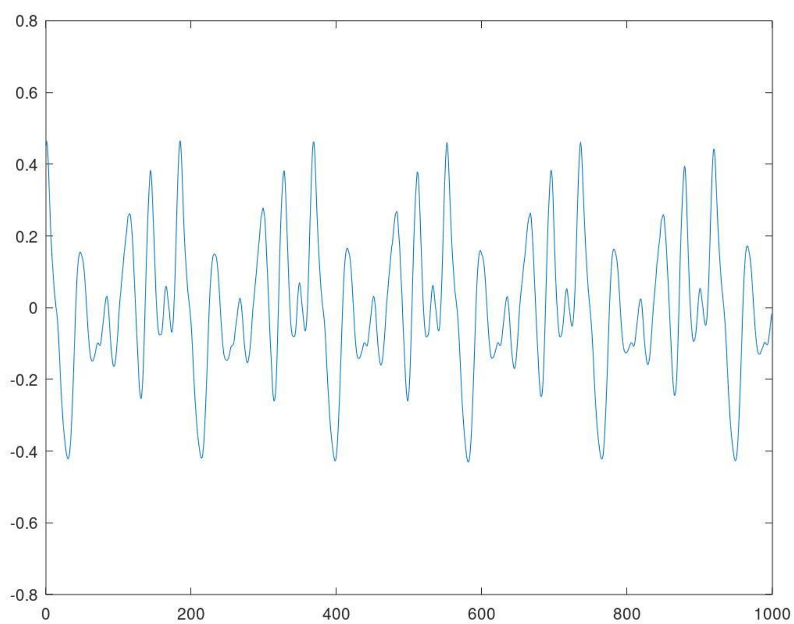
Για το flute βλέπουμε ότι είναι περιοδικό με περίοδο περίπου 85 δείγματα ή 1.93ms.



Για το clarinet βλέπουμε ότι είναι περιοδικό με περίοδο περίπου 170 δείγματα ή 3.85ms.

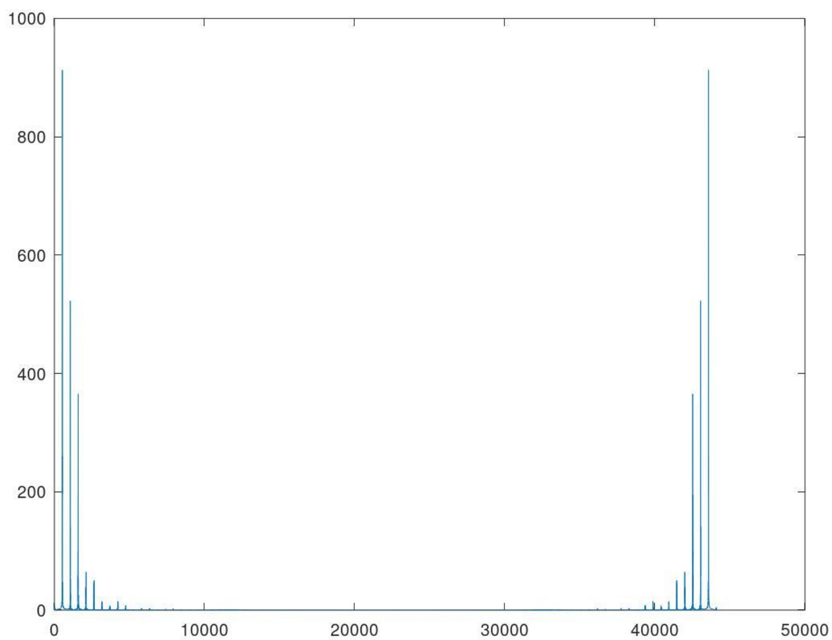


Για το cello βλέπουμε ότι είναι περιοδικό με περίοδο περίπου 185 δείγματα ή 4.19ms.

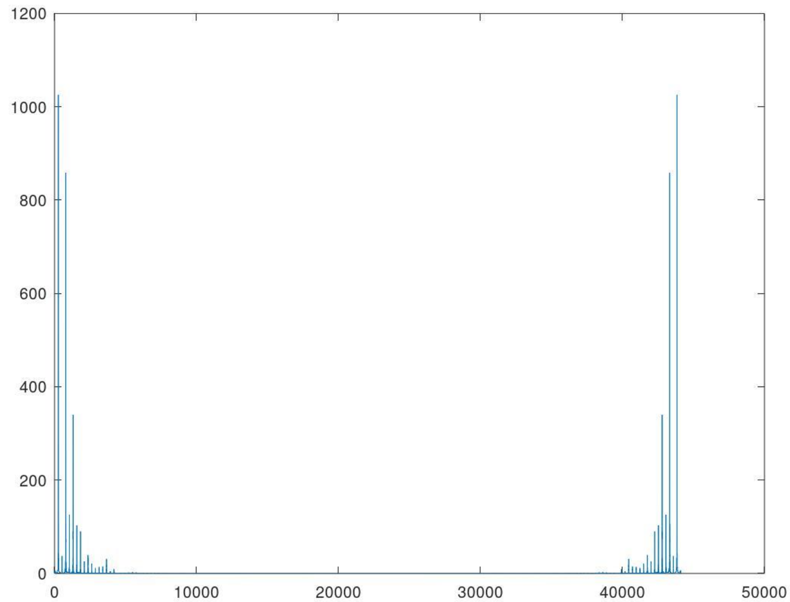


γ) Υπολογίζουμε το μετασχηματισμό με χρήση της εντολής fft(y,fs), ώστε να μπορούμε να υπολογίζουμε εύκολα και τη θεμελιώδη συχνότητα και λαμβάνουμε το απόλυτο του για να σχηματίσουμε το φάσμα του σήματος, του οποίου τη γραφική παράσταση παίρνουμε.

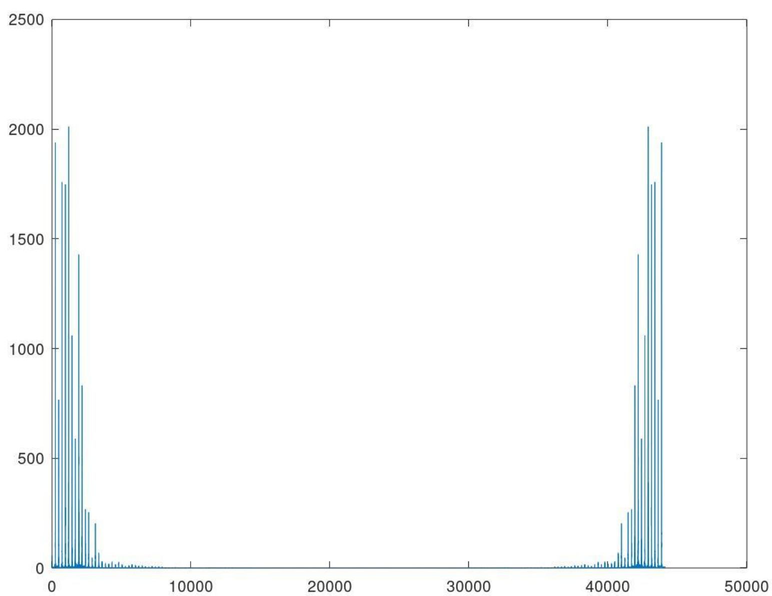
Για το flute έχουμε θεμελιώδη συχνότητα 529Hz. 529*1.93*10^-3 = 1.02 περίπου, αρκετά κοντά στην μονάδα, άρα επιβεβαιώνεται η σχέση μεταξύ θεμελιώδους συχνότητας και περιόδου.



Για το clarinet έχουμε θεμελιώδη συχνότητα 262Hz. 626*3.85*10^-3 = 1.009 περίπου, αρκετά κοντά στη μονάδα, άρα επιβεβαιώνεται η σχέση μεταξύ θεμελιώδους συχνότητας και περιόδου.



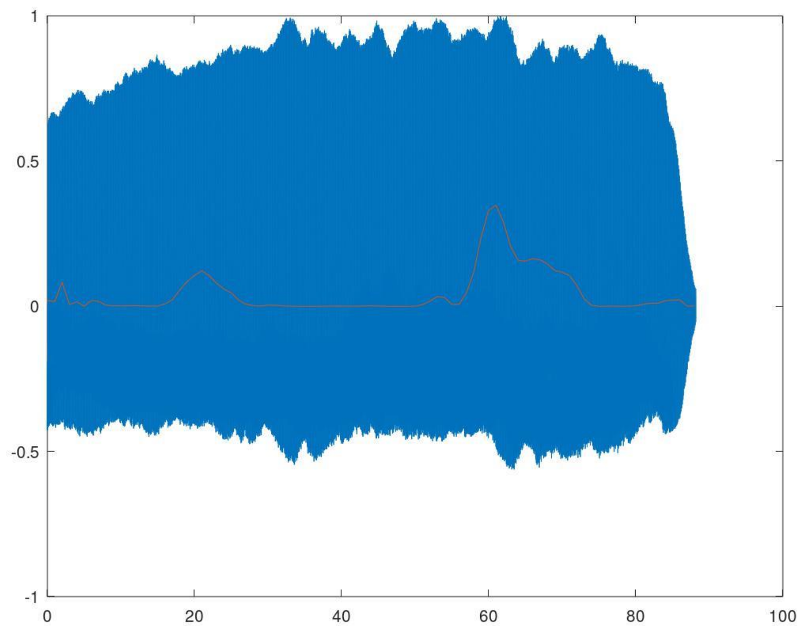
Για το cello έχουμε θεμελιώδη συχνότητα 241Hz. 241*4.19*10^-3 = 1.01 περίπου, αρκετά κοντά στη μονάδα, άρα επιβεβαιώνεται η σχέση μεταξύ θεμελιώδους συχνότητας και περιόδου.



Γενικά παρατηρούμε ότι όσο πιο βαθύς και μπάσος είναι ο ήχος τόσες περισσότερες αρμονικές υψηλότερων τάξεων εμφανίζονται, με το cello να εμφανίζει αισθητά περισσότερες αρμονικές σε σχέση με τα άλλα δύο.

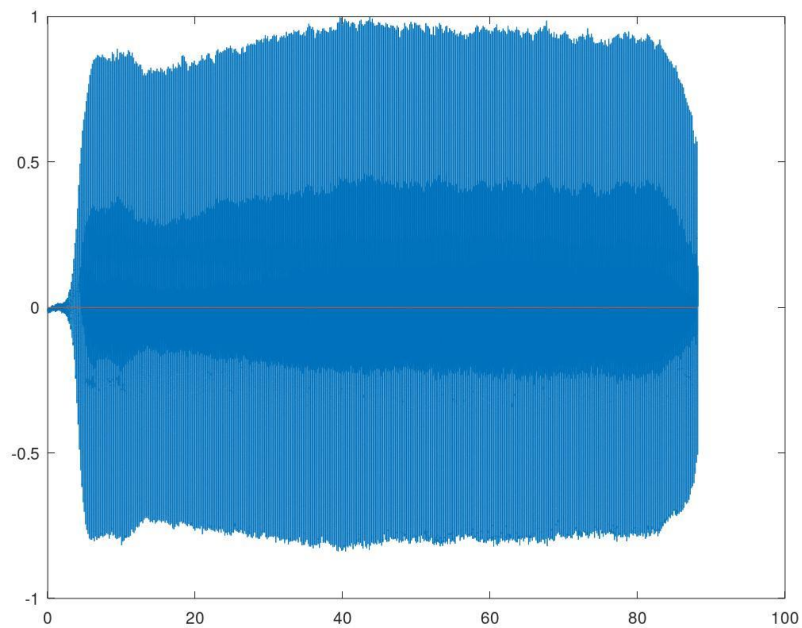
δ) Αρχικά κανονικοποιούμε κάθε σήμα στο [-1,1] βρίσκοντας τη μέγιστη τιμή του σήματος και διαιρώντας το με την απόλυτη τιμή της τιμής αυτής. Στη συνέχεια, χωρίζουμε με τη βοήθεια της εντολής buffer(x,1000) το σήμα μας σε χιλιάδες δειγμάτων και αναστρέφουμε τον πίνακα για να γίνει σωστά ο υπολογισμός της συνέλιξης στη συνέχεια, ενώ ορίζουμε νέο πίνακα ενέργειας που θα γεμίσουμε μέσω του υπολογισμού της συνέλιξης που ορίζεται κάθε φορά για κάθε χιλιάδα δειγμάτων, χρησιμοποιώντας την εντολή conv(a,b). Τέλος, ορίζουμε τους δύο άξονες δειγμάτων, έναν για το κλιμακωμένο σήμα μας και έναν για την ενέργεια του και χαράζουμε την κοινή γραφική.

Για το flute:

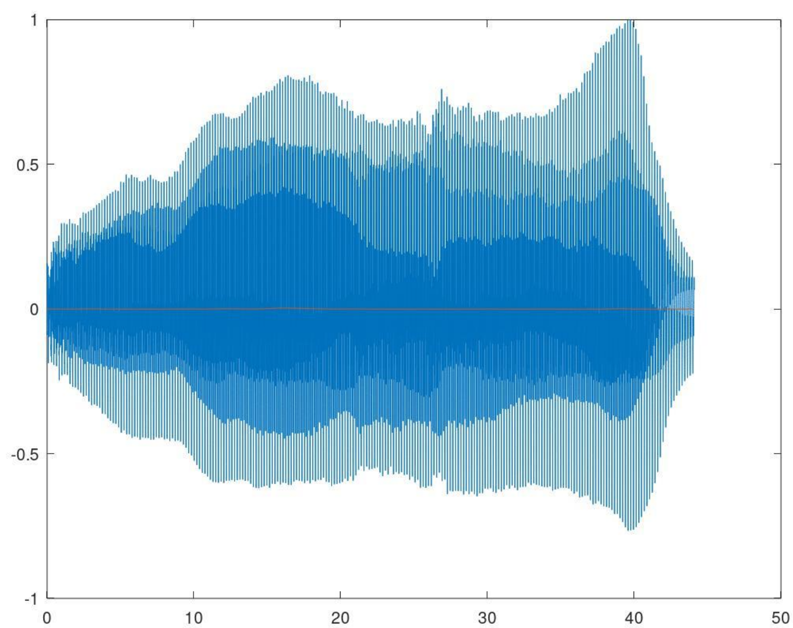


Βλέπουμε ότι η ενέργεια ανεβάζει αισθητά την τιμή της κοντά στα σημεία όπου η ένταση του σήματος επίσης ανεβαίνει.

Για το clarinet:

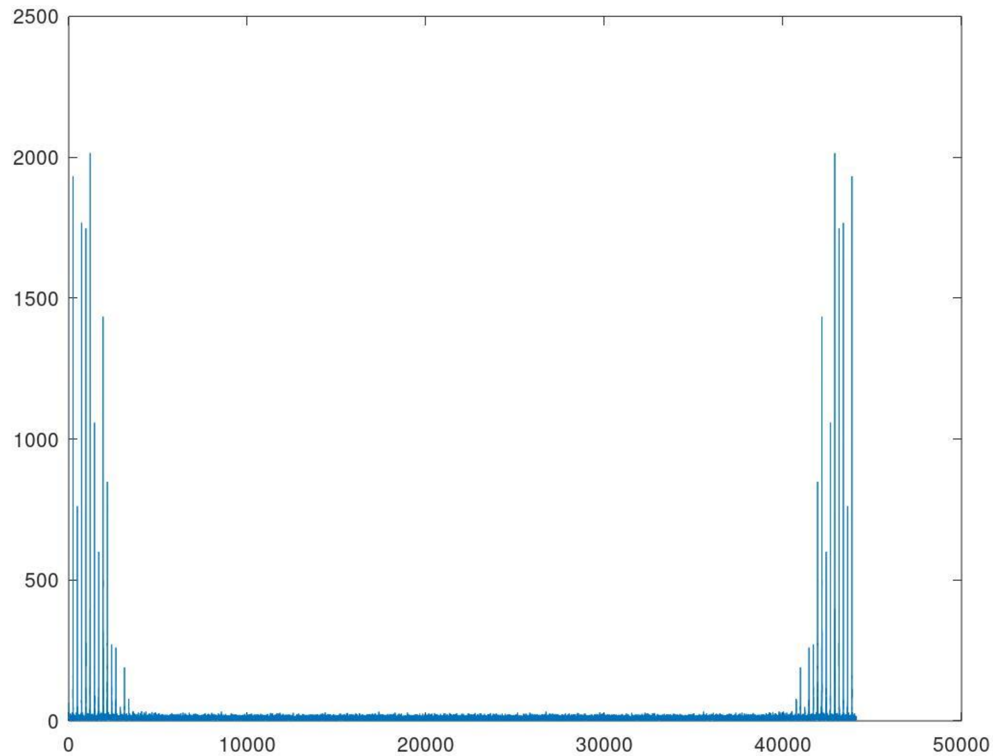


Για το cello:



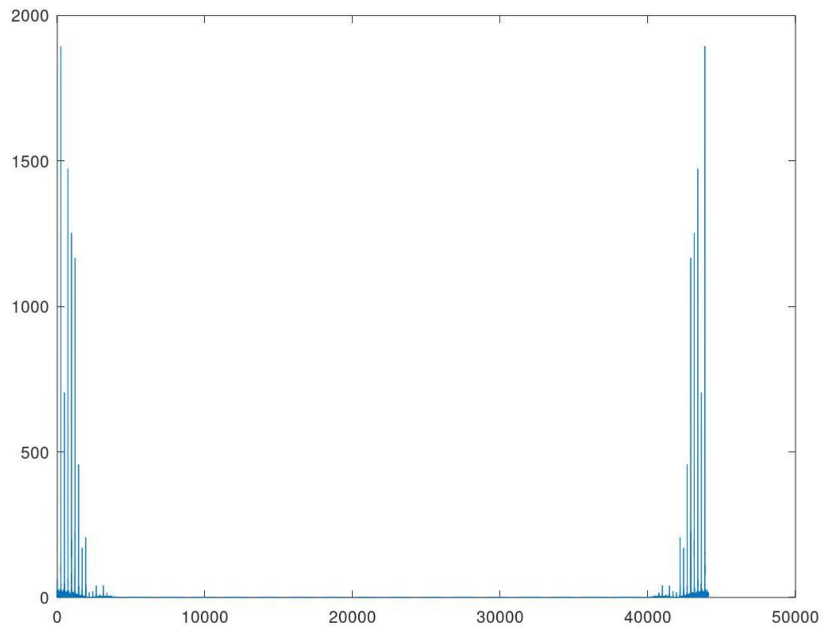
Στα δύο τελευταία σήματα βλέπουμε γενικά χαμηλή ενέργεια (μοιάζει με ευθεία, αλλά ουσιαστικά έχει χαμηλές τιμές μη συγκρίσιμες με τη μονάδα που είναι ο άξονας y στα διαγράμματα αυτά).

ε) Φορτώνουμε το αρχείο με την εντολή audioread('filename.wav'), όπως στο πρώτο ερώτημα και αφού ακούσουμε το σήμα, υπολογίζουμε το φάσμα του σήματος με παρόμοιο τρόπο με τα προηγούμενα ερωτήματα.



Παρατηρώ ότι σε σχέση με το φάσμα του σήματος χωρίς θόρυβο, υπάρχει μέτριος θόρυβος με μικρό κυρίως πλάτος σε όλο το μήκος του φάσματος του σήματος.

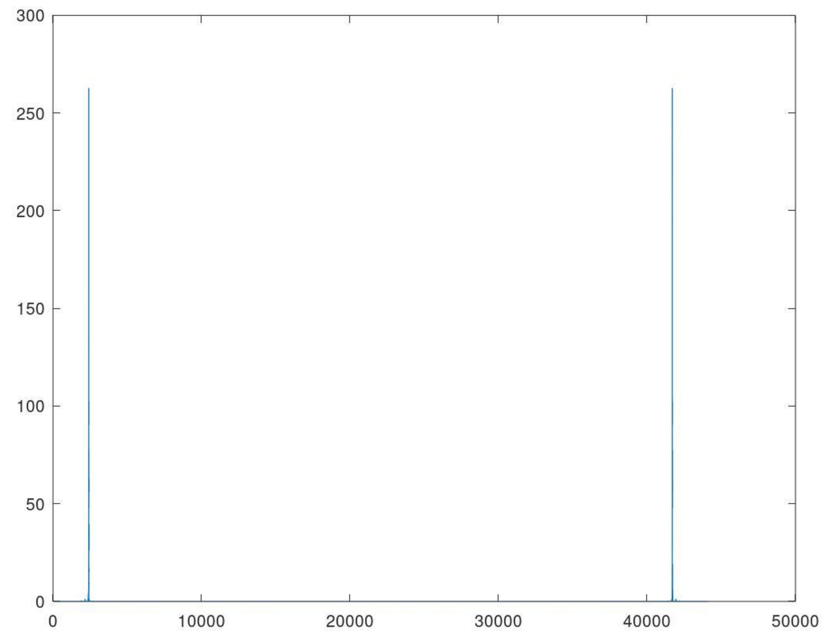
ζ) Θα κατασκευάσουμε ένα βαθυπερατό φίλτρο κινούμενου μέσου με τη βοήθεια της συνάρτησης filter(b,a,y) όπου δοσμένων των διανυσμάτων b,a της συνάρτησης μεταφοράς φιλτράρει το σήμα y. Έπειτα θα υπολογίσουμε τον μετασχηματισμό του φιλτραρισμένου σήματος με το γνωστό τρόπο και θα το ακούσουμε με την εντολή sound(f,fs). Για την τιμή του παραθύρου w έπειτα από μερικούς πειραματισμούς κατέληξα στην τιμή 20, η οποία μειώνει σημαντικά τον θόρυβο, μειώνει όμως και το πλάτος – ένταση του σήματος σχετικά.

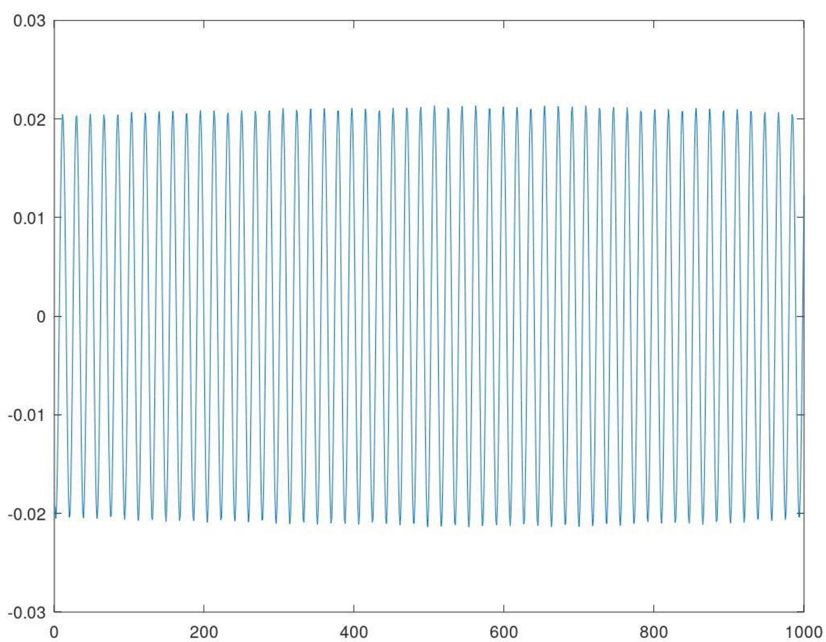


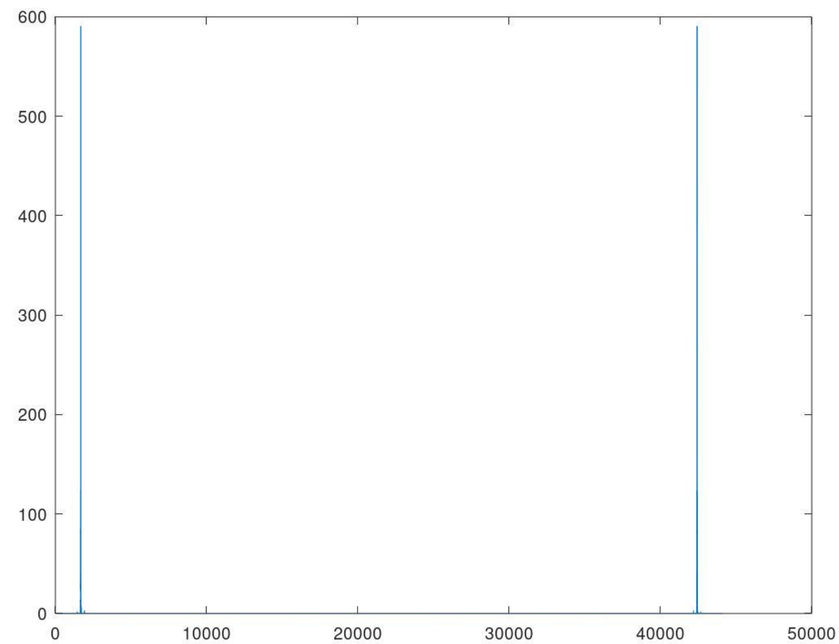
Το φίλτρο που σχεδιάσαμε κατάφερε να αφαιρέσει το μέτριο θόρυβο που υπήρχε σε όλο το μήκος του φάσματος, κρατώντας μόνο λίγο κοντά στις συχνότητες που βρίσκουμε τις αρμονικές, οι οποίες όμως έχουν συγκριτικά πολύ μεγαλύτερο πλάτος. Ο ήχος από το φιλτραρισμένο σήμα πλησιάζει αρκετά την καθαρή νότα του cello.

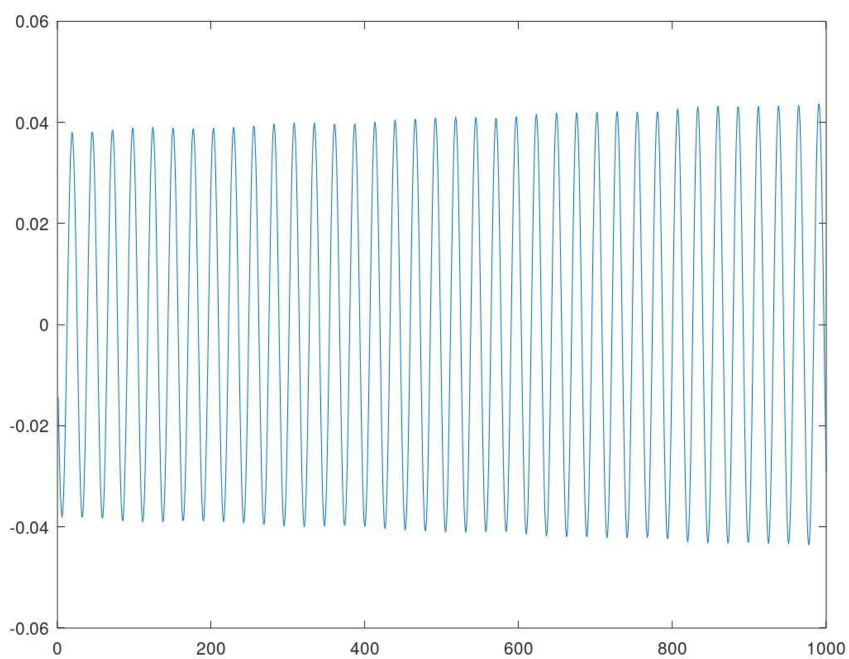
στ) Με χρήση της συνάρτησης butter(n,k), όπου n ο βαθμός του φίλτρου και k το διάνυσμα με τις συχνότητες αποκοπής σε radians και με την κατάλληλη μετατροπή που απαιτεί το όρισμα της συνάρτησης, θα λάβουμε τα διανύσματα της συνάρτησης μεταφοράς b,a ενός ζωνοπερατού φίλτρου Butter και στη συνέχεια με τη χρήση της γνωστής filter(b,a,y) θα απομονώσουμε αρχικά την $9^{η}$ αρμονική και στη συνέχεια την $6^{η}$ αρμονική. Επίσης πλοτάρουμε τα φάσματα του μετασχηματισμού και ένα απόσπασμα στο πεδίο του χρόνου για το κάθε σήμα.

Ακολουθούν η απομόνωση της $9^{ης}$ αρμονικής, το απόσπασμα του σήματος αυτού στο χρόνο, η απομόνωση της $6^{ης}$ αρμονικής και το αντίστοιχο απόσπασμα του σήματος στο χρόνο. Βλέπουμε ότι στα δύο αποσπάσματα έχουμε ουσιαστικά ένα σήμα που μοιάζει με ένα μοναδικό ημιτονοειδές, όπως άλλωστε θα περιμέναμε μετά την απομόνωση μίας αρμονικής συχνότητας, η οποία αντιστοιχεί μέσω και της ανάλυσης Fourier σε ένα συνημιμόνο με κατάλληλο συντελεστή πλάτους και κατάλληλη φάση.


300
250
200
150
100
50
0
0
10000
20000
30000
40000
50000


0.03
0.02
0.01
0
-0.01
-0.02
-0.03
0
200
400
600
800
1000


600
500
400
300
200
100
0
0
10000
20000
30000
40000
50000


0.06
0.04
0.02
0
-0.02
-0.04
-0.06
0
200
400
600
800
1000

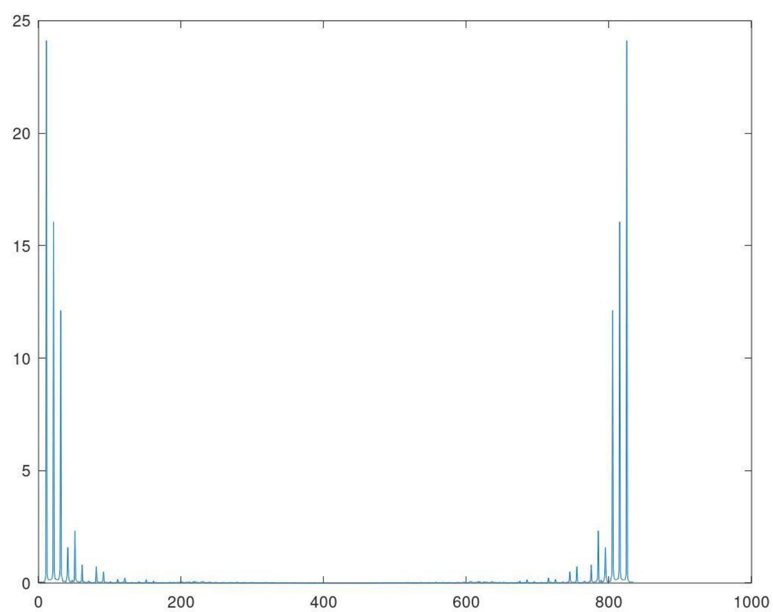
2.2 Σύνθεση Μουσικών Σημάτων ως Άθροισμα Ημιτονοειδών

α) Επιλέγω το αρχείο με όνομα ‘flute_note.wav’ και διαλέγω από αυτό ένα τυχαίο τμήμα με μήκος 10 φορές την θεμελιώδη περίοδο του (10/529), δηλαδή 834 δείγματα.

β) Υπολογίζω τον διακριτό μετασχηματισμό Fourier μέσω της εντολής fft(x) και πλοτάρω το πλάτος του σε άξονα δειγμάτων. Μετά για κάθε αρμονική που έχει ένα σημαντικό πλάτος υπολογίζω τον συντελεστή c_n. Επομένως, με βάση τον πίνακα s που περιέχει τα πλάτη των αρμονικών και επειδή έχουμε και μία αρμονική κάθε 10 στοιχεία ξεκινώντας από το 11 υπολογίζω τα cn με βάση το s(10*n+1)/s(11):

c1 = 1, c2 = 0.66587, c3 = 0.50252, c4 = 0.065599, c5 = 0.096453, c6 = 0.033759, c7 = 0.0023831, c8 = 0.030432, c9 = 0.021109

Και για το πλάτος του μετασχηματισμού Fourier έχουμε:

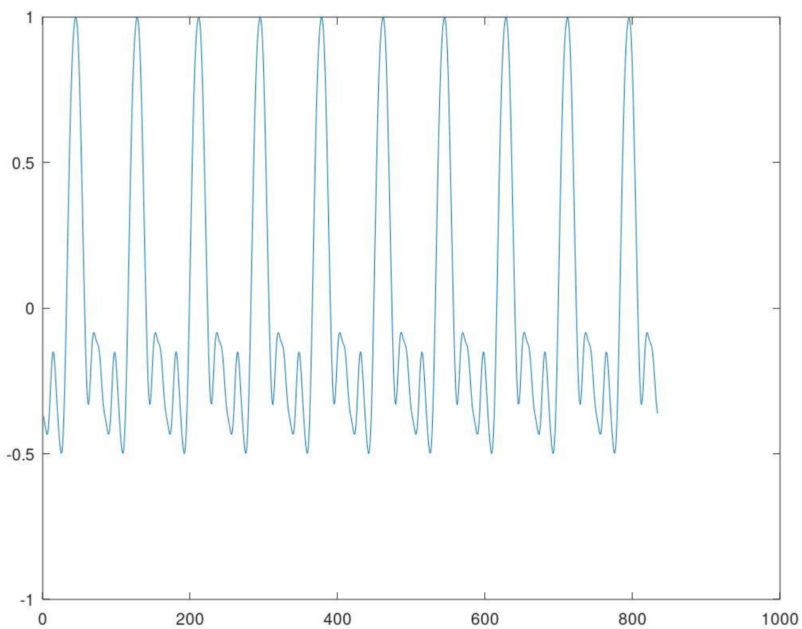


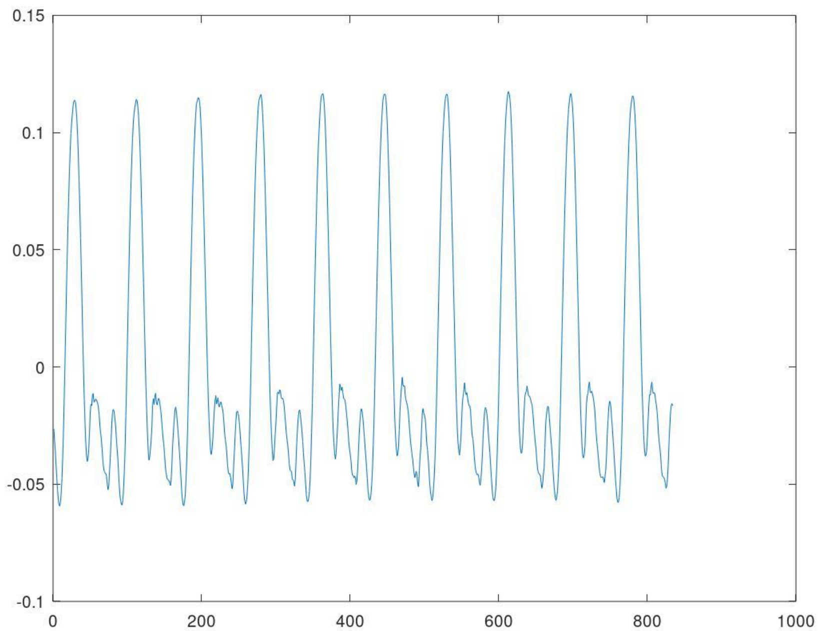
γ) Με χρήση της εντολής angle(x) υπολογίζω για κάθε peak την φάση του μετασχηματισμού στο διάστημα [-π,π]. Με βάση τον πίνακα a που περιέχει τον μετασχηματισμό, θα υπολογίζουμε τα αντίστοιχα fn ξεκινώντας με το 11:

f1 = -2.2353, f2 = 2.1756, f3 = 0.13884, f4 = 1.0532, f5 = -0.28843, f6 = 2.1883, f7 = 2.2669, f8 = 1.5103, f9 = 3.0457 δ) Η μεταβλητή χρόνου t έχει διάρκεια 2 δευτερόλεπτα και έχει fs δείγματα ανά δευτερόλεπτο. Προσθέτουμε όλα τα ημιτονοειδή με βάση τους υπολογισμούς των συντελεστών στα ερωτήματα β,γ.

ε) Λαμβάνουμε ένα απόσπασμα από το ανακατασκευασμένο, ίδιου μήκους με το αρχικό απόσπασμα και ελέγχουμε οπτικά τα δύο αυτά σήματα. Μοιάζουν αρκετά και θεωρούμε πως έχουμε μια σχεδόν τέλεια ανακατασκευή, σίγουρα καλύτερη από αυτή με την κύρια αρμονική και τις 5 πρώτες, καθώς εμείς πήραμε την κύρια και τις 9 πρώτες. Ο τόνος είναι επίσης παρόμοιος, αρκεί να διαιρέσουμε το πλάτος του ανακατασκευασμένου με το 10, ώστε να έχουμε ίδιο πλάτος με το αρχικό απόσπασμα.

Ακολουθεί το απόσπασμα από το ανακατασκευασμένο και στη συνέχεια το αρχικό απόσπασμα:





στ) Με χρήση της εντολής audiowrite('filename.wav',x,fs) γράφουμε σε ένα αρχείο reconstructed.wav το ανακατασκευασμένο σήμα που δημιουργήσαμε στα προηγούμενα ερωτήματα.